sid.inpe.br/mtc-m21c/2021/03.11.20.53-TDI

RESTAURAÇÃO DE IMAGENS ASTRONÔMICAS UTILIZANDO REDES NEURAIS, WAVELETS E REGULARIZAÇÃO

Vinicius Schmidt Monego

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Haroldo Fraga de Campos Velho, e Alice de Jesus Kozakevicius, aprovada em 04 de março de 2021.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/44ARSBE>

INPE
São José dos Campos
2021

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Coordenação de Ensino, Pesquisa e Extensão (COEPE)
Divisão de Biblioteca (DIBIB)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

sid.inpe.br/mtc-m21c/2021/03.11.20.53-TDI

RESTAURAÇÃO DE IMAGENS ASTRONÔMICAS UTILIZANDO REDES NEURAIS, WAVELETS E REGULARIZAÇÃO

Vinicius Schmidt Monego

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Haroldo Fraga de Campos Velho, e Alice de Jesus Kozakevicius, aprovada em 04 de março de 2021.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/44ARSBE>

INPE
São José dos Campos
2021

Dados Internacionais de Catalogação na Publicação (CIP)

Monego, Vinicius Schmidt.

M745r Restauração de imagens astronômicas utilizando redes neurais, wavelets e regularização / Vinicius Schmidt Monego. – São José dos Campos : INPE, 2021.

xx + 103 p. ; (sid.inpe.br/mtc-m21c/2021/03.11.20.53-TDI)

Dissertação (Mestrado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2021.

Orientadores : Drs. Haroldo Fraga de Campos Velho, e Alice de Jesus Kozakevicius.

1. Problemas inversos. 2. Restauração de imagens. 3. Ruído. 4. Regularização. 5. Transformada wavelet. I.Título.

CDU 004.82:520.3

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

Serviço de Pós-Graduação - SEPGR

DEFESA FINAL DE DISSERTAÇÃO DE VINICIUS SCHMIDT MONEGO
BANCA Nº 033/2021, REG 158003/2019

No dia 04 de março de 2021, as 09h, por teleconferência, o(a) aluno(a) mencionado(a) acima defendeu seu trabalho final (apresentação oral seguida de arguição) perante uma Banca Examinadora, cujos membros estão listados abaixo. O(A) aluno(a) foi APROVADO(A) pela Banca Examinadora, por unanimidade, em cumprimento ao requisito exigido para obtenção do Título de Mestre em Computação Aplicada. O trabalho precisa da incorporação das correções sugeridas pela Banca Examinadora e revisão final pelo(s) orientador(es).

Título: "Restauração de imagens astronômicas utilizando redes neurais, wavelets e regularização"

Eu, Stephan Stephan, como Presidente da Banca Examinadora, assino esta ATA em nome de todos os membros.

Dr. Stephan Stephan - Presidente - INPE
Dr. Haroldo Fraga de Campos Velho - Orientador - INPE
Dra Alice de Jesus Kozakevicius - Orientadora - UFSM/Lana
Dr. Gilberto Ribeiro de Queiroz - Membro Interno - INPE
Dr. Antônio José da Silva Neto - Membro Externo - UERJ
Dra Ana Paula Abrantes Castro Shiguemori - Membro Externo - IFSP

Documento assinado eletronicamente por **Stephan Stephany**, **Pesquisador Titular**, em 05/03/2021, às 10:58 (horário oficial de Brasília), com fundamento no art. 6º do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.mctic.gov.br/verifica.html, informando o código verificador **6577868** e o código CRC **58E4E57B**.

Referência: Processo nº 01340.001374/2021-97 SEI nº 6577868

AGRADECIMENTOS

Agradeço ao meu orientador, Dr. Haroldo Fraga de Campos Velho, pela oportunidade de realizar o curso de Mestrado em Computação Aplicada no Instituto Nacional de Pesquisas Espaciais e à CAPES pelo suporte financeiro que tornou possível aproveitar essa oportunidade. Agradeço também à Dra. Alice de Jesus Kozakevicius, com quem trabalho desde 2013 e que sempre me apoiou desde então.

RESUMO

Neste trabalho são estudadas diferentes técnicas de restauração de imagens, envolvendo métodos de regularização, filtros *wavelets* e redes neurais. Mais especificamente, as técnicas escolhidas foram regularização por Tikhonov, regularização por entropia, regularização da variação total, filtro de Wiener, filtragem wavelet, redes neurais convolucionais e filtro neural multiescala. As imagens de interesse são imagens astronômicas, provenientes da fonte HubbleSite, que disponibiliza imagens do telescópio espacial Hubble sob uma licença compatível a domínio público. As imagens são degradadas com ruído gaussiano de desvio padrão de 5%, 15% e 25%. A performance de cada um dos métodos de restauração é avaliada através das métricas: NRMSE (*Normalized Root-Mean-Square Error* Erro Médio Quadrático Normalizado), PSNR (*Peak Signal-to-Noise Ratio* – Razão de pico sinal-ruído) e SSIM (*Structural Similarity Index Measure* — Medida do índice de similaridade estrutural).

Palavras-chave: Problemas inversos. Restauração de imagens. Ruído. Regularização. Transforwada *wavelet*. Redes neurais.

ASTRONOMICAL IMAGE RESTORATION USING NEURAL NETWORKS, WAVELETS AND REGULARIZATION

ABSTRACT

In this work, different image restoration techniques are studied, involving regularization methods, wavelet filters, and neural networks. More specifically, the techniques chosen were regularization by Tikhonov, regularization by entropy, total variation regularization, Wiener filter, wavelet filtering, convolutional neural networks and multiscale neural filter. The images of interest are astronomical images from the HubbleSite source, which makes images from the Hubble space telescope available under a license compatible with the public domain. The images are degraded with standard deviation Gaussian noise of 5%, 10% and 15%. The performance of each of the restoration method is evaluated using the following metrics: Normalized Root-Mean-Squared Error (NRMSE), Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

Keywords: Inverse problems. Image restoration. Noise. Regularization. Wavelet Transform. Neural networks.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

AI	–	Artificial Intelligence
BS	–	BayesShrink
CC	–	Cognitive Computing
CNN	–	Convolutional Neural Network
DL	–	Deep Learning
EC	–	Evolutionary Computing
FNME	–	Filtro Neural Multi Escala
ML	–	Machine Learning
MLP	–	Multi-Layer Perceptron
MPCA	–	Multi-Particle Collision Algorithm
MSE	–	Mean Squared Error
MSNF	–	Multiscale Neural Filter
NLP	–	Natural Language Processing
PSNR	–	Peak Signal-to-Noise Error
SSIM	–	Structural Similarity Index Measure ou Structural SIMilarity
TW	–	Transformada Wavelet
VS	–	VisuShrink

LISTA DE SÍMBOLOS

A – modelo de degradação
u – imagem restaurada
f^{δ} – imagem degradada
α – parâmetro de regularização
Ω – operador de regularização
ϕ – função de escala *wavelet*
ψ – função *wavelet*
λ – valor de corte
σ – desvio padrão do ruído gaussiano
σ^2 – variância do ruído gaussiano
n – momentos nulos da *wavelet*
w – pesos das conexões da rede neural

SUMÁRIO

Pág.

1 INTRODUÇÃO

Restauração de imagens é uma disciplina da área de problemas inversos e consiste em aproximar, a partir de um sinal de imagem degradado obtido experimentalmente, o sinal original que gerou este modelo (LU; GUO, 2016). Degradação é um termo que engloba vários fatores que diminuem a qualidade de uma imagem. Dentre estes fatores pode-se citar principalmente ruído e borramento. Outros exemplos de degradação podem ser texto (LEHTINEN et al., 2018) e marcas d'água (CUN; PUN, 2020). No caso de documentos antigos a serem digitalizados, degradações podem ocorrer por ranhuras, poeira e manchas (SOUIBGUI; KESSENTINI, 2021).

Destacam-se duas áreas científicas principais que impulsionam o desenvolvimento de métodos de restauração de imagens: área espacial e área médica. Neste trabalho, serão exploradas algumas técnicas de restauração de imagens em aplicações na área espacial, envolvendo transformadas *wavelet*, teoria de regularização e redes neurais.

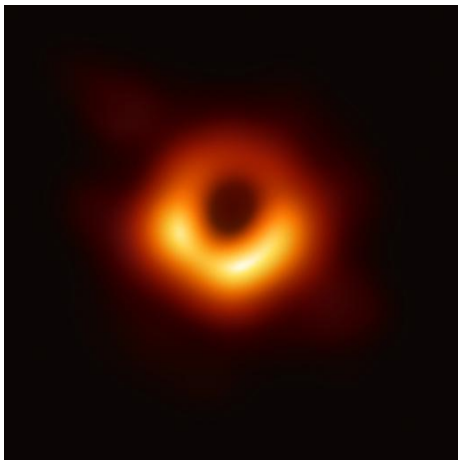
1.1 Motivação

Em aplicações da área espacial, um evento marcante ocorreu no dia 10 de abril de 2019: foi obtida a primeira imagem de um buraco negro a partir de dados de radiofrequência obtidos do Event Horizon Telescope (EHT), uma rede global de radiotelescópios. A Figura 1.1 é a reconstrução destes dados de radiofrequência, combinando as coletas individuais de todos os telescópios participantes do EHT. O que se vê na Figura 1.1 é a sombra do buraco negro projetada em seu anel de fótons.

Vários algoritmos foram utilizados na reconstrução da imagem da sombra do buraco negro, como CLEAN (HÖGBOM, 1974) e CHIRP (BOUMAN et al., 2016), baseados em deconvolução e método bayesiano, respectivamente. Apesar da grande conquista que foi a obtenção da imagem, ela saiu borrada devido à enorme distância entre o objeto e os telescópios. Para estudar mais detalhadamente a imagem será necessário aplicar outros métodos de restauração de imagem que já foram estudados.

Outro evento memorável na área espacial foi o lançamento do Telescópio Hubble (HST) (Figura 1.2) no ano de 1990. Porém, devido a um erro de montagem no espelho interno, o telescópio sofreu do problema de aberração esférica, tendo suas imagens capturadas contaminadas por borramento. Como o telescópio se encontrava no espaço quando o problema foi detectado, há dificuldades óbvias na correção do problema. Durante o período de lançamento e o de conserto no ano de 1993, foi necessário encontrar uma solução, que foi a aplicação de técnicas de restauração de

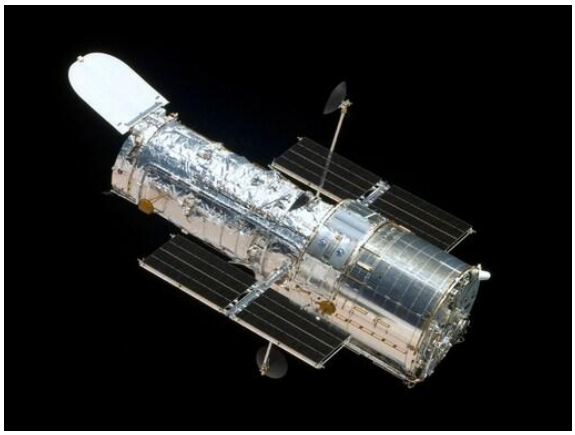
Figura 1.1 - Foto reconstruída de um buraco a partir a partir de dados obtidos pelo EHT.



Fonte: Wikipedia (2020a).

imagens (HANISCH, 1994).

Figura 1.2 - Telescópio Hubble.



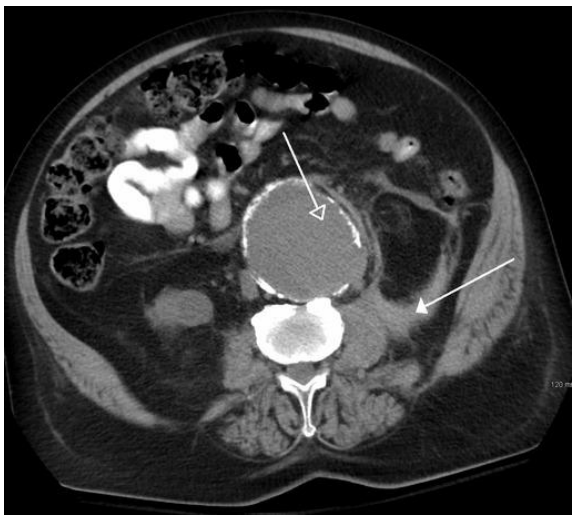
Fonte: Wikipedia (2020d).

Na área médica, o diagnóstico por imagens tem como objetivo principal identificar condições de saúde por meio de imagens. Vários tipos de tomografia costumam ser considerados, dentre os quais citam-se as tomografias de emissão de pósitrons (PET)

e computadorizada (CT). A primeira utiliza a faixa de frequência eletromagnética de raios gama, enquanto a segunda utiliza a faixa de frequência de raio X. A exposição prolongada a essas faixas de frequências causa danos no DNA humano, por isso a exposição humana aos aparelhos deve ser a mínima possível. Entretanto, a exposição mínima possui o efeito colateral de introduzir um nível maior de ruído na imagem resultado do exame (CHEN et al., 2017).

Apesar das restrições, as tomografias computadorizadas são fundamentais no diagnóstico de inúmeras anomalias. A Figura 1.3 mostra a imagem de um corte horizontal do abdômen de um paciente, no qual é identificado um aneurisma de aorta abdominal, segundo Wikipedia (2021a). As setas na imagem representam as anotações dos especialistas. A seta fechada indica o aneurisma e a seta aberta aponta para uma região na qual é identificado o acúmulo de sangue causado pelo aneurisma. As regiões mais brancas na imagem são os órgãos do paciente. Cada informação da

Figura 1.3 - Tomografia computadorizada de um aneurisma de aorta abdominal.



A seta fechada representa o aneurisma e a seta aberta o acúmulo de sangue associado.

Fonte: Wikipedia (2021a).

imagem é muito importante: por exemplo, em análises que tratam do diagnóstico de câncer, cada detalhe da imagem é crucial na distinção entre tumores benignos e malignos, especialmente a identificação de bordas (ASADZADEH et al., 2012). Assim, a restauração de imagens pode ser considerada uma área fundamental também para

o desenvolvimento de métodos computacionais que dão suporte a diagnósticos.

Além das aplicações nas áreas científicas, a restauração de imagem é relevante também na área de entretenimento. A indústria cinematográfica utiliza técnicas de restauração para melhorar a imagem de filmes antigos. A rede neural *DeepRemaster* (Figura 1.4) (IIZUKA; SIMO-SERRA, 2019), por exemplo, realiza restauração em filmes antigos.

Figura 1.4 - Restauração em filmes antigos por meio de rede neural.

Fonte: Iizuka e Simo-Serra (2019).

Outros casos de uso mais específicos também podem ser citados: dispositivos de monitoramento de trânsito que medem a velocidade de um veículo e precisam capturar uma imagem da placa dos veículos que ultrapassam a velocidade limite. Como o veículo se move em alta velocidade, a captura da imagem da placa apresenta borramento (ruído) que precisa ser removido por pós-processamento da imagem (KRUITHOF et al., 2012). Mais aplicações podem ser encontradas nas áreas de agricultura, ciência dos materiais, oceanografia, meteorologia, navegação autônoma por VANTs, dentre outras.

1.2 Objetivos da tese

O objetivo desta dissertação de mestrado é utilizar e avaliar diferentes técnicas de restauração de imagens, com aplicação em imagens astronômicas. As técnicas são avaliadas a partir de imagens degradadas por um ruído branco gaussiano aditivo.

As imagens astronômicas podem ser obtidas por telescópios óticos na superfície da Terra ou por dispositivos espaciais, bem como de outras fontes de dados em outras bandas de frequências.

As técnicas utilizadas em restauração de imagens neste trabalho são baseadas em teoria de regularização, filtragem *wavelet* e redes neurais. As imagens de referência (verdade), degradada e restauradas são mostradas para uma visualização. Além da inspeção visual (subjetiva), as diferentes metodologias de restauração são avaliadas seguindo métricas descritas na Seção 3.3.

2 RESTAURAÇÃO DE IMAGENS

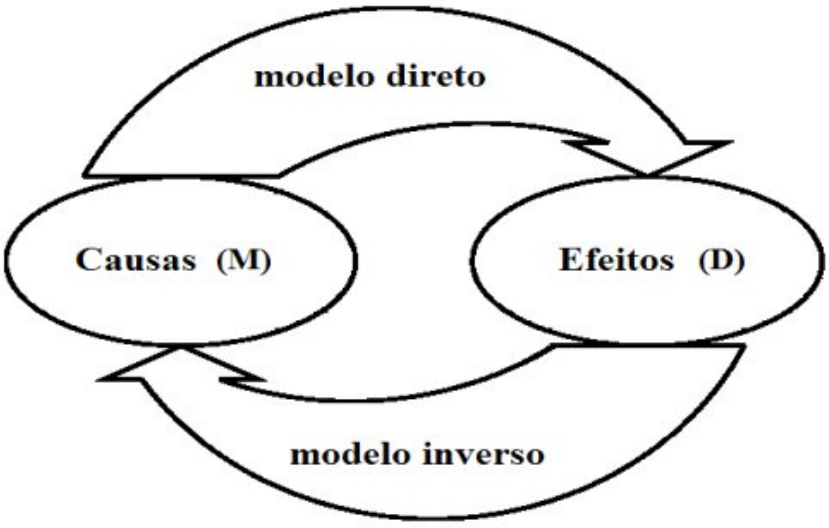
Como mencionado no capítulo anterior, esta é uma área de grande interesse em muitas aplicações e várias técnicas foram desenvolvidas de acordo com as características de cada problema. Neste capítulo, serão descritos somente os métodos de restauração aplicados na pesquisa de mestrado que envolve aplicações em imagens espaciais. O capítulo está estruturado com a descrição dos métodos de regularização na seção 2.1. Uma breve introdução sobre filtragem *wavelet* na seção 2.2 e a redes neurais profundas na seção 2.3.

2.1 Técnicas de regularização

Um problema inverso pode ser definido como sendo a determinação de causas desconhecidas a partir de efeitos observados (CAMPOS VELHO, 2019). A Figura 2.1 representa graficamente os tipos de problema direto e inverso.

No contexto de imagens, o termo "modelo", ou "causa" se refere à imagem original e o termo "efeito" à imagem degradada. Portanto, o problema inverso se refere a obter o modelo através do método inverso a partir do efeito. Os problemas podem ser

Figura 2.1 - Problema direto e problema inverso e suas relações de causa (modelo) e efeito (dados).



Fonte: Campos Velho (2019).

definidos como matematicamente *bem postos*, como em geral notados em problemas diretos, ou *mal postos*, que ocorre de forma sistemática com os problemas inversos. Um problema bem posto é aquele que obedece às três condições de Hadamard (HADARMARD, 1902):

a) A solução existe;

b) A solução é única;

c) A solução é linearmente dependente da condição inicial.

Analogamente, um problema mal posto é aquele que viola qualquer uma dessas condições. Problemas diretos geralmente são problemas bem postos e apresentam uma solução estável. Problemas inversos, ao contrário, quase sempre são problemas mal postos, violando as três condições, e tornam-se mais difíceis de resolver, pois a solução encontrada pode não ser coerente com o problema original (BERTERO, 1998).

Um problema inverso pode ser formulado como um problema de otimização, em que a função custo (f_{cost}) pode ser descrita na forma de mínimos quadrados, como mostrado abaixo:

$$f_{cost} = \min_u \left\| A(u) - f^{\delta} \right\|^2 \tag{2.1}$$

sendo $A(u)$ o modelo direto, u o vetor ou função de termos desconhecidos, f^{δ} representa as medidas ou valor de referência e δ é o nível de ruído das observações.

Porém, a função custo expressa pela Equação 2.1 pode não gerar uma solução inversa estável. Uma maneira de se obter soluções inversas robustas é através da adicionação de uma restrição ao problema de otimização. Este termo ou restrição é aqui representado por $\Omega[u]$:

$$f_{cost} = \min_u \left\| A(u) - f^{\delta} \right\|_2^2 \ , \quad \text{sujeito a: } \Omega[u] \leq \rho \ , \tag{2.2}$$

com ρ sendo um limite superior vinculado ao problema em estudo. A restrição $\Omega[.]$ atua como uma penalidade ao método dos mínimos quadrados.

Em geral, na teoria da regularização, o operador $\Omega[.]$ significa um quantificador da suavidade da função e é chamado de *operador de regularização*. A solução inversa regularizada é a mais suave vinculada ao problema de mínimos quadrados.

Pelo método dos multiplicadores de Lagrange, é possível acoplar a restrição à função objetivo através de um parâmetro α, como mostrado abaixo:

$$f_{cost} = \min_u \left\{ \left\| A(u) - f^\delta \right\|_2^2 + \alpha\Omega[u] \right\} . \tag{2.3}$$

Na Equação 2.3, foi introduzido o parâmetro α, aqui denominado de *parâmetro de regularização*. Este parâmetro expressa o equilíbrio entre o termo de diferença quadrática e a suavidade da solução. Desta forma, quando $\alpha \hookrightarrow 0$, não há regularização. Por outro lado, quando $\alpha \hookrightarrow \infty$, a coerência entre o modelo direto $A[u]$ e a informação experimental f^δ é perdida.

Um dos desafios da teoria de regularização é estimar o valor de α que melhor obtenha um equilíbrio entre a suavidade desejada com a menor discrepância entre o modelo direto e as observações. Uma forma de estimar esse parâmetro é pelo *princípio da discrepância de Morozov* (MOROZOV, 2012).

Este critério é baseado na diferença ou discrepância entre os dados observados e os dados do modelo. Ou seja, a discrepância entre os dados citados deve possuir a mesma magnitude do erro do instrumento de medida. Sendo δ o nível experimental de ruído, tem-se que o valor ótimo de α é a raiz da equação abaixo:

$$\alpha_{opt} = ||A(u) - f^\delta||_2^2 \approx N\sigma^2. \tag{2.4}$$

Como mencionado, um problema inverso pode ser transformado num problema bem posto, quando formulado como um problema de otimização associado a um operador de regularização – ver Equação 2.3. Todavia, se o problema direto for linear, a solução inversa pode ser obtida por decomposição em valores singulares truncada (TSVD: *truncated singular value decomposition*) (HANSEN, 1998). Hansen e Jensen mostraram que a solução inversa por TSVD é equivalente a um filtro de passa baixa (HANSEN; JENSEN, 1998). Como funções wavelets estão associadas a famílias de filtros FIR (DAUBECHIES, 1992), estas funções e suas transformadas serão estudadas e consideradas como metodologia de solução inversa deste trabalho.

Uma abordagem do problema de otimização descrito na Equação 2.3 é tratar o modelo direto $A(u)$ como uma operação de convolução entre uma função chamada de Função de Espalhamento Pontual (*PSF - Point Spread Function*), que representa a câmera, e os parâmetros desconhecidos u.

A PSF é uma imagem obtida a partir da convolução com a cena real, resultando num *espalhamento* (ou *borramento*, ou ainda *blurring* – como descrito na literatura em inglês) da imagem original . Uma função para representar a PSF – $h(x)$ – é a gaussiana $h(x)$:

$$h(x) = ae^{-[(x-\mu)^2/(2\sigma^2)]}, \tag{2.5}$$

sendo μ e σ média e desvio padrão, respectivamente, e a é a amplitude do sinal.

2.1.1 Regularização pelo operador de Tikhonov

Um dos operadores de regularização mais utilizados foi formalizado pelo matemático russo Andrei Tikhonov. Um caso particular da teoria de Tikhonov é o caso de *ridge regression* (HOERL; KENNARD, 1970). O operador de regularização de Tikhonov é expresso pela equação abaixo em que k representa a ordem da derivada em problemas contínuos ou a ordem do operador de diferenças no caso discreto, em que k define a ordem da regularização e é denotada por:

$$\Omega[u] = \sum_k \alpha_k \, \|u^{(k)}\|_2^2, \tag{2.6}$$

sendo α_k distintos paramêtros de regularização. A ordem da derivada ou operador de diferença é por vezes denotada por *Tikhonov-k*, para indicar a ordem do operador de regularização aplicado.

Deste modo, a solução inversa é obtida minimizando o funcional

$$J(u) = \frac{1}{2}\|h \star u - f^\delta\|_2^2 + \frac{1}{2}\sum_k \alpha_k \|L_k \, u\|_2^2. \tag{2.7}$$

Aqui o símbolo " $\star$ " denota a convolução[1] de uma PSF h com a imagem u e L_k é a derivada ou matriz de diferença finita (caso discreto) de k-ésima ordem. Pelo teorema da convolução, a transformada de Fourier das funções convoluídas é dada pelo produto das funções no espaço de Fourier. Deste modo,

$$\mathcal{F}\{h \star u\} = \hat{H}\hat{U}, \tag{2.8}$$

sendo $\hat{H}$ e $\hat{U}$ as transformadas de Fourier das funções h e u, respectivamente.

A solução ótima para para o funcional da Equação 2.7 pode ser obtida tomando-se

[1] Convolução entre as funções h e u: $h(x,y) \star u(x,y) = \int\int h(x-\tau, y-\zeta)\, f(\tau,\zeta)\, d\tau d\zeta$.

o gradiente do funcional e igualando a zero, que para o caso discreto resulta em:

$$\frac{\partial J(U)}{\partial U} = \left[H^T\, H + \alpha_k\, (L_k^T\, L_k)\right] U - H^T F^T = 0\ . \tag{2.9}$$

Na equação acima, são consideradas as formas discretas dos operadores, com regularizador de Tikhonov-k, desta forma:

$$H = [h_{injm}]\ ,\quad L_k = [L_{injm}]\ ,\quad F = [f_{ij}^{\delta}]\ ,\quad U = [u_{ij}]\ ,$$

na qual todas as matrizes envolvidas são consideradas matrizes circulantes (DAVIS, 1979). Contudo, a PSF pode ser assumida como invariante sob translação espacial e, neste caso, pode ser representada por uma função de duas variáveis: $h(x, \tau, y, \zeta) = h(x - \tau, y - \zeta)$. Desta forma, tomando a transformada de Fourier:

$$\mathcal{F}\{H \star U]_{ij} = \mathcal{F}\left\{\sum_{n=0}^{N}\sum_{m=0}^{M} h_{i-n,j-m}\, f_{nm}^{\delta}\right\}\ . \tag{2.10}$$

Aplicando a transformada discreta de Fourier na Equação 2.9, ou seja, mapeando para o domínio da frequência, tem-se:

$$\left[\hat{H}^*\hat{H} + \alpha\, L_k^T L_k\right]\hat{U} = \hat{H}^*\,\hat{F}^{\delta}, \tag{2.11}$$

sendo que o superescrito "$*$" denota o complexo conjugado da matriz transposta. Assim, a imagem restaurada é calculada pela transformada inversa:

$$U = \mathcal{F}^{-1}\left\{\left[\frac{\hat{H}^*}{|\hat{H}|^2 + \alpha|L_k|^2}\right]\,\hat{F}^{\delta}\right\}. \tag{2.12}$$

O operador de Tikhonov pode ser representado por matrizes bloco, onde o bloco que vai realizar a convolução depende da ordem da regularização. A equação abaixo mostra os blocos das sub-matrizes que correspondem a cada ordem do operador de Tikhonov (T0: ordem-0, T1: ordem-1, T2: ordem-2):

$$T0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\ ,\qquad T1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -2 & 0 \\ 0 & 0 & 0 \end{bmatrix}\ ,\qquad T2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}\ . \tag{2.13}$$

2.1.2 Regularização entrópica

Outro operador de regularização é baseado na entropia da teoria de comunicação de Claude Shannon. Uma das aplicações em problemas inversos foi descrita por Gull & Daniel em restauração de imagens – ver (GULL; DANIELL, 1978). Similar à regularização de alta ordem do operador de Tikhonov, o grupo de computação científica do INPE desenvolveu para este operador a formulação de entropia de alta ordem (CAMPOS VELHO; RAMOS, 1997; RAMOS et al., 1999; CAMPOS VELHO et al., 2006).

O operador de entropia tem como argumento o valor s_q, que é função de $r_q^{(k)}$ e este por sua vez depende da ordem do operador de entropia. Para a ordem zero, $r_q^{(0)} = u_q$, sendo u_q o valor dos *pixels* da imagem. A expressão matemática do operador de entropia de alta ordem $S[u]$ é dada por:

$$S[u] = -\sum_{q=1}^{N_q} s_q \, \log(s_q) \; ; \quad s_q = \frac{r_q^{(k)}}{\sum_{q=1}^{N_q} r_q^{(k)}} \; . \tag{2.14}$$

O expoente do fator $r_q^{(k)}$ expressa a ordem do operador entrópico, similar à regularização de Tikhonov de ordem-k. Explicitamente, o fator $r^{(k)}$ até a segunda ordem é definido para problemas discretos com o uso de operadores de diferença finita:

$$r_q^{(k)} \equiv \begin{cases} u_q & \text{para } k = 0 \\ u_{q+1} - u_q + (u_{\max} - u_{\min}) + \varsigma & \text{para } k = 1 \\ u_{q+1} - 2u_q + u_{q-1} + 2(u_{\max} - u_{\min}) + \varsigma & \text{para } k = 2, \end{cases}$$

sendo ς um parâmetro com valor absoluto próximo de zero ($\varsigma \leq O(10^{-8})$).

Entretanto, o operador de entropia é não-linear e portanto, mesmo para um problema direto linear, não há uma solução inversa explícita para este operador. Desta forma, a regularização por entropia requer a solução numérica do problema de otimização.

Alguns métodos de solução (como os esquemas determinísticos) requerem somente uma estimativa inicial, enquanto outros (como algoritmo genético) podem requerer uma população de soluções inversas candidatas. Como o problema inverso possui muitas variáveis (número de pixels da imagem), os métodos baseados em gradiente podem ter convergência mais rápida. Uma boa condição inicial para os métodos baseados em gradiente é a própria imagem com ruído, visto que esta pode ser considerada uma aproximação da imagem original.

A Figura 2.2 mostra a mesma imagem visualizada com diferentes níveis de ruído e pode-se notar que a imagem tem quase o mesmo conteúdo de informação independentemente do nível de ruído associado. Contudo, esta é uma avaliação meramente visual. Outras formas de análise evidenciam com mais facilidade as diferenças entre imagens degradadas e imagens sem ruído, como mostrado por Cidade e Silva Neto (CIDADE et al., 2012). Estes autores apresentam perfis de amplitude traçados ao longo de uma determinada linha da imagem, o que permite comparar os diferentes níveis de ruído para os pixeis selecionados – ver Figura 5.3 da página 31 (CIDADE et al., 2012).

Figura 2.2 - Comparação visual de uma imagem e três níveis de ruído.

Fonte: Produção do autor.

2.1.3 Método da variação total

O método da variação total (TVR: *total variation regularization*) foi proposto por Rudin, Osher e Fatemi (RUDIN et al., 1992) – ROF – para solucionar o problema da suavização excessiva causada pela regularização baseada em normas L_2 (como a regularização de Tikhonov), principalmente nas bordas da imagem (DURAN et al., 2013). A solução foi resolver o problema de minimização baseada em outra métrica. Ou seja, o objetivo é minimizar o funcional:

$$J(u) \equiv \int_D |\nabla u(x,y)| \, \mathrm{d}x \, \mathrm{d}y = \int_D \sqrt{u_x^2 + u_y^2} \, \mathrm{d}x \, \mathrm{d}y \tag{2.15}$$

onde o subíndice denota derivada parcial. Na versão discreta, o operador acima é calculado com a soma finita do gradiente (∇) – operador de diferença finita de primeira ordem – da imagem u.

A Equação 2.15 é sujeita às restrições em que a média da imagem restaurada u deve ser a mesma da imagem degradada f – ver Equação 2.16 e a diferença quadrática

entre as duas imagens u e f deve estar balanceada com nível de ruído branco gaussiano de variância σ^2 multiplicado pela dimensão da imagem $|D|$ (número de pixels) – ver Equação 2.17 (CHAMBOLLE; LIONS, 1997):

$$\int_D u(x)dx = \int_D f(x)dx \ , \tag{2.16}$$

$$\int_D |u(x) - f(x)|^2 dx = \sigma^2 \ |D| \ . \tag{2.17}$$

Na verdade, as restrições acima são informações *a priori* adicionadas ao problema de inversão.

Chambolle e Lions demonstraram que minimizar o funcional da Equação 2.15 sujeito às condições dadas pelas Equações 2.16 e 2.17 equivale a minimizar o seguinte funcional, em que α é o parâmetro de regularização:

$$J(u) = \|\nabla u\|_1^2 + \frac{\alpha}{2}\|u - f^\delta\|_2^2. \tag{2.18}$$

Como a norma L_1 envolve o valor absoluto da função[2], a solução ótima envolve otimização não diferenciável. No entanto, uma opção simples é considerar a norma L_1 para todos os termos do funcional acima, tornando viável a solução de mínimos quadrados (ASTER et al., 2018, Sec 7.3).

A solução proposta por ROF baseia-se na modelagem por uma equação diferencial parcial (RUDIN et al., 1992), a partir da solução da equação de Euler-Lagrange do problema de minimização do funcional da Equação 2.18:

$$-\alpha\nabla \cdot \frac{\nabla u}{|\nabla u|} + (u - f^\delta) = 0 \tag{2.19}$$

com condição de contorno de Neumann: $\partial u/\partial \vec{n}|_{\partial D} = 0$. A Equação 2.19 é um problema não linear de difícil resolução e a solução pode não ser computacionalmente eficiente, como no caso da técnica de elementos finitos. Um algoritmo iterativo foi proposto por Chambolle (CHAMBOLLE, 2004), denominado *Chambolle Projection Algorithm*, e será usado neste trabalho. O algoritmo calcula a solução u como a diferença entre a imagem degradada e uma projeção:

$$u = f^\delta - p, \tag{2.20}$$

onde a projeção p é calculada por um processo iterativo. Abaixo, são definidas as

[2]Norma L_1: $\|u\|_1^2 = \int_D |u(x)|dx$. Norma L_2: $\|u\|_2^2 = \int_D u^2(x)dx$.

funções discretas bidimensionais do gradiente e do divergente.

Operador gradiente:

$$\nabla v|_{(i,j)} \equiv [\partial_x v(i,j), \partial_y v(i,j)] \tag{2.21}$$

$$\partial_x v(i,j) = \begin{cases} v(i+1,j) - v(i,j) \ , & \text{para: } i < N \\ 0 \ , & \text{para: } i = N \end{cases} \tag{2.22}$$

$$\partial_y v(i,j) = \begin{cases} v(i,j+1) - v(i,j) \ , & \text{para: } j < N \\ 0 \ , & \text{para: } j = N \end{cases} \tag{2.23}$$

Operador divergente – sendo $v = (v_1, v_2)$:

$$\begin{aligned} \nabla \cdot v|_{(i,j)} \equiv & \begin{cases} v_1(i,j) \ , & \text{se: } i = 1 \\ -v_1(i,j) \ , & \text{se: } i = N \\ v_1(i,j) - v_1(i-1,j) \ , & \text{se: } i \neq 1, N \end{cases} \\ & + \begin{cases} v_2(i,j) \ , & \text{se: } j = 1 \\ -v_2(i,j) \ , & \text{se: } j = N \\ v_2(i,j) - v_1(i,j-1) \ , & \text{se: } j \neq 1, N \end{cases} \end{aligned} \tag{2.24}$$

O processo iterativo para o cálculo da projeção $p(i,j)$ é dado pela expressão abaixo (DURAN et al., 2013):

$$p_m^{n+1} = \frac{p_m^n + \delta_t \nabla(\nabla \cdot p_m^n - \alpha f_m^\delta)(i,j)}{1 + \delta_t \sqrt{\sum_{m=1}^{M} |\nabla(\nabla \cdot p_m^n - \alpha f_m^\delta)(i,j)|^2}}. \tag{2.25}$$

sendo δ_t um parâmetro livre e a estimativa inicial para a projeção p é fixada como $p_m^0 = 0$. A interrupção do processo iterativo ocorre quando a seguinte desigualdade é satisfeita:

$$\max_{1 \leq i,\, j \leq N} \{|p^{n+1}(i,j) - p^n(i,j)|\} \leq \epsilon \tag{2.26}$$

sendo ϵ a tolerância requerida.

2.1.4 Filtro de Wiener

Este filtro foi proposto por Norbert Wiener na década de 1940 e publicado em 1949 (WIENER, 1950) e tem como proposta a redução do ruído presente em um sinal. A versão discreta equivalente foi publicada por Andrey Kolmogorov em 1941 (KOLMOGOROV, 1978) e essa teoria ficou conhecida como filtro de Wiener-Kolmogorov, mas frequentemente abreviada como filtro de Wiener. Este foi o primeiro filtro estatís-

tico a ser proposto, seguido pelo filtro de Kalman e filtro de partículas. No filtro de Wiener, o projeto é baseado em minimizar o valor esperado entre a imagem original e a imagem restaurada por um estimador de mínimos quadrados com hipótese de ruído gaussiano – processo estocástico gaussiano.

A imagem degradada pode ser expressa como:

$$f^{\delta}(x,y) = h(x.y) \star u(x,y) + n(x,y), \tag{2.27}$$

sendo $n(x,y)$ um ruído branco guassiano. O objetivo é estimar um filtro $w(x,y)$ que minimize o valor esperado do erro entre a imagem original $u(x,y)$ e a imagem restaurada $u_w(x,y)$:

$$e^2(u_w) = \int_x \int_y \left[u(x,y) - u_w(x,y)\right]^2 \, dx\, dy \ . \tag{2.28}$$

Pelo teorema de Parseval:

$$\int_x \int_y \left[u(x,y) - u_w(x,y)\right]^2 \, dx\, dy = \int_r \int_s \left[\hat{U}(r,s) - \hat{U}_W(r,s)\right]^2 \, dr\, ds, \tag{2.29}$$

sendo $\hat{U}$ e $\hat{U}_W$ as tranformadas de Fourier das imagens $u(x,y)$ e $u_w(x,y)$. A imagem é restaurada aplicando o filtro sobre a imagem degradada: $u_w(x,y) = w(x,y) \star f^{\delta}(x,y)$, ou ainda, no domínio da frequência: $\hat{U}_W(r,s) = W(r,s)\, \hat{F}^{\delta}(r,s)$. Portanto, o filtro W é estimado minimizando o erro quadrático:

$$\min e^2(W) \implies \frac{\partial e^2(W)}{\partial W} = \frac{\partial \left\{ \left[\hat{U} - (W\, \hat{H}\, \hat{U} - W\hat{N})\right]^2 \right\}}{\partial W} = 0 \ . \tag{2.30}$$

E a derivada do valor esperado é dada por:

$$\begin{aligned}
\frac{\partial e^2(W)}{\partial W} &= \frac{\partial}{\partial W} \left\{ \sum_r \sum_s \left| (1 - W\hat{H})\hat{U} - W\hat{N} \right|^2 \right\} && (2.31)\\
&= \frac{\partial}{\partial W} \left\{ \sum_r \sum_s \left[|(1 - W\hat{H})\hat{U}|^2 - |W\hat{N}|^2 \right] \right\} && (2.32)\\
&= 2\left[-(1 - W^*\hat{H}^*)\hat{H}|\hat{U}|^2 + W^*|\hat{N}|^2 \right] = 0 && (2.33)
\end{aligned}$$

e a igualdade da Equação 2.32 é justificada porque u e n são decorrelacionados. O filtro $W(r,s)$ que minimiza o erro quadrátido é então expresso por:

$$W(r,s) = \frac{\hat{H}^*(r,s)\, S_{uu}(r,s)}{|\hat{H}(r,s)|^2\, S_{uu}(r,s) + S_{nn}(r,s)} \ . \tag{2.34}$$

Na expressão acima, S_{uu} e S_{nn} são os espectros de potência da imagem e do ruído, respectivamente. Porém, a forma mais conhecida do filtro de Wiener é dada por (GONZALEZ; WOODS, 2010):

$$W(r,s) = \frac{\hat{H}^*(r,s)}{|\hat{H}(r,s)|^2 + [S_{nn}(r,s)/S_{uu}(r,s)]} \ . \tag{2.35}$$

Embora o filtro de Wiener tenha a propriedade de remover eventuais singularidades e também como consequência seja mais estável durante as etapas do processo de restauração, a relação sinal-ruído $[S_{nn}/S_{uu}]$ não é conhecida. Na verdade, este é um parâmetro livre estimado empiricamente.

2.2 Transformada Wavelet

Como mencionado anteriormente, um filtro passa-baixa pode servir como uma solução inversa (HANSEN; JENSEN, 1998). Desta forma, projetos de filtro passa-baixa podem ser investigados como uma técnica para obtenção de soluções de problemas inversos. Aqui, trata-se de projeto de filtro passa-baixa baseado na transformada *Wavelet*.

As *wavelets* são funções usadas na representação de dados ou outras funções de acordo com diferentes escalas. A ideia de usar funções sobrepostas para representar outras funções não é nova. Ela vem desde o início do século XIX, quando Joseph Fourier descobriu que poderia sobrepor senos e cossenos de diferentes frequências para realizar esta tarefa. Na análise com base em funções *wavelets*, a grande contribuição é o fato de se considerar a escala utilizada para representar os dados iniciais como sendo um segundo parâmetro da família de funções a serem sobrepostas.

Como mencionado em Williams e Amaratunga (1994), a existência de funções semelhantes às *wavelets* é conhecida desde o início do século XX (alguns exemplos notáveis são as denominadas *wavelets* de Haar e *wavelets* Littlewood-Paley). No entanto, a formalização da teoria de funções *wavelets* ocorreu na década de 80, com os trabalhos seminais de J. Morlet e seus colaboradores A. Grossmann e P. Goupillaud, seguidos por contribuições de Y. Meyer, I. Daubechies e S. Mallat (MALLAT, 2008; DOMINGUES; KAIBARA, 2012).

Muitas das ideias incorporadas na construção de *wavelets* foram originadas em diferentes áreas, como por exemplo em estudos sobre codificação de sub-bandas em engenharia, estados coerentes e teoria de grupos de renormalização em física e o estudo de operadores de Calderon-Zygmund em matemática, ver (DOMINGUES; KAI-

BARA, 2012) e suas referências para uma revisão mais abrangente sobre *wavelets* e suas transformadas contínuas.

Nesta dissertação será considerada a formulação discreta para as *wavelets* ortogonais da família de Daubechies e suas transformadas rápidas. A seguir, são apresentados um resumo sobre as principais propriedades desta família de funções e alguns esquemas simplificados para os algoritmos rápidos a serem utilizados.

2.2.1 Família Daubechies

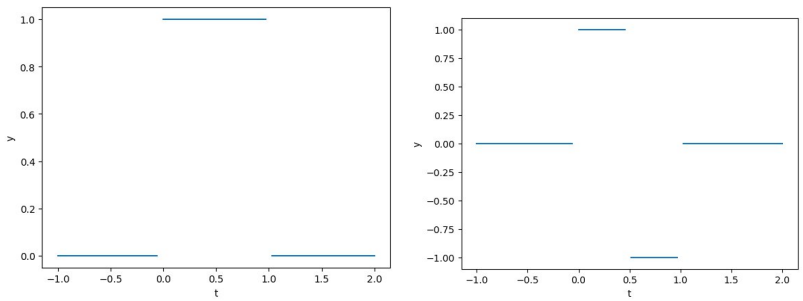
Em Daubechies (1992), é apresentada a construção da família de funções *wavelet* ortogonais de suporte compacto, que são caracterizadas por propriedades impostas às suas transformadas de Fourier. Esta família de funções *wavelet* é gerada por duas funções linearmente independentes, uma chamada *função escala* $\phi(x)$ e outra chamada *função wavelet* $\psi(x)$.

No caso das funções de Haar, as duas funções são definidas por partes e são não diferenciáveis:

$$\varphi(t) = \begin{cases} 1, & 0 \leq t < 1, \\ 0, & \text{caso contrário.} \end{cases} \qquad \psi(t) = \begin{cases} 1, & 0 \leq t < \frac{1}{2}, \\ -1, & \frac{1}{2} \leq t < 1, \\ 0, & \text{caso contrário.} \end{cases} \tag{2.36}$$

A Figura 2.3 mostra as respectivas função de escala e função *wavelet* da *wavelet* Haar.

Figura 2.3 - Funções de Haar.



Função Escala $\phi(x)$ (esquerda) e Wavelet $\psi(x)$ (direita)
Fonte: Produção do autor.

Uma destas propriedades, referente à suavidade da *wavelet* construída, é a possibilidade de polinômios de grau $< n$ serem representados exatamente como combinação linear apenas de funções escalas, sendo nulos todos os seus coeficientes *wavelet*, independentemente da escala considerada na decomposição. Esta é conhecida como sendo a propriedade dos momentos nulos (n) que a *wavelet* possui. Este valor (n) ainda define a largura dos filtros ($2n$) associados a estas funções, usualmente denotadas por "dbn" . Neste sentido, as funções de Haar, denotadas por "db1", são associadas a filtros de largura 2 e possuem 1 momento nulo. Os filtros $h[1] = [\frac{1}{2}, \frac{1}{2}]$ estão associados à função escala $\phi(x)$ de Haar e os filtros $g[1] = [\frac{1}{2}, -\frac{1}{2}]$, à função *wavelet* $\psi(x)$, correspondente.

Em Daubechies (1992), foi demonstrado que as funções $\phi(x)$ e $\psi(x)$, junto com suas translações e dilatações, denotadas por $\phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k)$ e $\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k)$, $k \in \mathrm{Z}$, formam uma base ortogonal para o espaço vetorial das funções de quadrado integrável, $\mathcal{L}_2 = \{f(x) \mid \int_{-\infty}^{\infty} f^2(t)dt < \infty\}$. Assim, qualquer função $f(x) \in \mathcal{L}_2$ pode ser definida por uma série *wavelet*.

Além disso, a partir da base *wavelet* obtém-se uma estrutura de *análise multiresolução* (MALLAT, 2008), que é uma sequência infinita de subespaços encaixados, $... \subset V_{j-1} \subset V_j \subset V_{j+1} \subset ...$, gerados pelas funções escalas no nível de resolução j, $V_j = [2^{j/2}\phi(2x - k)]$, de tal modo $\mathcal{L}_2 = \overline{\bigcup_{j=-\infty}^{\infty} V_j}$, e $\{0\} = \overline{\bigcap_{j=-\infty}^{\infty} V_j}$.

Por causa da ortogonalidade entre funções escala e *wavelets*, tem-se $V_{j+1} = V_j \oplus W_j$. Ou seja, os espaços $W_j = [2^{j/2}\psi(2x - k)]$, gerados pelas *wavelets*, representam as informações ortogonais que podem ser acrescentadas aos dados suaves do nível j. Daubechies demonstra, então, que o espaço $\mathcal{L}_2$ pode ser também gerado apenas pelas funções *wavelets*. No entanto, na maior parte das aplicações, é conveniente que a decomposição da informação ocorra até um certo nível mais grosseiro (V_0), $\mathcal{L}_2 = \overline{V_0 \bigcup_{j=0}^{\infty} W_j}$. Esta estrutura permite que uma função qualquer $f(x) \in \mathcal{L}_2$ possa ser representada por uma aproximação grosseira dependendo de ϕ no nível $j = 0$ e demais componentes ortogonais em diferentes níveis, ou escalas, geradas por ψ, suas translações e dilatações.

A equação 2.37 apresenta uma representação para $f(x)$ na base *wavelet* para $\mathcal{L}_2$ (DAUBECHIES, 1992). Na Equação 2.37, o parâmetro k controla as translações em relação a cada nível-j de resolução:

$$f(x) = \sum_{k=-\infty}^{+\infty} C_{0,k}\phi(x - k) + \sum_{j=0}^{+\infty}\sum_{k=-\infty}^{+\infty} D_{j,k}2^{-j/2}\psi(2x - k), \tag{2.37}$$

sendo os coeficientes $C_{0,k} = \langle f(x), \phi(x-k)\rangle$ e $D_{j,k} = \langle f(x), 2^{-j/2}\psi(2x-k)\rangle$, obtidos na prática[3] pelas transformadas rápidas, que são maneiras de se obter os produtos internos através dos filtros discretos $g[n]$ e $h[n]$ associados à base *wavelet*.

Os algoritmos a seguir ilustram a forma decimada não normalizada da transformada de Haar, direta dada pelo Algoritmo 1 e inversa dada pelo Algoritmo 2, (SILVEIRA; KOZAKEVICIUS, 2016).

Algorithm 1 Transformada Haar 1D

Require: C_J, J
  for $j \leftarrow J : 1$ **do**
    for $i \leftarrow 0 : 2^{j-1} - 1$ **do**
      $C_{j-1,i} \leftarrow \frac{1}{2}C_{j,2i} + \frac{1}{2}C_{j,2i+1}$
      $D_{j-1,i} \leftarrow \frac{1}{2}C_{j,2i} - \frac{1}{2}C_{j,2i+1}$
    end for
  end for

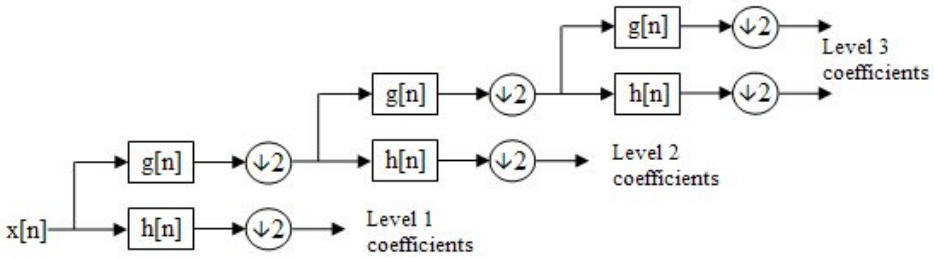
Algorithm 2 Transformada Haar 1D Inversa

Require: C_{j-1}, D_{j-1}
  for $j \leftarrow 0 : J - 1$ **do**
    for $i \leftarrow 0 : I - 1$ **do**
      $C_{j,2i} \leftarrow C_{j-1,i} + D_{j-1,i}$
      $C_{j,2i+1} \leftarrow C_{j-1,i} - D_{j-1,i}$
    end for
  end for

A Figura 2.4 ilustra o Algoritmo 1 para a transformada decimada, na qual são apresentados três níveis de decomposição de um sinal unidimensional $x[n]$. Nesta figura, as operações realizadas pela transformada direta são representadas como convoluções do sinal com os respectivos filtros passa-alta $g[n]$ e passa-baixa $h[n]$ associados à base da transformação. A operação de decimação, representada pelo círculo contendo o número 2 e a seta para baixo, indica que a cada nível de decomposição as componentes obtidas possuem metade do tamanho do sinal de entrada no nível anterior. Essa decomposição em cascata é chamada de (*filter bank*), proposta por Mallat (MALLAT, 2008), cuja ordem do esquema numérico é $NlogN$, N=número de elementos no nível inicial $j = 0$.

[3] O produto interno entre duas funções aqui é definido como: $\langle f(x), g(x)\rangle \equiv \int_{-\infty}^{+\infty} f(x)\, g(x)\mathrm{d}x$.

Figura 2.4 - *Filter bank.*



TWD em 3 níveis representada como convolução entre o sinal de entrada $x[n]$ e os filtros wavelet $g[n]$ e $h[n]$ em cada nível. Assim tem-se: $x[n] \leftrightarrow [C_2 D_2] \leftrightarrow [C_1 D_1 D_2] \leftrightarrow [C_0 D_0 D_1 D_2] = \boldsymbol{v} = TWD_3(x)$.

Fonte: Wikipedia (2020c).

Desta forma, obtém-se $\mathbf{v}$ a decomposição multiresolução do sinal de entrada $x[n]$ em j níveis, na qual todos os blocos de coeficientes wavelets nos j níveis e os coeficientes escala do último nível são agregados: $\mathbf{v} = TWD_j(x) = [C_0 D_0 D_1 ... D_{j-1}]$.

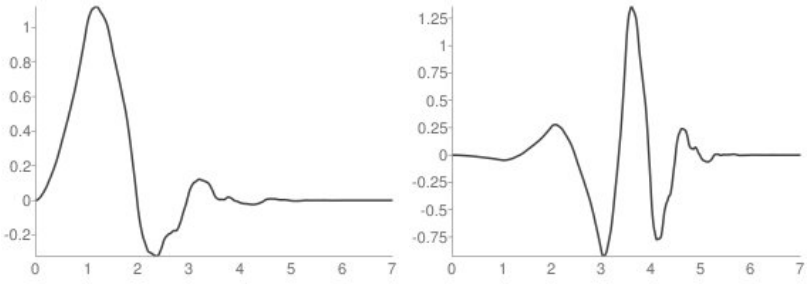
Além da *wavelet* de Haar e demais *wavelets* de Daubechies, existem outras famílias de funções *wavelets* (MALLAT, 2008), dentre elas são apresentadas, a seguir, as funções Symlet e Biortogonal, cujos resultados para o problema de restauração serão verificados numericamente na seção de testes e discussões.

2.2.2 Família Symlet

A família Symlet foi também proposta por Daubechies como uma modificação da família dbn de funções ortogonais de suporte compacto para a obtenção de maior simetria. As Symlets costumam apresentar melhores resultados em problemas de remoção de ruído, segundo (SRIDHAR et al., 2014).

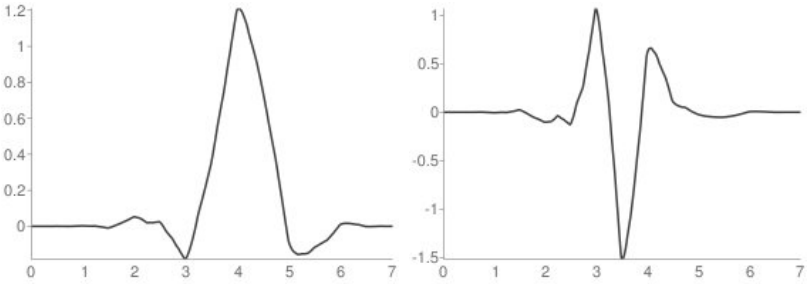
Apesar do nome sugerir que a família seja simétrica, ela é, na verdade, quase simétrica. As funções Symlet, denotadas por symn, são ortogonais e ainda mantém a propriedade de possuírem n momentos nulos. A simetria completa só é possível abandonando a propriedade de ortogonalidade (ABDELNOUR; SELESNICK, 2004). A exceção é a função de Haar (db1), que é tanto ortogonal quanto simétrica, no entanto, não é suave. As famílias de funções de Daubechies e Symlet são idênticas para $n = 1, 2, 3$. A partir de quatro momentos nulos, $n = 4$, estas funções diferem. As Figuras 2.5 e 2.6 mostram as funções escala e wavelets com $n = 4$ momentos

Figura 2.5 - Funções db4.



Função Escala $\phi(x)$ (esquerda) e Wavelet $\psi(x)$ (direita)
Fonte: PyWavelets (2020).

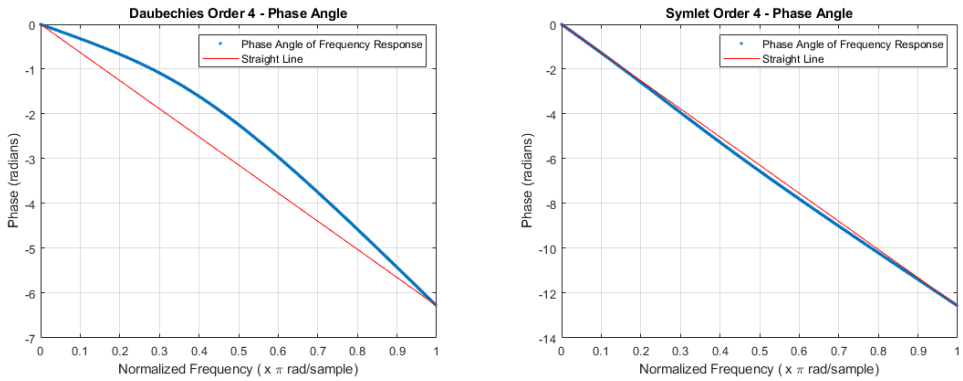
Figura 2.6 - Funções sym4.



Função Escala $\phi(x)$ (esquerda) e Wavelet $\psi(x)$ (direita).
Fonte: PyWavelets (2020).

nulos para as duas famílias *wavelets*, db4 e sym4. Na Tabela 2.1 são apresentados os $2n$ filtros da db4 e da sym4 a serem usados nas transformadas direta e inversa das respectivas transformações. As wavelets simétricas ou quase simétricas tornam-se interessantes para o problema de restauração de imagens por terem a linearidade da fase do filtro proporcional à simetria. Quanto mais simétrica a wavelet, mais linear é o filtro e, por consequência, menor a distorção de fase (STANHILL; ZEEVI, 1998). A Figura 2.7 mostra o ângulo de fase dos filtros sym4 e db4. Nota-se como a fase da sym4 é aproximadamente, não perfeitamente simétrica enquanto a fase da db4 é uma curva não linear.

Figura 2.7 - Ângulos de fase db4 e sym4.



Fase db4 (esquerda) e fase sym4 (direita).
Fonte: MathWorks (2020).

Tabela 2.1 - Filtros db4 e sym4.

n	db4 g[n]	db4 h[n]	sym4 g[n]	sym4 h[n]
0	-0.011	0.230	-0.076	0.032
1	0.033	-0.715	-0.030	0.013
2	0.031	0.631	0.498	-0.100
3	-0.187	0.028	0.804	-0.298
4	-0.028	-0.187	0.298	0.804
5	0.631	-0.031	-0.100	-0.498
6	0.715	0.033	-0.013	-0.030
7	0.230	0.011	0.032	0.076

2.2.3 Família Biortogonal

A família Daubechies é formada por *wavelets* ortogonais mas não simétricas, a família Symlet possui *wavelets* ortogonais e aproximadamente simétricas. Como mencionado na seção anterior, segundo (DAUBECHIES, 1992) não é possível obter simetria completa e ortogonalidade entre as funções $\phi(x)$ e $\psi(x)$ ao mesmo tempo. No entanto, uma alternativa para se obter uma família de funções que preservem simetria e ortogonalidade é através da construção das wavelets *Biortogonais*. Para isso, além dos espaços $V_j = [2^{j/2}\phi(2x-k)]$ e $W_j = [2^{j/2}\psi(2x-k)]$, como feito em 2.2.1, são introduzidos os espaços duais V_j^* e W_j^*. A ortogonalidade, então, é obtida entre os espaços e seus complementos ortogonais duais. Para uma discussão detalhada sobre as análises multiresolução envolvendo os espaços duais, ver (DOMINGUES; KAIBARA,

2012). Nesta seção, apresenta-se uma descrição sucinta das principais propriedades das wavelets *Biortogonais* com base nas referências (CHENG; ZHANG, 2014; DOMINGUES; KAIBARA, 2012).

A família *Biortogonal* é caraterizada por uma base composta de duas funções escala e duas funções *wavelet*, referidas como funções primais $\{\phi, \psi\}$ e funções duais $\{\phi^*, \psi^*\}$, além de suas translações e dilatações $\phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k)$, $\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k)$, $\phi^*_{j,k}(x) = 2^{j/2}\phi^*(2^j x - k)$ e $\psi^*_{j,k}(x) = 2^{j/2}\psi^*(2^j x - k)$, $j, k \in \mathrm{Z}$. Para cada nível j, tem-se que as funções escala e as funções wavelets duais são ortogonais ente si, Equação 2.38. O mesmo ocorre entre as funções escala duais e as wavelets de um mesmo nível, independentemente do parâmetro de translação, Equação 2.39:

$$\langle \phi_{j,k}(x), \psi^*_{j,l}(x) \rangle = 0, \ \forall j, k, l \in \mathrm{Z}, \tag{2.38}$$

$$\langle \phi^*_{j,k}(x), \psi_{j,l}(x) \rangle = 0, \ \forall j, k, l \in \mathrm{Z}. \tag{2.39}$$

Além disso, as wavelets e as wavelets duais são ortogonais entre si independentemente do nível de resolução j e do parâmetro de translação $k \in \mathrm{Z}$. Considerando-se δ a função de Dirac, tem-se

$$\langle \psi^*_{j,k}(x), \psi_{m,l}(x) \rangle = \delta_{j,m}\delta_{k,l}, \ \forall j, k, l, m \in \mathrm{Z}. \tag{2.40}$$

Da mesma forma como as wavelets ortogonais, as biortogonais também são definidas por relações de escala envolvendo os filtros h, g^* e h^*, g, equações 2.41 e 2.42,

$$\phi(t) = \sqrt{2}\sum_k h(k)\phi(2t-k), \quad \psi(t) = \sqrt{2}\sum_k g(k)\phi(2t-k) \tag{2.41}$$

$$\phi^*(t) = \sqrt{2}\sum_k h^*(k)\phi(2t-k), \quad \psi^*(t) = \sqrt{2}\sum_k g^*(k)\phi(2t-k), \tag{2.42}$$

sendo que os filtros, quando satisfazem $h(n) = (-1)^{1-n}g^*(1-n)$ e $h^*(n) = (-1)^{1-n}g(1-n)$, também preservam as relações de ortogonalidade dadas pelas expressões 2.38, 2.39 e 2.40. (CHENG; ZHANG, 2014). Novamente a associação entre cada uma das funções (primais e duais) e seus filtros correspondentes possibilita a obtenção de algoritmos rápidos para as transformadas nesta base (MALLAT, 2008). As equações abaixo expressam essa correspondência entre as funções

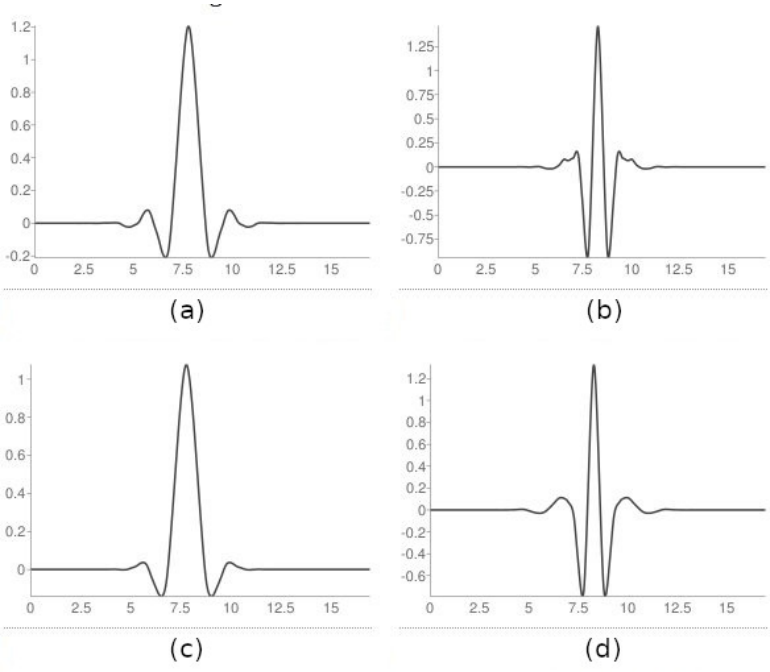
$\phi(t), \phi^*(t), \psi(t), \psi^*(t)$ e os filtros da *wavelet*:

$$\{\phi(t), \phi^*(t)\} \equiv \{h(n), h^*(n)\}, \tag{2.43}$$

$$\{\psi(t), \psi^*(t)\} \equiv \{g(n), g^*(n)\}. \tag{2.44}$$

Os filtros $h^*(n)$ e $g^*(n)$ são filtros, respectivamente, passa-baixa e passa-alta de de-

Figura 2.8 - Decomposição *Wavelet* Biortogonal 6.8. Funções (a) escala e (b) *wavelet* da base primal. Funções (c) escala e (d) *wavelet* da base dual.



Fonte: PyWavelets (2020).

composição (ou seja, aqueles utilizados na transformada direta), cujos comprimentos podem assumir valores distintos N_{h^*} e N_{g^*}. Analogamente, $h(n)$ e $g(n)$ são filtros passa-baixa e passa-alta de reconstrução (considerados na transformada inversa) e seus comprimentos são N_g e N_h, respectivamente. Aqui, os compromentos dos filtros são todos dados por valores pares. Observa-se ainda que existe a possibilidade de se considerar filtros com diferentes comprimentos para a transformada inversa. Isso possibilita que se escolha funções com regularidades distintas para a decomposição e reconstrução dos sinais (imagens) (CHENG; ZHANG, 2014).

Assim a Equação 2.45 apresenta uma representação para $f(x)$ na base *wavelet biortogonal* para $\mathcal{L}_2$ (DOMINGUES; KAIBARA, 2012).

$$f(x) = \sum_{k=-\infty}^{+\infty} C_{0,k}\phi(x-k) + \sum_{j=0}^{+\infty}\sum_{k=-\infty}^{+\infty} D_{j,k}2^{-j/2}\psi(2x-k), \tag{2.45}$$

no entanto os coeficientes $C_{0,k} = \langle f(x), \phi^*(x-k)\rangle$ e $D_{j,k} = \langle f(x), 2^{-j/2}\psi^*(2x-k)\rangle$, são dados em relação às funções duais. Na prática, os somatórios em 2.45 são finitos e os coeficientes são obtidos via transformadas rápidas. Os filtros duais h^* e g^* são utilizados na transformada rápida biortogonal direta e os filtros primais h e g são considerados para a transformada inversa (DOMINGUES; KAIBARA, 2012).

Como exemplo, a Figura 2.8 mostra as funções escala e *wavelet* das funções de decomposição e reconstrução da wavelet Biortogonal 6.8, que será utilizada neste trabalho. Esta notação indica que as wavelets primais possuem 6 momentos nulos e as duais, 8.

A família biortogonal é amplamente utilizada em problemas de compressão de imagens (ZHANG et al., 2012) e é também utilizada em restauração (SHEN; SUN, 2004).

2.2.4 A Transformada Wavelet 2D

A transformada *wavelet* considerando diferentes famílias de funções, apresentada até então, é aplicada a sinais unidimensionais. Porém, para dados bidimensionais (imagens) é necessária a aplicação de uma transformada de dimensão correspondente. Para sua obtenção, é necessária, então, a construção de uma família de funções ortogonais bidimensionais. Daubechies em (DAUBECHIES, 1992) propôs a seguinte construção a partir das funções definidas na subseção anterior:

$$\begin{aligned} \Phi(u,v) &= \phi(u)\phi(v), \quad & \Psi^H(u,v) &= \psi(u)\phi(v), \\ \Psi^V(u,v) &= \phi(u)\psi(v), \quad & \Psi^D(u,v) &= \psi(u)\psi(v), \end{aligned} \tag{2.46}$$

sendo $\Phi(x,y)$ a função escala bidimensional. As funções Ψ^H, Ψ^V e Ψ^D são as funções *wavelet*, cada uma representando variações de acordo com uma direção (H=horizontal, V=vertical, D=diagonal). A base bidimensional é então obtida a partir das novas funções, além de suas dilatações e translações, cuja notação segue o mesmo tipo de regra fixada anteriormente para as funções unidimensionais, como

mostra a Equação 2.47:

$$\phi_{j,k,l}(u,v) = 2^{j/2}\phi(2^j u - k, 2^j v - l) \ , \tag{2.47}$$

$$\psi^i_{j,k,l}(u,v) = 2^{j/2}\psi(2^j u - k, 2^j v - l) \ , \quad \text{com} \ \ i \in \{H, V, D\} \ . \tag{2.48}$$

O algoritmo padrão (*forma standard*) da transformada *wavelet* discreta bidimensional é obtido pela aplicação da transformada *wavelet* 1D em todas as linhas e depois em todas as colunas da imagem. O Algoritmo 3 ilustra 1 nível da transformada. A ordem de execução do algoritmo, se primeiro linhas ou colunas, influencia no resultado. Também ocorre alteração nos coeficientes *wavelet* se as transformadas por linha forem aplicadas em vários níveis antes de serem intercaladas com as transformadas aplicadas nas colunas da imagem. Quando o algoritmo padrão for considerado, caso sejam realizados mais de um nível de decomposição, apenas o bloco de coeficientes correspondentes à função escala será decomposto, o que corresponde a um quarto dos dados do nível anterior.

Algorithm 3 Transformada Haar 2D

Require: $I_{M\times N}$,
  for $l \leftarrow 0 : M-1$ **do**
    $I_{l;0,1,\ldots,N-1} \leftarrow TWD_{log(N)}(I_{l;0,1,\ldots,N-1})$
  end for
  for $c \leftarrow 0 : N-1$ **do**
    $I_{0,1,\ldots,M-1;c} \leftarrow TWD_{log(M)}(I_{0,1,\ldots,M-1;c})$
  end for

No caso da escolha da base de Haar, nenhum tratamento especial nas fronteiras da imagem será necessário. Caso sejam escolhidas *wavelets* com mais de dois filtros, então serão necessárias alternativas de extensão dos dados em posições "fantasmas" que estão fora do domínio de variação dos dados originais (KOZAKEVICIUS; SCHMIDT, 2013). O tratamento mais comum é a periodização e pode ser visto em mais detalhes em (MALLAT, 2008; DAUBECHIES, 1992). A transformada inversa pode ser obtida seguindo a ordem reversa do Algoritmo 3, e é exemplificada para um nível no Algoritmo 4.

A Figura 2.9 apresenta as decomposições *wavelet* bidimensionais de uma mesma imagem, disponível no pacote PyWavelets (LEE et al., 2019), considerando-se db1, db4 e sym4. Em cada linha da sequência de imagens estão os coeficientes escala

Algorithm 4 Transformada Haar Inversa 2D

Require: $(K = LL, LH, HL, HH)_{\frac{M}{4} \times \frac{N}{4}}$

  for $c \leftarrow 0 : N - 1$ **do**

    $I_{0,1,...,M-1;c} \leftarrow ITWD_{log(M)}(K_{0,1,...,M-1;c})$

  end for

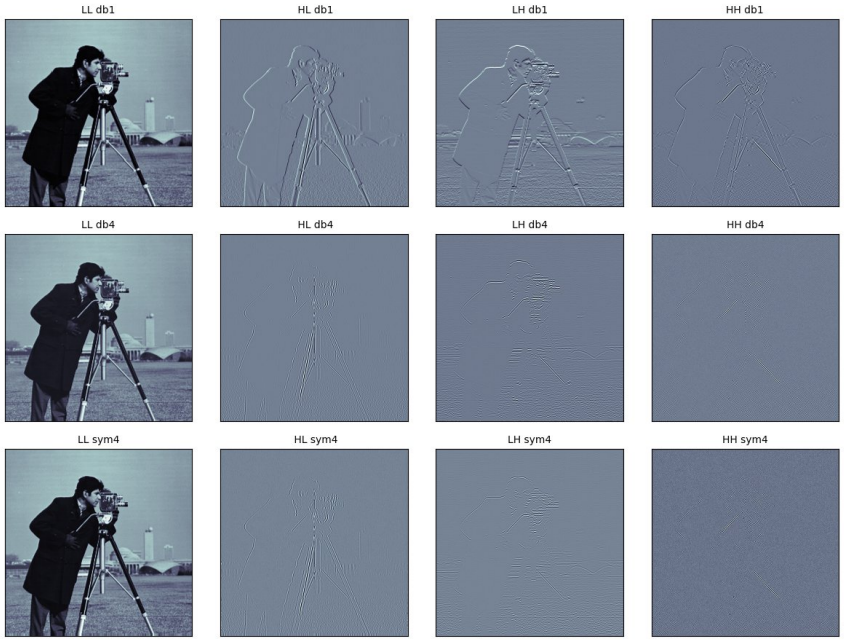
  for $l \leftarrow 0 : M - 1$ **do**

    $I_{l;0,1,...,N-1} \leftarrow ITWD_{log(N)}(K_{l;0,1,...,N-1})$

  end for

(LLdb1) e as três componentes de coeficientes *wavelet* H, V, e D (HLdb1, LHdb1, HHdb1) para um nível de decomposição via Transformada wavelet bidimensional, via algoritmo padrão.

Figura 2.9 - Decomposição *Wavelet* 2D Haar, db4 e sym4. Da esquerda para a direita: Coeficientes de escala, coeficientes *wavelet* horizontal, vertical, diagonal.



Fonte: Produção do autor.

Nota-se na Figura 2.9 que os coeficientes diagonal são obtidos pela convolução da imagem com filtros passa-alta. Assim, esses coeficientes podem atuar como um detector de bordas. É na componente diagonal da decomposição que se espera a maior

propagação de ruído, dado que o ruído [gaussiano] introduz variações na imagem e essa variação é vista como componente de alta frequência (MALLAT, 2008). O coeficiente de escala preserva as informações de baixas frequências da imagem original, portanto espera-se a menor propagação do ruído nesta componente.

2.2.5 Truncamento

A transformada *wavelet* torna-se interessante para o problema de restauração de degradação por ruído devido à possibilidade de se retirar as altas frequências dos dados, visto que o ruído propaga-se nas frequências mais altas (RAJNI; ANUTAM, 2013). Devido a essa propagação, a restauração está ligada à análise dos coeficientes *wavelet* nas três direções (H, V, D) (CHANG et al., 2000). Os coeficientes do bloco diagonal (D) captam o ruído em todas as direções, enquanto os coeficientes horizontal (H) e vertical (V), apenas na respectiva direção. Como parte do sinal também é propagada nas altas frequências, a eliminação do ruído requer um processamento que diferencie sinal e ruído. Este está associado ao truncamento dos coeficientes wavelets, que é o corte dos coeficientes wavelet cujos módulos estejam acima de um certo valor λ, chamado de *threshold* (limiar de corte) (NIU; SHEN, 2007).

Existem duas formas principais de truncamento: *soft* e *hard thresholding*. Em ambas é necessária a determinação do valor de corte λ segundo algum critério, discutido na Seção 2.2.5.1. Neste trabalho será utilizada a forma *soft thresholding*, vista na Equação 2.49, como parte do método de restauração de imagens. No truncamento *soft*, visto na Equação 2.49, os coeficientes wavelets com módulo menor do que λ são descartados e os demais possuem seus módulos subtraídos de λ. Essa diminuição no módulo de todos os coeficientes wavelets da série (*shrinkage*: encolhimento) produz uma suavização nos dados reconstruídos. Na forma *hard*, no entanto, os coeficientes com módulo acima de λ ficam inalterados.

$$thr_{\lambda}^{S}(D_{j,k}) = \begin{cases} 0, & se\ |D_{j,k}| \leq \lambda \\ sign(D_{j,k})(|D_{j,k} - \lambda|), & se\ |D_{i,j}| > \lambda \end{cases} \tag{2.49}$$

O critério de obtenção do limiar de truncamento λ e, consequentemente, o algoritmo que o calcula a cada nível da transformada é a parte mais relevante do método de truncamento/filtragem do sinal. Uma propriedade em comum entre vários métodos de truncamento é a hipótese de que o desvio padrão do ruído, σ, seja inicialmente conhecido, dado como um valor de entrada. Esta informação geralmente é desconhecida e sua estimativa pode ser difícil de ser obtida, mesmo que apenas visualmente.

Donoho (DONOHO; JOHNSTONE, 1994) propôs um método para estimar o desvio padrão através do desvio médio absoluto (Equação 2.50), e este pode ser obtido a partir dos coeficientes do bloco diagonal de primeiro nível de decomposição da transformada wavelet 2D:

$$\sigma \approx \frac{Median(|D_0|)}{0.675}. \tag{2.50}$$

2.2.5.1 Determinação do valor de corte

Entre os métodos de estimação do valor de truncamento λ propostos na literatura, destacam-se dois: *VisuShrink* (DONOHO; JOHNSTONE, 1994) e BayesShrink (CHANG et al., 2000). O *VisuShrink* utiliza o chamado *threshold universal* definido por Donoho, como mostra a equação abaixo:

$$\lambda_{vs} = \sigma\sqrt{2\log\left(M \times N\right)}. \tag{2.51}$$

A variável σ representa o desvio padrão do ruído. $M \times N$ é a dimensão do dado de entrada (no caso de imagens). Aqui nota-se ainda que a restauração *wavelet* assume que o ruído possui uma distribuição de probabilidade gaussiana.

Em geral, o método *VisuShrink* tende a gerar um valor de λ alto e com isso o processo de truncamento acaba eliminando uma parte considerável do sinal. Isto causa uma suavização excessiva do sinal (ou da imagem) analisado(a). O *threshold universal* também tem a desvantagem de ser não-adaptativo, uma vez que é assumido como sendo constante ao longo de todos os níveis e todos os blocos de coeficientes wavelets no caso das imagens.

Um método mais robusto de determinação do valor de *threshold* foi apresentado como *BayesShrink*. Este é um método adaptativo, ou seja, cada decomposição possui seu próprio coeficiente de threshold estimado a partir dos dados analisados. Seu cálculo é realizado pela Equação 2.52 (CHANG et al., 2000). A variância σ^2 é calculada pelos coeficientes do bloco diagonal de nível 1 pela Equação 2.50, enquanto o desvio σ_y é estimado localmente para cada bloco da decomposição *wavelet*: $y = HL, LH, HH$, o que permite a determinação de um valor de corte λ_y para cada bloco.

$$\lambda_y = \frac{\sigma^2}{\sigma_X}, \quad \sigma_X = \sqrt{max(\sigma_y^2 - \sigma^2, 0)}, \quad \sigma_y^2 = \frac{1}{n^2}\sum_{i,j=1}^{n} y_{ij}^2. \tag{2.52}$$

O processo geral (unidimensional) para gerar o sinal restaurado $\hat{x}$ a partir do sinal x possui 3 etapas, descritas pelo Algoritmo 5.

Algorithm 5 Filtragem *wavelet*

Require: $x[n], J$

$\mathbf{v} \leftarrow TWD_J(x) \leftarrow [C_0 D_0 D_1 ... D_{J-1}]$ ▷ $\mathbf{v}$ é a decomposição do sinal original x em J níveis por meio do Algoritmo 3

$\mathbf{v}^{thr} = thr_\lambda(\mathbf{v}) = [C_0 D_0^{thr} D_1^{thr} ... D_{J-1}^{thr}]$ ▷ Usa o método *soft*, sendo o valor de λ dado por 2.51 ou 2.52.

$\hat{x}$=$TWDI_J(\mathbf{v}^{thr})$ ▷ $\hat{x}[n]$ é o sinal reconstruído obtido por meio do Algoritmo 4

2.2.5.2 Transformação do canal de cor

As imagens coloridas costumam ser representadas como uma composição de canais RGB (*Red-Green-Blue*, Vermelho-Verde-Azul). Estes canais representam a intensidade de brilho de cada cor. Essa representação é a mais popular, mas não é adequada para muitos casos por apresentar redundância. Um exemplo é o caso de transmissão, onde a imagem precisa ser compactada para minimizar o custo de dados.

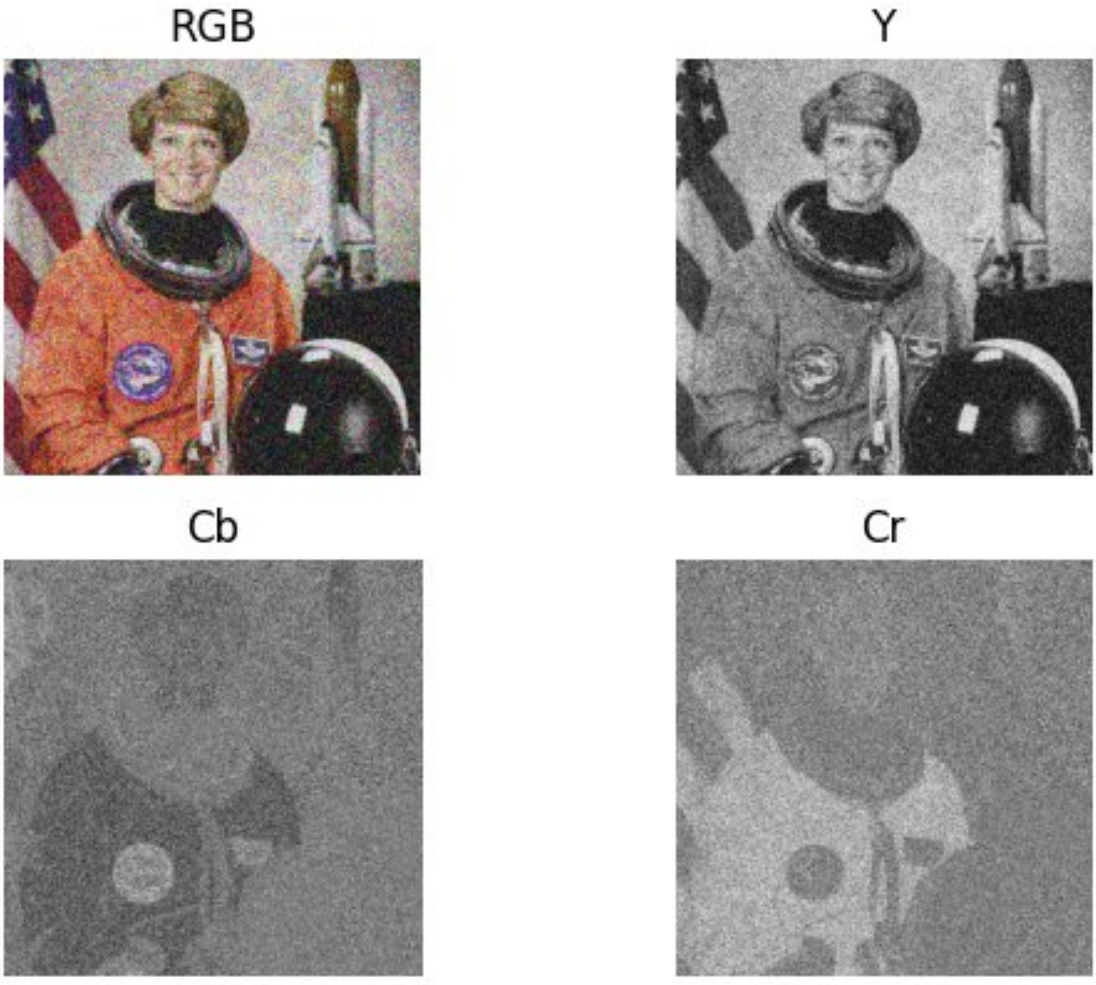
Uma representação mais eficiente da imagem é a YCbCr, que a representa em um canal Y chamado de *luma*, ou luminância, mais dois canais Cb e Cr chamados de *croma*, ou crominância. O canal Y contém a imagem em tons de cinza enquanto os canais Cb e Cr contém as informações de cores. Estes canais podem ser compactados e comprimidos, enquanto o canal Y contém a informação relevante da imagem.

Esta representação é utilizada com a transformada *wavelet* por preservar melhor as altas frequências. Por esta atuar como um filtro passa baixa, as bandas Cb e Cr, que possuem informação de baixa frequência, são melhor restauradas do que usando canais RGB. Sendo assim, esta representação remove menos informação de sinal do que a representação RGB, resultando em uma restauração mais fiel à imagem original (LIAN et al., 2005). A Figura 2.10 ilustra uma imagem representada nesta base.

2.2.5.3 Cycle spin

A base das TW ortogonais possuem a propriedade de não invariância na translação. A esta propriedade é atribuída a ocorrência do Fenômeno de Gibbs nas bordas (COIFMAN; DONOHO, 1995). Outras transformada como a SWT (*Stationary Wavelet Transform*) (FOWLER, 2005) e a DTCWT (*Dual-Tree Complex Wavelet Transform*)

Figura 2.10 - Representação YCbCr.



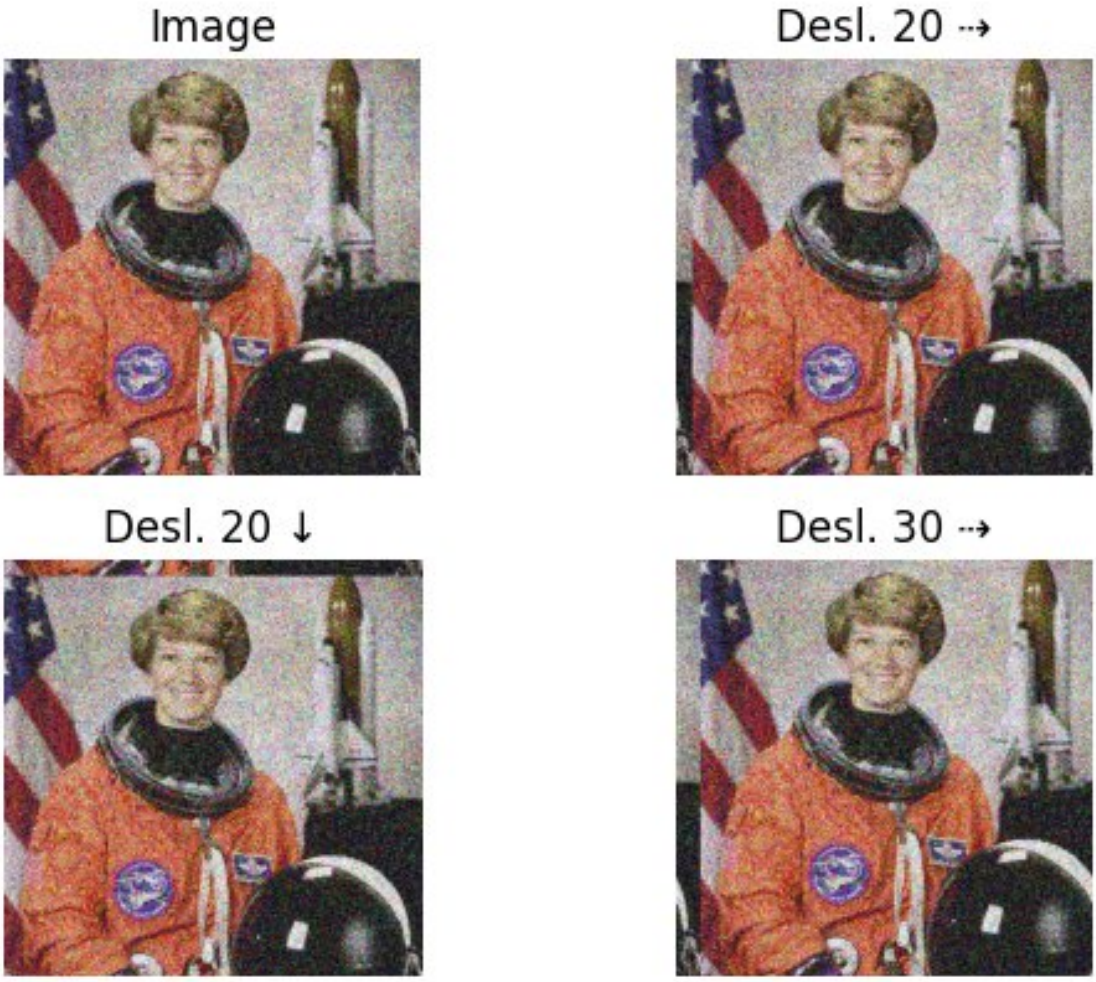
Fonte: Produção do autor.

(BARRI et al., 2012) são modificações da DWT e são invariantes e aproximadamente invariantes na translação, respectivamente. Uma técnica para suprimir os artefatos gerados pelo Fenômeno de Gibbs é realizar o processo de filtragem várias vezes após deslocar as linhas e colunas da imagem em uma certa quantidade de posições e calcular a média entre todas as restaurações. Este processo é chamado de *cycle spin* (COIFMAN; DONOHO, 1995).

Essa técnica consiste em considerar a imagem como uma matriz circulante, como visto na Figura 2.11.

Sendo assim, o algoritmo do *cycle spin* é composto dos seguintes passos, supondo

Figura 2.11 - *Cycle spin*.



Fonte: Produção do autor.

$s = 5$ deslocamentos:

a) Deslocar a imagem iniciando em $p = 0$ posições, nas duas dimensões.

b) Restaurar essa imagem deslocada.

c) Deslocar a imagem restaurada em $p = -p$ posições. Ou seja, desfazer o deslocamento.

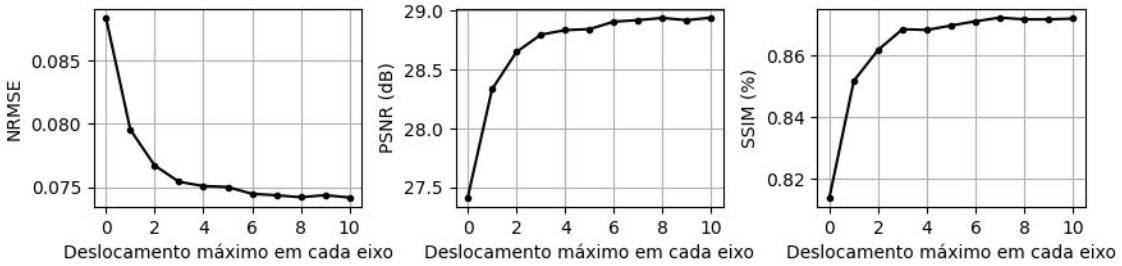
d) Guardar este resultado.

e) Repetir o processo com $p = p + 1$ até $p = s$.

Após a conclusão dos passos acima, é realizada uma média entre todos os resultados (neste caso, 6 no total), e a média entre todos estes resultados é o resultado final. Na prática, o valor de p pode ser diferente para cada dimensão. Sendo $s_0 = \{0, ..., s\}$ (linhas) e $s_1 = \{0, ..., s\}$ (colunas), os deslocamentos são formados por cada combinação diferente dos elementos entre cada conjunto: $s_{01} = \{\{0, 0\}, \{0, 1\}, \{1, 0\}, ...\{s, s\}$.

A Figura 2.12 mostra a evolução de três métricas diferentes (erro, razão sinal ruído, porcentagem de similaridade) para uma imagem qualquer. Nota-se que a partir de 7 deslocamentos a curva é estável e não há ganho com mais deslocamentos. Dentre as técnicas mencionadas, o *cycle spin* é a que mais contribui para o aumento mais expressivo nos valores obtidos pelas métricas de avaliação (MONEGO et al., 2020).

Figura 2.12 - Evolução das métricas de deslocamento para uma imagem qualquer.



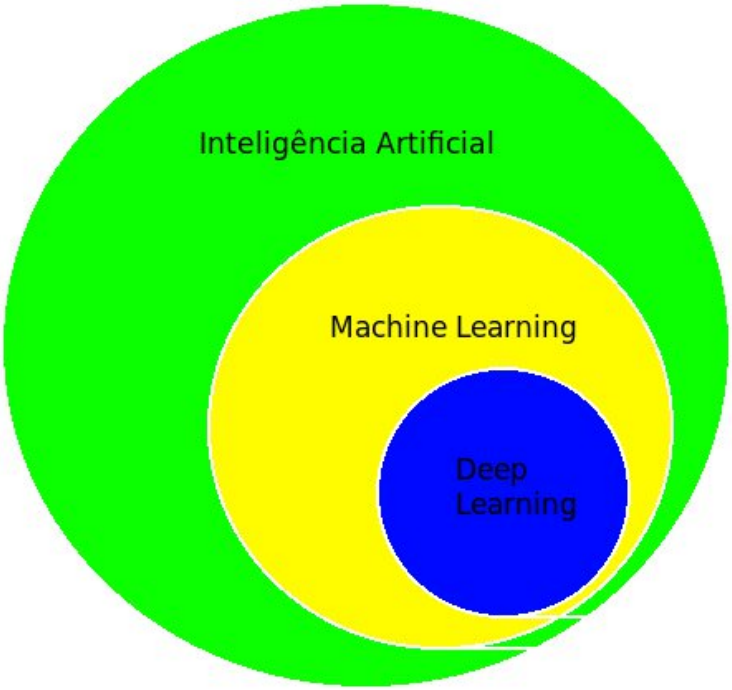
Fonte: Produção do autor.

2.3 Redes neurais

A Inteligência Artificial (AI: *Artificial Intelligence*) é uma área da ciência da computação dedicada a desenvolver algoritmos que façam com que máquinas possam emular um comportamento de inteligência. A AI tipicamente se divide em várias subáreas, três das quais são Computação Evolucionária (EC: *Evolutionary Computing*), Computação Cognitiva (CC: *Cognitive Computing*) e programação de linguagem natural (NLP: *Natural Language Programming*).

Uma outra subárea da AI é a Aprendizagem de Máquina (ML: *machine learning*). Esta trata de *ensinar* uma máquina a resolver um problema sem programação explícita para tal. Dentro da ML existe uma subárea muito importante chamada de *deep learning* (DL) (aprendizagem profunda) – ver Figura 2.13.

Figura 2.13 - Ilustração da relação entre inteligência artificial, *machine learning* e *deep learning*.



Fonte: Produção do autor.

As redes neurais artificiais (ANN: *artificial neural network*) (HAYKIN, 1999), tentam modelar o comportamento de aprendizado de um cérebro. Elas foram propostas inicialmente em 1943 (MCCULLOCH; PITTS, 1943), mas tiveram seu interesse renovado apenas em 1986 principalmente devido ao algoritmo *Backpropagation* (RUMELHART et al., 1986), utilizado até os dias de hoje. Outros três fatores também contribuiram com a retomada do interesse nesta técnica:

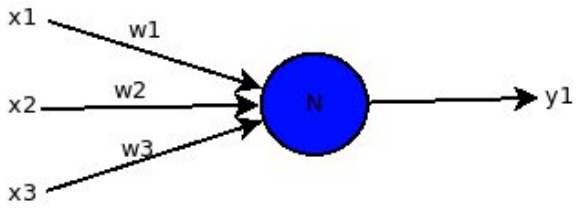
a) *Volume de dados*. A ciência trata com problemas cada vez maiores em volume de dados e estes problemas são classificados como *big data*.

b) *Recursos computacionais*. Computação paralela com processadores multi-núcleo e computação híbrida são desenvolvimentos recentes que melhoram o aproveitamento dos algoritmos de redes neurais.

c) *Facilidade*. Como consequência do interesse em redes neurais, foram criadas várias bibliotecas para facilitar o seu uso.

A rede neural percéptron mais simples possível é composta de um único neurônio, como visto na Figura 2.14. O exemplo da figura possui três entradas, uma única saída, sendo que cada conexão de entrada possui um peso associado. O objetivo de treinar a rede é encontrar a combinação de pesos que melhor resolvam o problema a partir de pesos iniciados aleatoriamente.

Figura 2.14 - Rede percéptron de um neurônio.



Fonte: Produção do autor.

A saída para este e para qualquer neurônio é a soma ponderada por pesos de conexões das entradas. A Equação 2.53 é a forma geral para calcular a saída de qualquer neurônio (HAYKIN, 1999):

$$z(w,x) = \sum_{j=1}^{N} w_j x_j + b \tag{2.53}$$

em que z é a saída da rede, x são as entradas, w são os pesos de cada conexão e b é o viés.

A Equação 2.53 é linear. De forma a melhorar o aprendizado, é necessário introduzir não-linearidade na saída de cada neurônio através de uma *função de ativação* f (HAYKIN, 1999, p. 10). Adaptando a Equação 2.53, obtém-se a Equação 2.54.

$$y(w,x) = f\left[\sum_{j=1}^{N} w_j x_j + b\right], \tag{2.54}$$

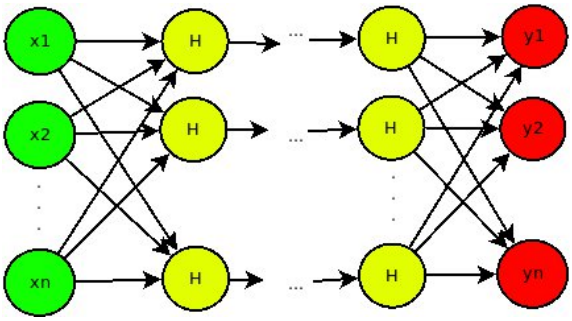
Exemplos de função de ativação são o retificador linear (TÓTH, 2013) (ReLu: *Rectified Linear Unit*), visto na Equação 2.55, e a função sigmoide, vista na Equação 2.56

$$f[z] = max(0, z) \tag{2.55}$$

$$f[z] = \frac{1}{1 + e^{-z}} \tag{2.56}$$

. Como a rede de um único neurônio é muito simples para se resolver um problema real, as redes são compostas de vários neurônios artificiais, distribuídos em camadas e esta arquitetura de rede é chamada de *rede percéptron de múltiplas camadas* (MLP: *Multilayer Perceptron*) (HAYKIN, 1999, p. 21). A Figura 2.15 ilustra uma rede genérica de n neurônios e m camadas. As camadas se dividem em *camada de entrada* (em verde), *camadas escondidas* (em amarelo) e *camada de saída* (em vermelho). A camada de saída pode conter uma ou mais saídas. Apesar de não constar na figura, cada conexão possui seu próprio peso.

Figura 2.15 - Rede percéptron de múltiplas camadas: em verde: camada de entrada, em amarelo: camadas escondidas, em vermelho: camada de saída.



Fonte: Produção do autor.

As ANN utilizam aprendizagem supervisionada: a rede recebe um conjunto de dados fonte e um conjunto correspondente de dados alvo, chamado de *conjunto de treinamento* (HAYKIN, 1999, p. 24). Considera-se que a rede aprendeu a resolver um problema se, ao introduzir um dado fonte de fora do conjunto de treinamento, a rede identifica a solução com a qualidade desejada.

Ainda falta uma informação à rede: como medir o seu progresso de aprendizado? Esta medição é feita pela função custo (*loss function*). A função pode ser, por exemplo, o erro quadrático, visto na Equação 2.57, em que z é o resultado obtido pela rede e t é o resultado alvo:

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} [z(w, x) - t]^2 \ . \tag{2.57}$$

A rede ajusta os pesos minimizando a função acima (HAYKIN, 1999, p. 51). No algoritmo *backpropagation*, a função custo é minimizada computando o seu gradiente e realizando a operação de descida do gradiente, por um método de otimização. Um exemplo de função de otimização é a função Adam (KINGMA; BA, 2014).

A rede necessita de um conjunto de dados para o processo de aprendizado. A identificação da matriz de pesos de conexão pode ser calculada por um processo iterativo em que as iterações são chamas de *épocas*. Cada época é uma *etapa* em que a rede inicia o processo de cálculo do erro das entradas a partir de pesos ajustados pelo algoritmo *backpropagation* na época anterior.

Assim, uma rede neural é composta dos seguintes elementos

- Entradas (x_i) Imagem degradada.
- Pesos (w_i): são ajustados durante a fase de treinamento.
- Viés (b_j): também são ajustados durante a fase de treinamento.
- Função de ativação (f): introduz não linearidade na soma ponderada descrita na Equação 2.53.
- Função custo ($E(w)$): é a discrepância para ajuste dos pesos. Um exemplo é o erro médio quadrático da Equação 2.57.
- Função de otimização: calcula os novos pesos para a iniciar a próxima época.

As variáveis de configuração da rede, inclusive o processo de aprendizado – como (tipo de otimizador, número de camadas, número de neurônios, tipo de função de ativação, parâmetros para otimizador e para a função de ativação), são chamadas de *hiperparâmetros*.

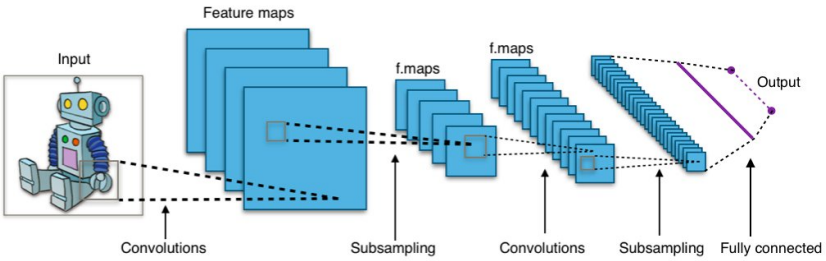
Outro hiperparâmetro muito importante utilizado em redes neurais é a *taxa de aprendizagem*, que define o passo com que o gradiente se move em direção ao mínimo (GOODFELLOW et al., 2016, p. 84). Uma taxa de aprendizagem muito alta tem como consequência uma maior probabilidade da rede estagnar em um mínimo local, enquanto uma taxa de aprendizagem muito baixa faz com que a rede precisa de muito mais épocas para chegar ao resultado esperado. Algoritmos mais avançados utilizam taxas de aprendizado variáveis durante o treinamento (GOODFELLOW et al., 2016, p. 271).

Estas são variáveis que o engenheiro escolhe e nem sempre essa escolha resulta numa combinação ótima. Quando os hiperparâmetros são otimizados por meio de outro problema de otimização, diz-se que a rede é *auto-configurada*.

2.3.1 Redes convolucionais

A rede percéptron de múltiplas camadas pode ser utilizada para tratamento de imagens, mas esta não preserva informação espacial da imagem – a MLP aceita apenas um vetor como entrada (ver Figura 2.15), portanto a imagem precisaria ser linearizada. Para lidar com imagens foi proposta uma nova classe de rede denominada *rede convolucionais* (LECUN et al., 1989). Esta analisa a imagem por blocos, o que preserva a informação espacial. As redes convolucionais (Figura 2.16) e tem sido amplamente utilizada em várias aplicações, principalmente envolvendo imagens (SUN et al., 2021; HAMILA et al., 2021; PODER, 2021).

Figura 2.16 - Modelo de uma rede convolucional.



Janelas de convolução deslizam sobre a imagem aprendendo atributos de alto nível, e entre cada camada a imagem é subamostrada para aprender atributos de baixo nível.

Fonte: Wikipedia (2020b).

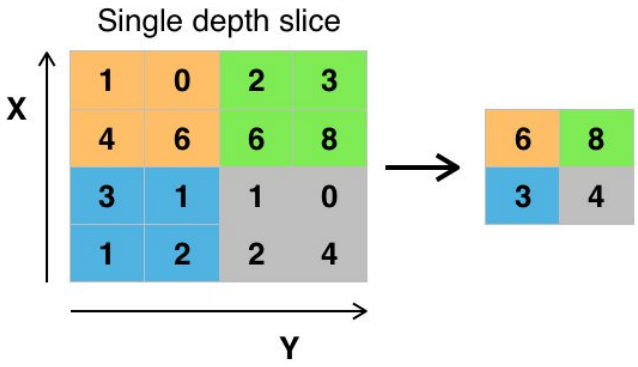
Essa classe de rede é capaz de extrair atributos de uma imagem através de operações de convolução. Uma janela de tamanho pré-definido desliza pela imagem e é capaz de aprender atributos (*feature maps*) da mesma. Os atributos podem ser de alto nível para uma imagem maior, ou de baixo nível para uma imagem menor. Quanto menor o nível, maior é a quantidade de detalhes que a rede consegue aprender sobre a imagem.

Cada janela de convolução é o *núcleo* da rede. Os núcleos costumam ser janelas de tamanho 3×3, 5×5, 7×7 e são posteriormente convoluídos com uma região de

mesmo tamanho da imagem. Cada camada convolucional é definida por um número de núcleos. Nas redes convolucionais, os pesos são os valores dos núcleos e cada neurônio é representado pelos valores de cada núcleo. Os núcleos são representados na Figura 2.16 pelas janelas na imagem e em cada camada.

Outro tipo de camada importante na rede convolucional é a camada de *pooling*, vista na Figura 2.17, que realiza uma sub-amostragem da imagem, reduzindo seu tamanho. A operação de *pooling* mais comum é a *max pooling*, em que um bloco da imagem é representado pelo maior elemento do bloco. Esta camada é representada na Figura 2.16 pela operação *subsampling*.

Figura 2.17 - Exemplo de uma operação MaxPooling.



Fonte: Wikipedia (2020b).

As redes convolucionais têm sido empregadas com sucesso em problemas de classificação (LECUN et al., 1998), segmentação (RONNEBERGER et al., 2015) e restauração de imagens (YE et al., 2020). As redes neurais, em geral, são independentes do tipo de degradação, o que representa uma vantagem em relação aos métodos anteriores. A rede Noise2Noise (LEHTINEN et al., 2018) ilustra um caso de restauração de imagens utilizando vários tipos de degradação, incluindo texto.

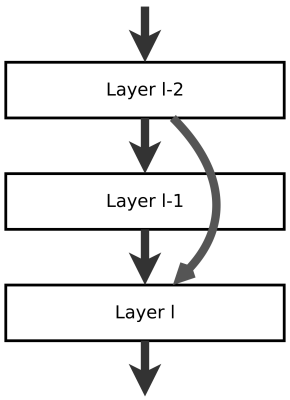
A principal contribuição de Lehtinen et al. (2018) foi mostrar que uma rede neural pode aprender a restaurar imagens utilizando dados alvo degradados, daí o nome *"Noise2Noise"*. Entretanto, devido aos resultados com imagens não degradadas apresentarem melhores resultados, este trabalho utilizará como referência o modelo apresentado anteriormente, com os dados fonte degradados e dados alvo

não degradados. Quando a rede é treinada com dados alvo degradados, é chamada "Noise2Noise" e quando é treinada com as imagens originais como dados alvo, é chamada "Noise2Clean". Ambas as redes serão referidas posteriormente como pelo nome em comum "rede convolucional".

Em termos de implementação, a rede convolucional original utiliza uma rede profunda do tipo RED30 (MAO et al., 2016). Neste trabalho, será utilizada uma arquitetura do tipo SRResNet (LEDIG et al., 2017).

A SRResNet é uma rede profunda com 35 camadas convolucionais projetada para problemas de super-resolução, mas que foi treinada com dados para restauração. Esta utiliza camadas residuais (HE et al., 2015) que permitem que a rede "salte" camadas durante o processo de aprendizagem. A Figura 2.18 ilustra este processo, onde a camada I-2 salta para a camada I, ignorando a camada I-1.

Figura 2.18 - Redes com residuais (ResNet).



Fonte: Wikipedia (2021b).

A rede neural descrita em Lehtinen et al. (2018) teve como conjunto de treinamento a quantidade de 50.000 imagens, porém a implementação disponível em Uchida (2018) utiliza apenas 291 imagens. Em ambos os casos, essas imagens são provenientes do banco de imagens *ImageNet* (IMAGENET, 2020) e são dos mais variados contextos. O conjunto de dados fonte foi formado utilizando ruído gaussiano variável de 0% a 50% de desvio padrão (UCHIDA, 2018) nessas imagens. A função custo utilizada no treinamento desta rede foi o erro médio quadrático, a função de otimização utilizada foi a *Adam* e o treinamento foi realizado com 60 épocas. A taxa de aprendizado foi

variável e decrescente a cada época, entre 0.01 e $0,00125$. A função de ativação utilizada foram a ReLU parametrizada, uma modificação da Equação 2.55 em que a variável z é multiplicada por um fator a, que é estimado durante o processo de aprendizagem.

2.3.2 Filtro Neural Multiescala

As redes e configurações citadas anteriormente necessitam de um banco de dados de imagens para gerar resultados próximos ao esperado. Uma outra abordagem para treinar redes neurais para restauração de imagens foi mencionada na sub-seção anterior, com a estratégia do filtro neural multi-escala (MSNF: *multiscale neural filter*) (CASTRO et al., 2008), que possui a vantagem de ter apenas **uma única imagem como conjunto de treinamento**. O processo geral deste filtro é visto na Figura 2.19. As três etapas do processo serão vistas a seguir.

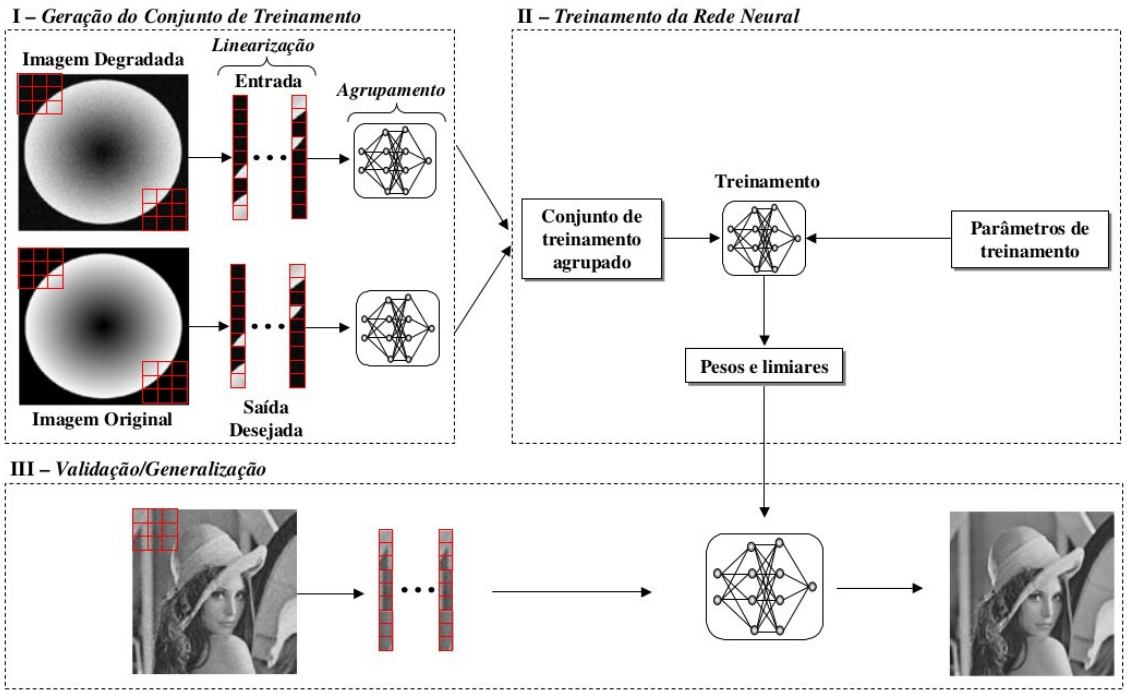
A imagem única utilizada para gerar o conjunto de treinamento é uma sobreposição de 256 círculos de raio 0 a 255, com valor de brilho igual ao valor do raio. A Figura 2.20 mostra à esquerda a imagem dos círculos da imagem não degradada, que gerará o conjunto de dados alvo (referência), e à direita a imagem dos círculos degradados – esta será utilizada para gerar os dados fonte (dados de entrada).

O termo "multiescala" se refere a uma nova janela que contém a união das três janelas mencionadas. A união é definida como a extração de 9 *pixels* de cada uma das janelas em torno de um *pixel* central, segundo o formato apresentado na Figura 2.21b. Assim uma outra janela 7×7 é obtida. Esta nova janela é também associada ao conjunto de treinamento.

No MSNF, o conjunto de treinamento é formado a partir de janelas (não convolucionais) de tamanhos 3×3, 5×5, 7×7 em torno de cada *pixel* da imagem degradada. Cada janela é associada ao *pixel* central correspondente da imagem limpa, como visto na Figura 2.21a. Sendo assim, o método MSNF considera as adjacências de cada *pixel* no conjunto de treinamento.

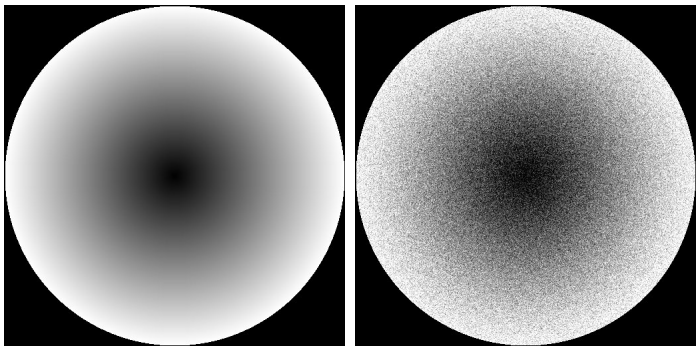
A Figura 2.22 mostra o processo de treinamento desta rede. Os vetores de entrada são enviados para uma camada escondida de 28 neurônios, cuja saída é enviada para a camada de saída e gera um pixel correspondente sem ruído. Após a extração, é realizado o processo de linearização (Figura 2.21a). Cada janela 3×3, 5×5, 7×7 é linearizada (cada coluna ao final da linha anterior), como um vetor de 9 elementos, e estes vetores são linearizados na ordem correspondente e transformados em um vetor

Figura 2.19 - Processo geral do filtro neural multiescala.



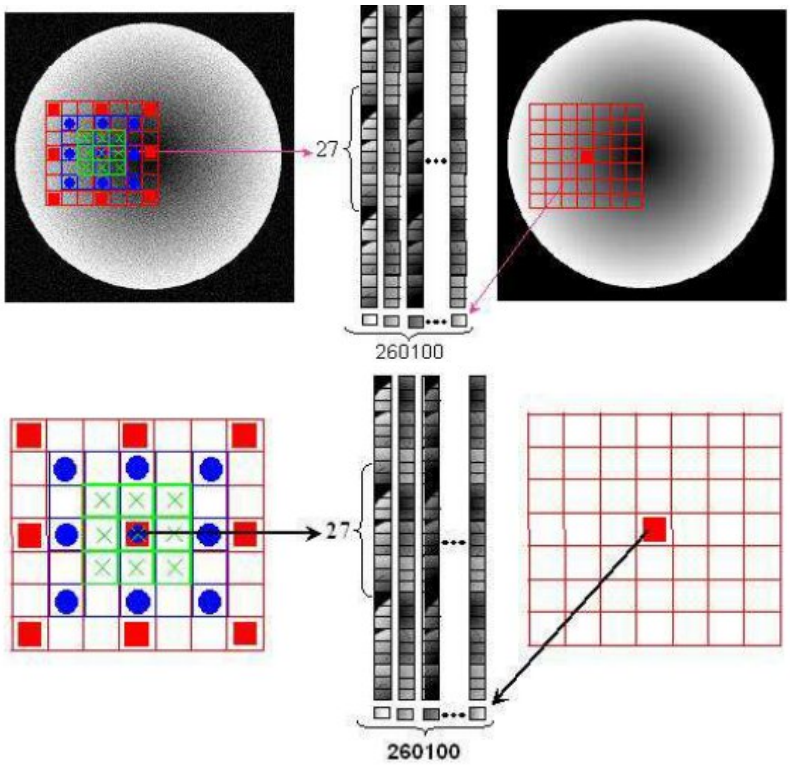
Fonte: Castro (2009).

Figura 2.20 - Imagens do conjunto de treinamento.



Fonte: Produção do autor.

Figura 2.21 - MSNF. (a) (b) filtro sub-amostrado da janela 7x7.



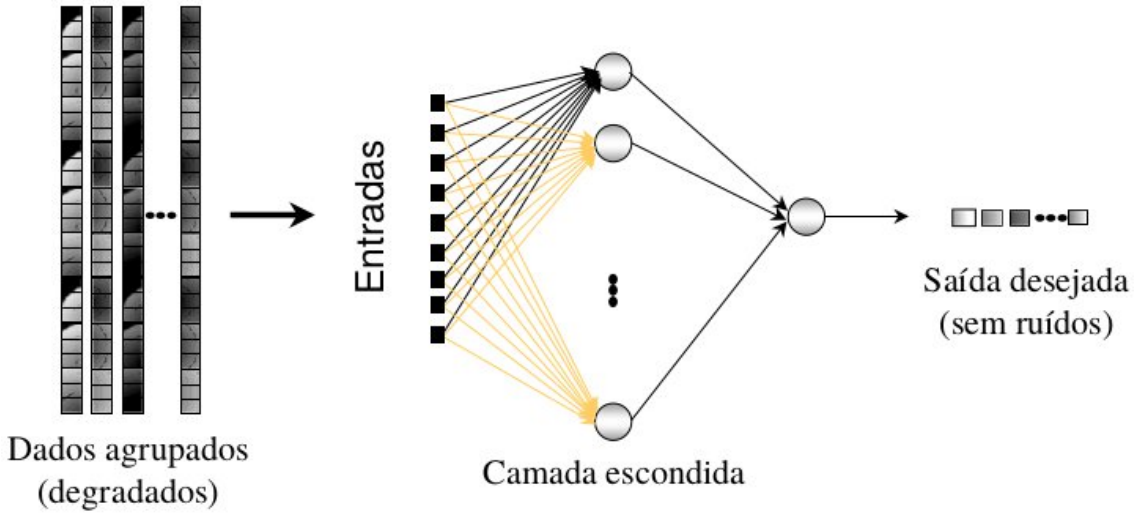
Fonte: Castro et al. (2008).

único contendo 27 elementos. A associação entre este vetor e o *pixel* correspondente ao elemento central da janela forma o conjunto de treinamento.

O processo de treinamento é realizado transformando o problema em um problema de regressão utilizando o erro médio quadrático como função custo. A restauração é realizada *pixel* a *pixel*, extraindo janelas da imagem a ser restaurada e encontrando o *pixel* correspondente.

A rede foi treinada inicialmente com uma camada de entrada com 27 neurônios, uma camada escondida com 28 neurônios, e uma saída. A função de ativação utilizada tanto para a camada escondida quanto a camada de saída foi a logística sigmoidal.

Figura 2.22 - Processo de treinamento do MSNF.



Fonte: Castro et al. (2008).

3 ASPECTOS COMPUTACIONAIS E TESTES DE VALIDAÇÃO

Neste capítulo são abordados diferentes aspectos computacionais relacionados ao problema central de análise de imagens e aos testes numéricos realizados para validação da metodologia proposta. Na Seção 3.1 são apresentadas as imagens teste utilizadas para a realização dos experimentos, na Seção 3.2 são apresentados os modelos de degradação de imagens considerados nos testes de validação, na Seção 3.3, as métricas para a avaliação da qualidade da restauração são apresentadas, e na Seção 3.4 alguns aspectos relevantes sobre o ambiente de execução são destacados com o intuito de permitir ao leitor a implementação e comparação dos resultados obtidos.

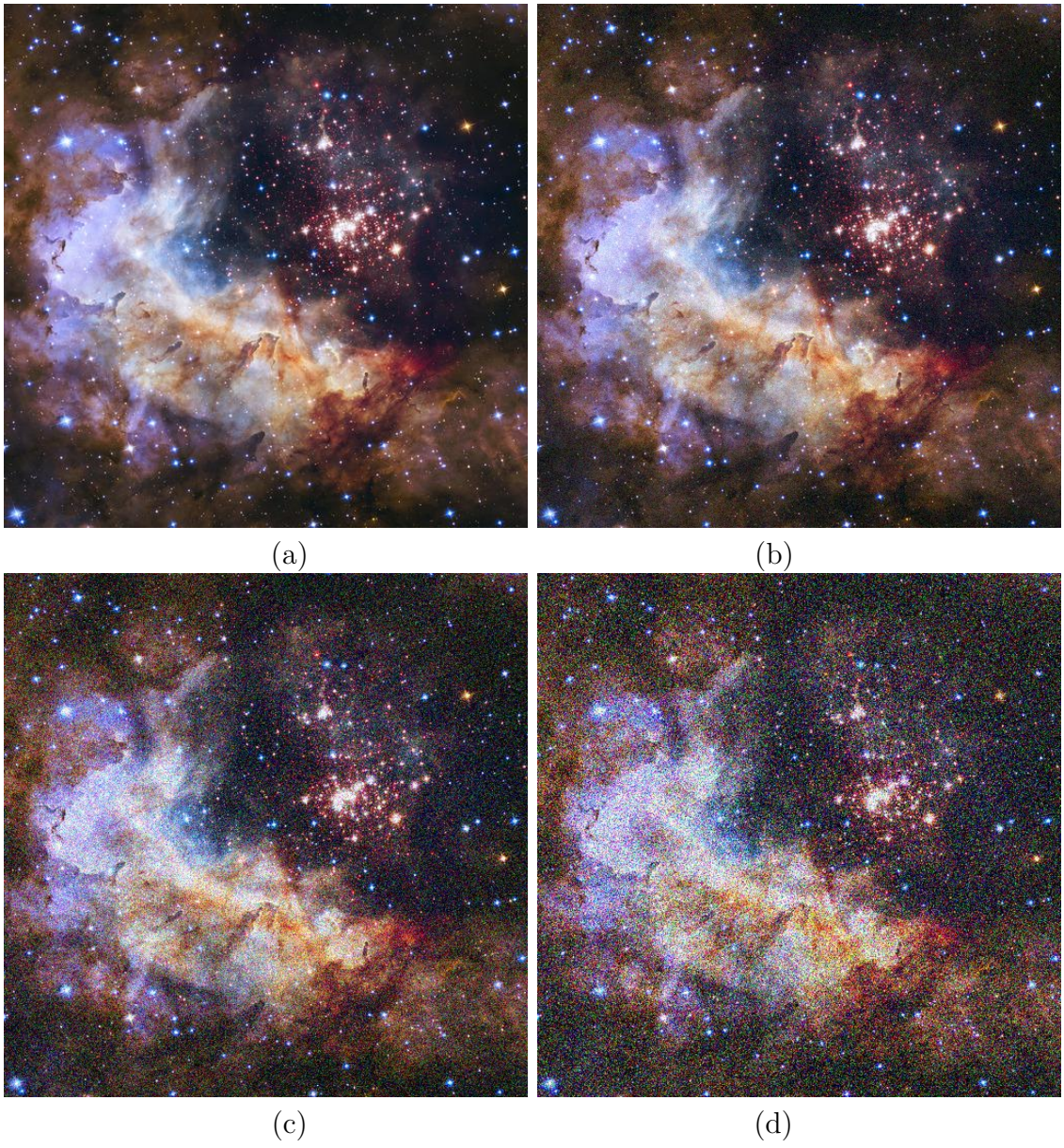
3.1 Imagens teste

Para a realização dos experimentos, foram escolhidas oito imagens astrônomicas provenientes da fonte HubbleSite (HUBBLESITE, 2020), que disponibiliza imagens do telescópio espacial Hubble sob uma licença compatível a domínio público. Em ordem de 1 a 8, as imagens astronômicas são:

- Westerlund
- NGC 3147
- Jupiter
- NGC 4258
- Cluster Phoenix
- Saturno
- NGC 6744
- Galáxia do Triângulo

As Figuras 3.1 a 3.8 mostram, da esquerda para a direita, cima para baixo, a imagem original, degradada 5%, degradada 15% e degradada 25%, respectivamente.

Figura 3.1 - Figura 1 - Westerlund.



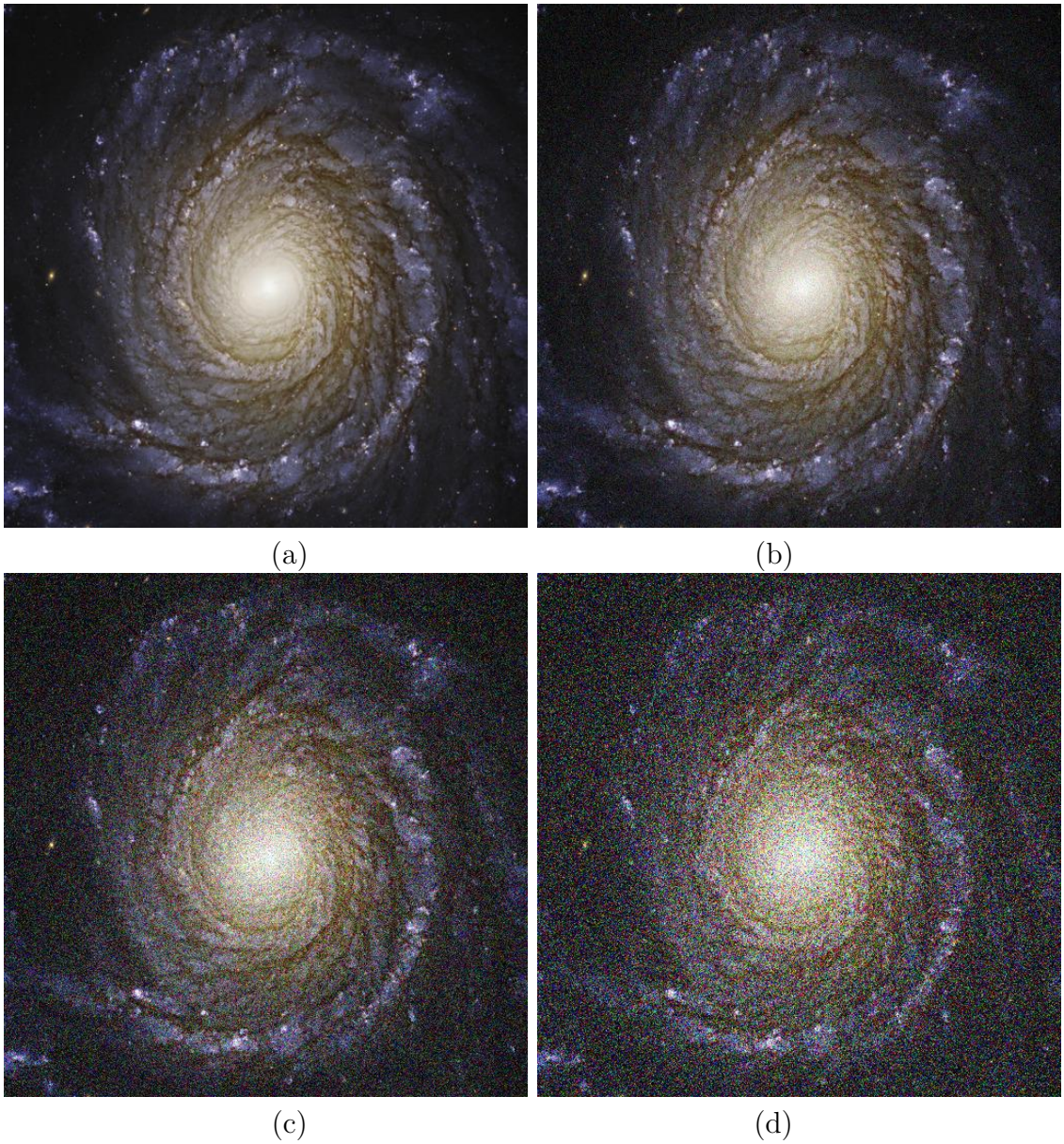
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.2 - Imagem NGC 3147.



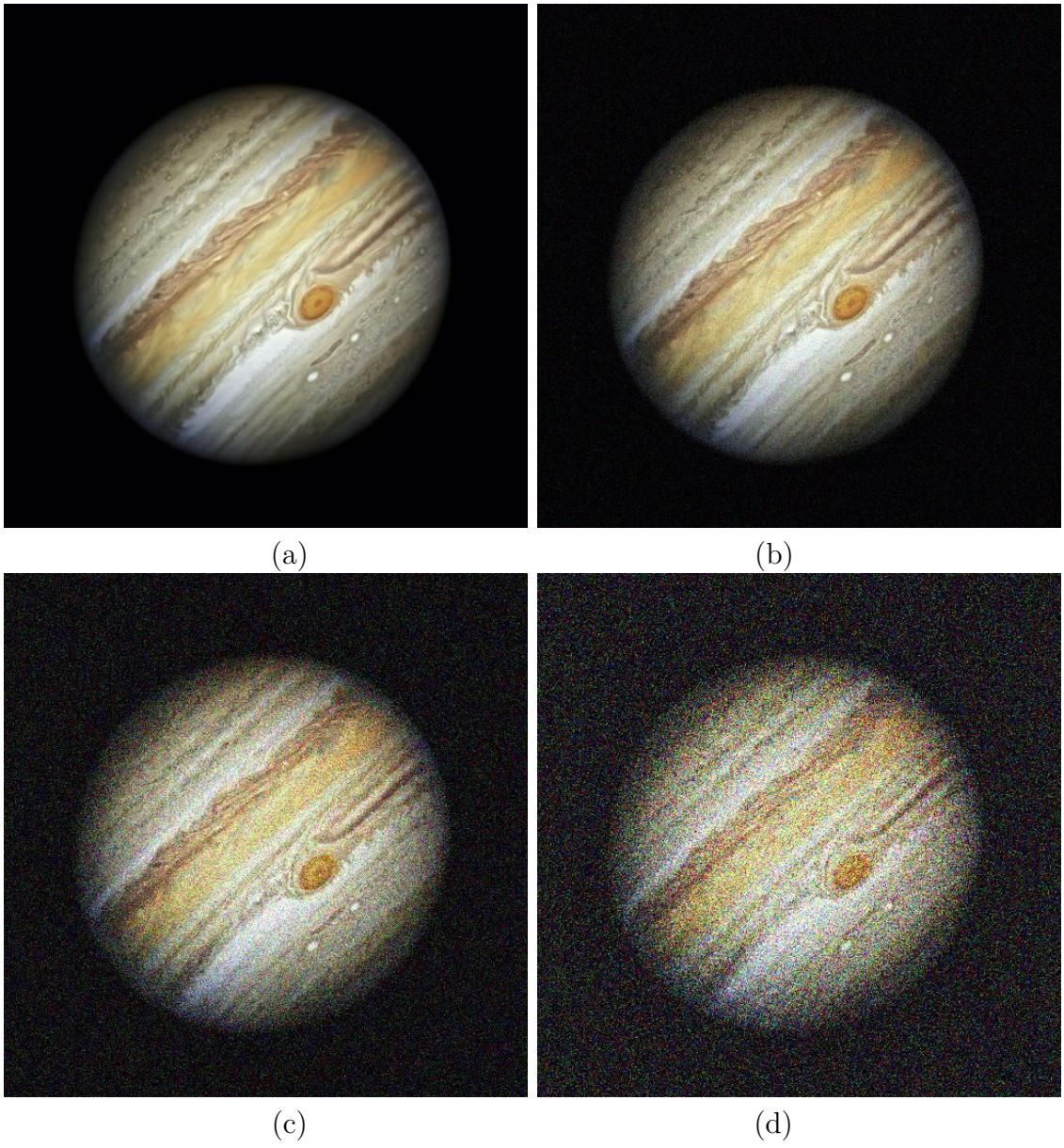
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.3 - Figura 3 - Jupiter.



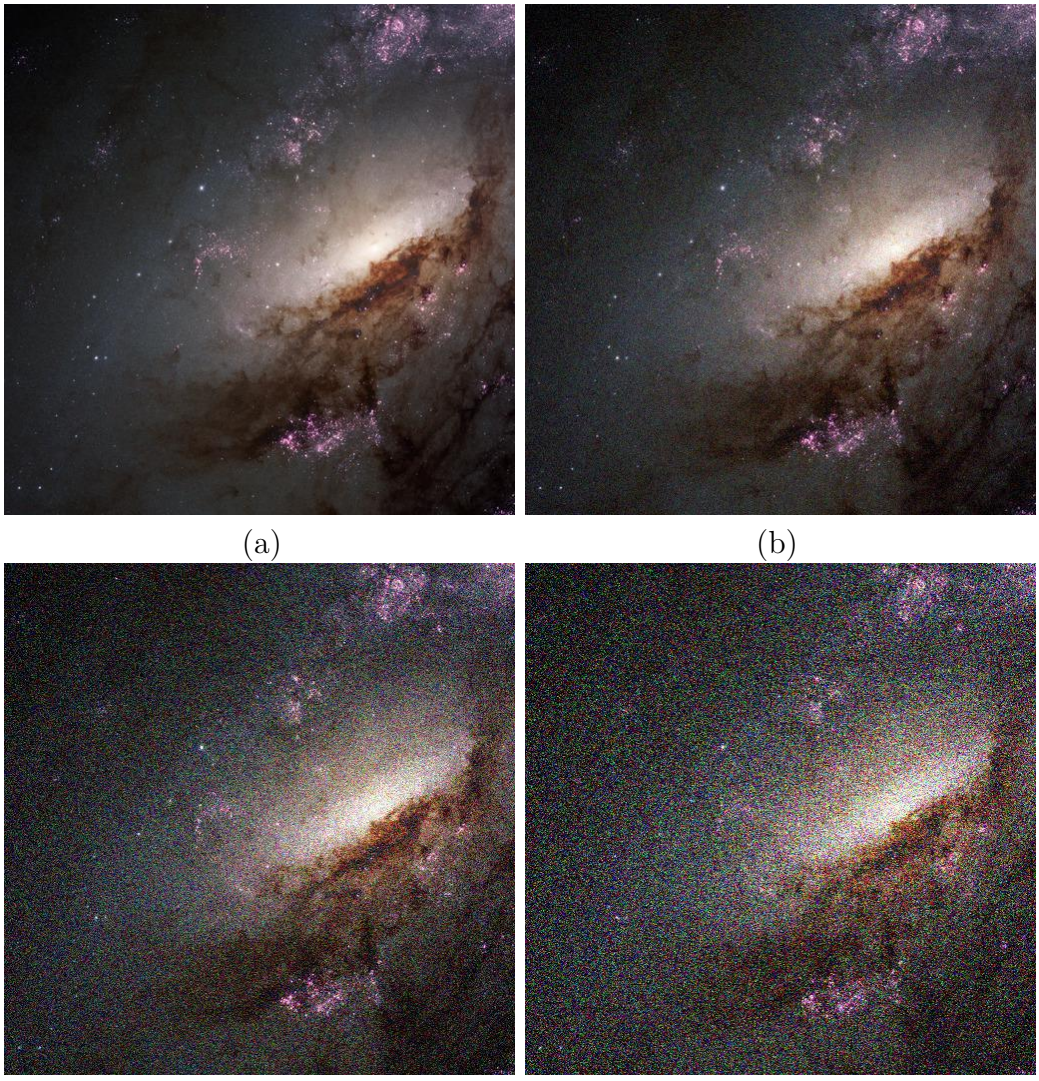
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.4 - Figura 4 - NGC 4258.



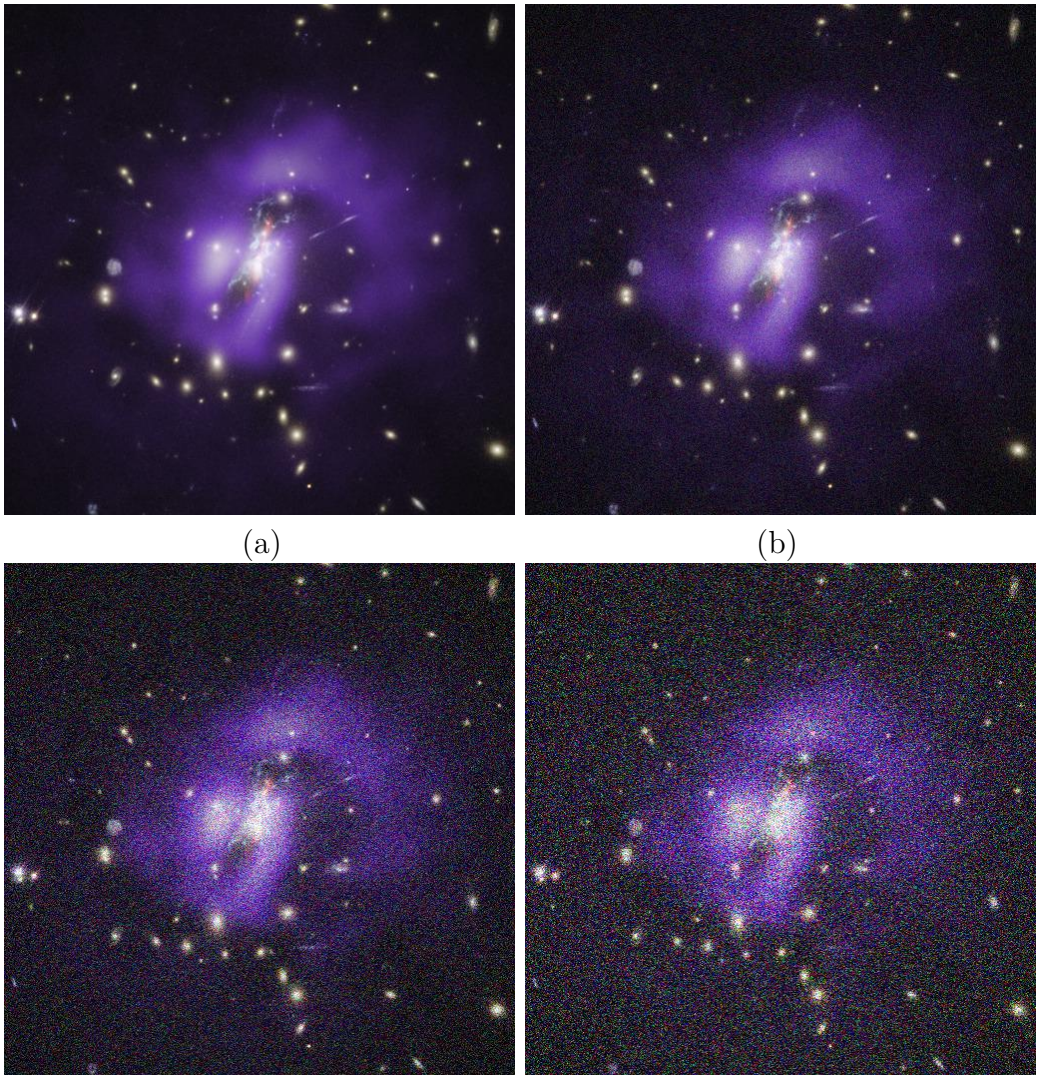
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.5 - Figura 5 - Cluster Phoenix.



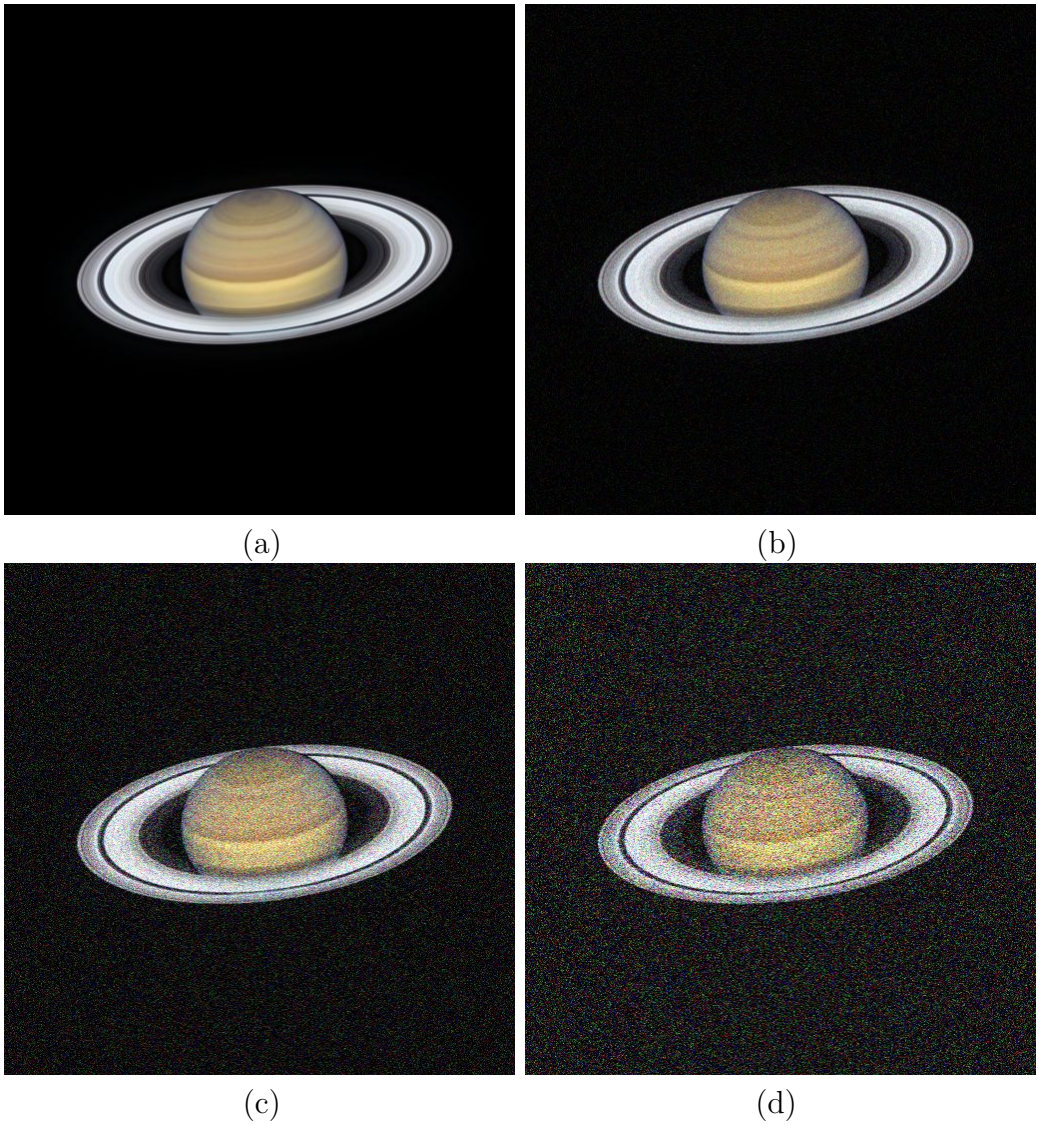
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.6 - Figura 6 - Saturno.



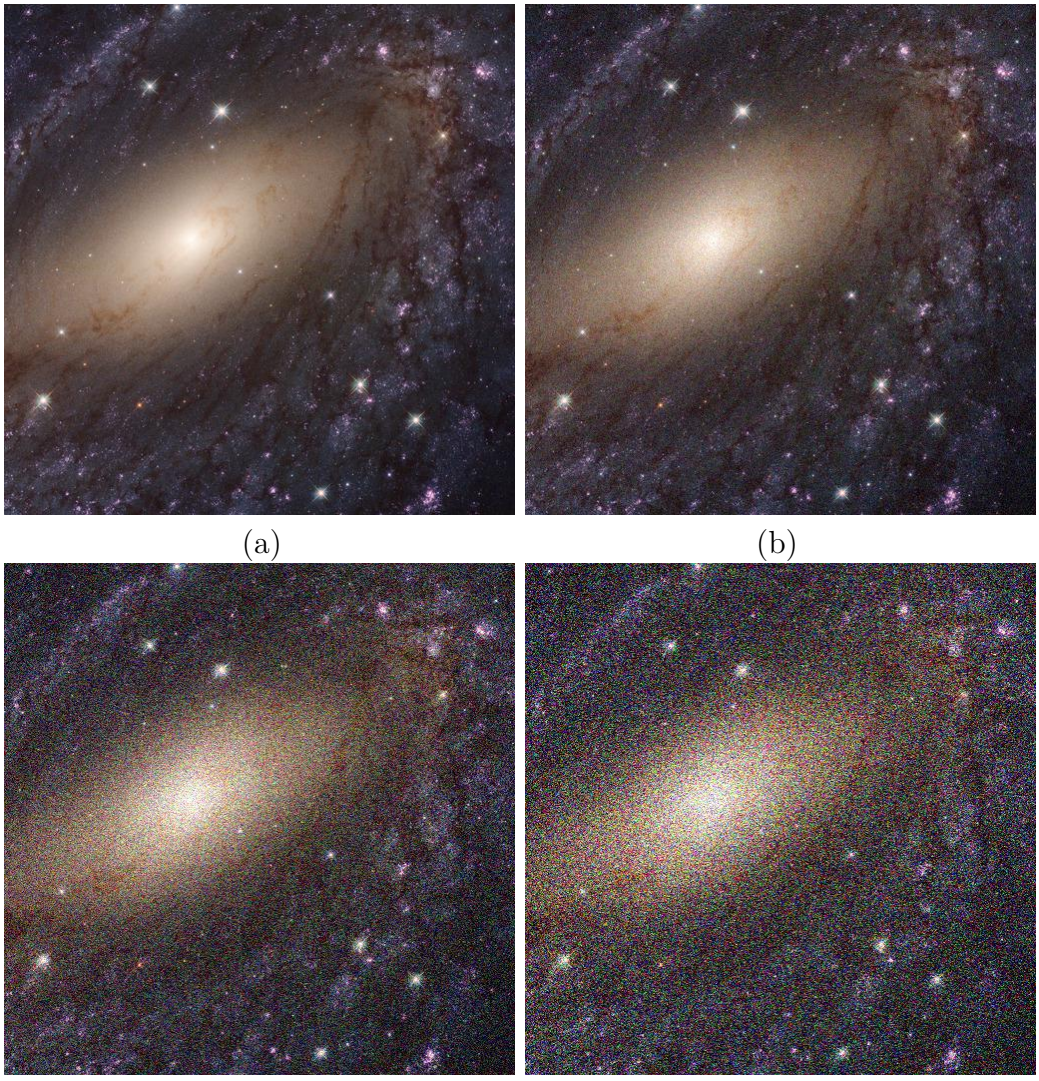
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.7 - Figura 7 - NGC 6744.



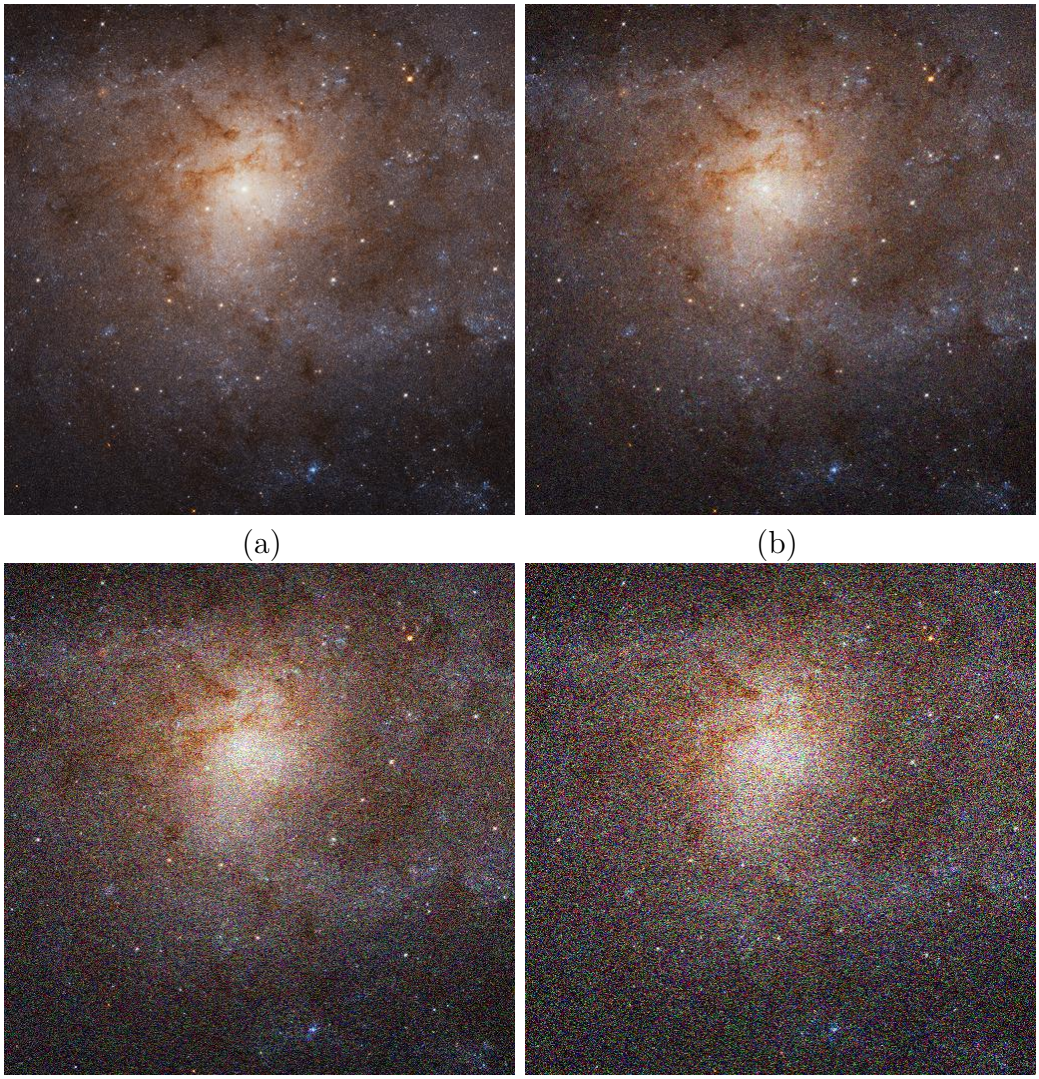
(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

Figura 3.8 - Figura 8 - Galáxia do Triângulo.



(a) (b)

(c) (d)

Das figuras acima: (a) original; e degradadas com ruído gaussiano de desvio padrão: (b) 5%, (c) 15%, (d) 25%.

Fonte: Produção do autor.

A escolha das imagens foi feita com base nas diferentes características de cada uma:

a) predominância de altas frequências (imagens 1, 7, 8), balanceamento entre altas e médias frequências (imagens 2, 4, 5) e predominância de baixas frequências (imagens 3, 6).

b) Diferentes objetos: planetas, galáxias, nebulosas. Características que podem dificultar o processo de restauração (estrelas que podem ser confundidas com ruído).

3.2 Modelo de degradação

Todas as imagens acima mencionadas foram redimensionadas (*degradadas*) para o tamanho padrão de 512×512 pixels e, caso necessário, recortadas para uma dimensão quadrada preservando as características mais relevantes para os critérios de escolha das imagens.

As imagens a serem restauradas são imagens sintéticas obtidas adicionando um ruído gaussiano aditivo com desvio padrão σ à imagem I_{ij}, obtendo a imagem ruidosa I_{ij}^{σ}:

$$I_{ij}^{\sigma} = I_{ij}(1 + \sigma\,\nu_{ij}), \tag{3.1}$$

sendo ν_{ij} uma variável aletória com distribuição gaussiana:

$$P(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(z-\mu)^2/(2\sigma^2)} \tag{3.2}$$

com média $\mu = 0$ e desvios padrão $\sigma = 5\%, 15\%$ e 25%, compondo 3 tipos de imagens degradadas. As oito imagens foram consideradas na sua forma colorida. Na restauração das imagens, foi considerada a representação RGB, exceto para a restauração com wavelets, onde foi considerada a representação YCbCr – ver Seção 2.2.5.2. O ruído é adicionado individualmente a cada canal RGB.

3.3 Métricas de avaliação

Após a obtenção das imagens com ruído, o processo de restauração pelos métodos anteriormente descritos foi iniciado. A avaliação da qualidade da restauração pode ser realizada por diferentes métricas. Neste trabalho, as métricas (avaliação da qualidade da restauração) utilizadas para comparação foram o erro médio quadrático normalizado (NRMSE: *Normalized Root-Mean-Squared Error – Erro médio quadrático normalizado*, ver Equação 3.3), a *Peak Signal-to-Noise Ratio* (PSNR, ver Equação 3.4) e a similaridade estrutural (SSIM: Structural Similarity Index Measure).

3.3.1 Erro médio quadrático normalizado

O *NRMSE* calcula o erro médio quadrático normalizado entre duas imagens. Este método é calculado pela normal euclidiana do erro entre a imagem original I e a imagem degradada I_{ref} relativo à norma euclideana da imagem original. Ele é dado pela Equação 3.3:

$$NRMSE = \frac{||I - I_{ref}||}{||I||}. \tag{3.3}$$

3.3.2 Razão sinal-ruído de pico

A PSNR é descrita pela Equação 3.4:

$$PSNR = 10 \log_{10} \frac{L^2}{MSE}, \tag{3.4}$$

em que L é o maior valor de brilho representável pela imagem. No caso das imagens representadas em ponto flutuante, $L = 1$. No caso de valores inteiros de 8 bits, $L = 255$.

3.3.3 Medida de índice de similaridade estrutural

A medida de índice de similaridade estrutural (SSIM) é uma métrica de similaridade entre uma imagem e a sua referência. Ela é vista como a combinação entre as comparações de luminância, contraste e estrutura entre as duas imagens – ver Figura 3.9.

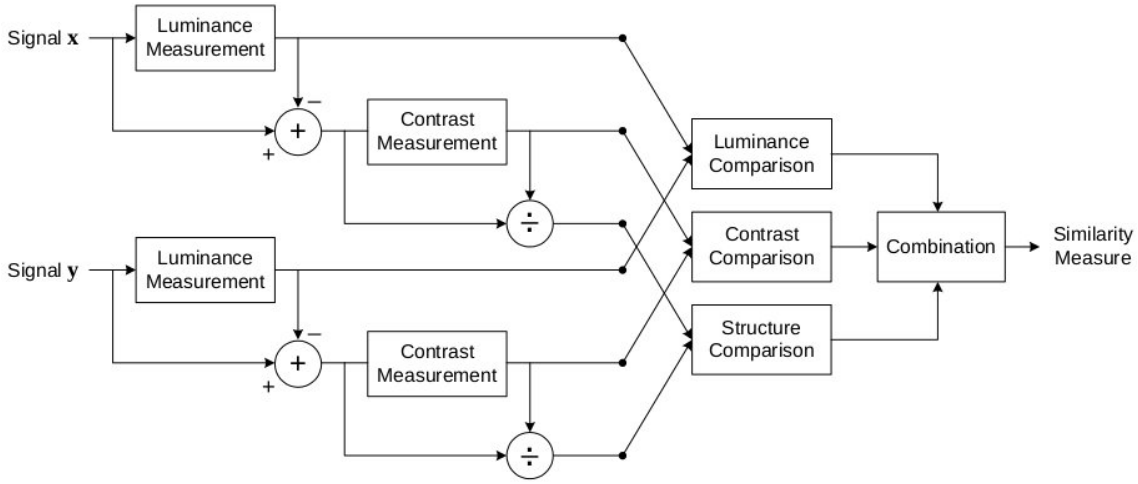
O SSIM utiliza duas janelas deslizantes x e y calculando luminância, contraste e estrutura por blocos. A luminância l, contraste c e estrutura s são dados nas Equações 3.5 a 3.7:

$$l(x, y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}, \tag{3.5}$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}, \tag{3.6}$$

$$s(x, y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma y + c_3}, \tag{3.7}$$

Figura 3.9 - Diagrama do SSIM.



Fonte: Wang et al. (2004).

em que μ são as médias e σ o desvio padrão de cada janela, e σ_{xy} a covariância entre as janelas x e y. O SSIM é dado como combinação entre $l(x, y)$, $c(x, y)$ e $s(x, y)$ como na Equação 3.8:

$$SSIM(x, y) = l(x, y) \times c(x, y) \times s(x, y). \tag{3.8}$$

A Equação 3.8 pode ainda ser ponderada atribuindo pesos a cada fator, mas assume-se que os três têm o mesmo peso. As constantes c_1, c_2 e c_3 são dadas na Equação 3.9 a 3.11, em que L representa a mesma variável da Equação 3.4.

$$c_1 = (k_1 L)^2 \ , \qquad k_1 = 0.01 \tag{3.9}$$

$$c_2 = (k_2 L)^2 \ , \qquad k_2 = 0.03 \tag{3.10}$$

$$c_3 = \frac{c_2}{2} \ . \tag{3.11}$$

A SSIM não é aplicada na imagem inteira, mas em janelas de tamanho arbitrário, sendo que neste trabalho o tamanho utilizado foi igual a 7. Estas janelas são deslocadas com passo igual a 1 pixel e percorrem toda a imagem. A amplitude do resultado é um valor entre 0 e 1. A similaridade é dita total para $SSIM(x, y) = 1$, e dissimilaridade total quando for 0. Assim, o resultado pode ser interpretado como

uma *porcentagem de similaridade*.

3.4 Ambiente de execução

O ambiente de execução dos testes é composto de um interpretador Python 3.7.3, Debian 10, os pacotes *scikit-image* (WALT et al., 2014) versão 0.17.3, SciPy (VIRTANEN et al., 2020) versão 1.4.1, *TensorFlow* (ABADI et al., 2015) versão 2.3, PyWavelets (LEE et al., 2019) versão 1.1.1 e Octave versão 4.4.0.

Os algoritmos considerados para as transformadas Wavelet biortogonal direta e inversa, disponíveis no pacote *PyWavelets*, foram testados fazendo as decomposições e reconstruções para diferentes dados iniciais. O erro obtido em cada um dos casos foi da ordem de 10^{-12}, tanto para dados periódicos quanto para dados não periódicos.

As três métricas utilizadas (NRMSE, PSNR, SSIM) estão disponíveis no pacote *scikit-image* e no sub-pacote *metrics*, que foram utilizadas com os parâmetros padrão.

4 RESULTADOS

Neste capítulo, os resultados são apresentados para restauração de imagens pelos métodos: (i) regularização (Tikhonov-2, variação total, Wiener não supervisionado e entropia), (ii) transformada *wavelet*, (iii) rede convolucional e (iv) filtro neural multiescala. Os casos para diferentes níveis de ruído são avaliados pelas métricas NRMSE, PSNR e SSIM, citadas anteriormente.

Para a comparação visual, são apresentadas 6 imagens dispostas em uma grade 3×2. A coluna à esquerda apresenta as imagens degradadas, sendo que em cada linha a imagem contém um nível de ruído distinto (em ordem de cima para baixo: 5%, 15%, 25%). A coluna à direita apresenta a restauração correspondente, de acordo com o método escolhido.

As tabelas apresentam as métricas obtidas em cada resultado. Cada linha é uma imagem (1 a 8, ver Seção 3.1) e cada uma das nove colunas uma métrica. As colunas são agrupadas em três de três elementos, sendo três níveis de ruído e três métricas para cada. A Tabela 4.1 apresenta estas métricas para 8 imagens degradadas mostradas na Seção 3.1, para exemplo e comparação da qualidade das restaurações:

Tabela 4.1 - Métricas das imagens degradadas.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.129	26.07	0.75	0.358	17.21	0.32	0.552	13.45	0.18
2	0.171	26.09	0.7	0.468	17.35	0.26	0.72	13.61	0.13
3	0.121	27.53	0.61	0.353	18.23	0.15	0.566	14.14	0.06
4	0.178	26.13	0.67	0.492	17.32	0.23	0.756	13.59	0.11
5	0.214	26.22	0.64	0.562	17.84	0.22	0.871	14.04	0.11
6	0.152	28.33	0.51	0.449	18.91	0.12	0.734	14.64	0.06
7	0.158	26.03	0.72	0.446	17.04	0.26	0.683	13.34	0.13
8	0.157	26.02	0.75	0.445	16.99	0.29	0.683	13.26	0.14

4.1 Regularização

Resultados são apresentados para as técnicas de regularização: Tikhonov-2, Entropia e filtro de Wiener não-supervisionado. Shiguemori e co-autores (SHIGUEMORI et al., 2017) avaliaram a regularização de Tikhonov de ordens zero, um e dois e foi o operador Tikhonov-2 que apresentou os melhores resultados. Desta forma, somente

este tipo de regularizador foi aplicado ao processo de restauração.

4.1.1 Tikhonov-2

O método Tikhonov utilizado foi uma conversão 1:1 em Python da implementação em MATLAB que foi utilizada em (SHIGUEMORI et al., 2017). O valor do parâmetro de regularização α foi estimado pelo método da discrepância de Morozov, resultando em $\alpha \approx 2.2$. A solução foi calculada no domínio da frequência.

As Figuras 4.1 e 4.2 apresentam os resultados visuais para a regularização por Tikhonov de ordem 2. Nota-se que a restauração introduz um borramento, ou suavização excessiva na imagem, principalmente para o nível de ruído mais baixo.

A Tabela 4.2 mostra as métricas da restauração obtida pela regularização Tikhonov-2. Nota-se uma leve degradação no caso do ruído baixo (5%), mas a qualidade da restauração é bem mais evidente com um nível de ruído maior.

Tabela 4.2 - Métricas para a regularização *Tikhonov-2*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.156	24.45	0.79	0.168	23.76	0.75	0.201	22.22	0.68
2	0.129	28.52	0.86	0.157	26.84	0.8	0.222	23.84	0.72
3	0.076	31.63	0.75	0.152	25.54	0.49	0.245	21.41	0.4
4	0.123	29.39	0.87	0.154	27.39	0.81	0.224	24.17	0.73
5	0.097	33.09	0.96	0.16	28.78	0.88	0.278	23.94	0.77
6	0.166	27.53	0.59	0.269	23.36	0.26	0.403	19.85	0.19
7	0.134	27.51	0.79	0.154	26.3	0.74	0.197	24.15	0.68
8	0.151	26.4	0.69	0.168	25.43	0.66	0.207	23.62	0.6

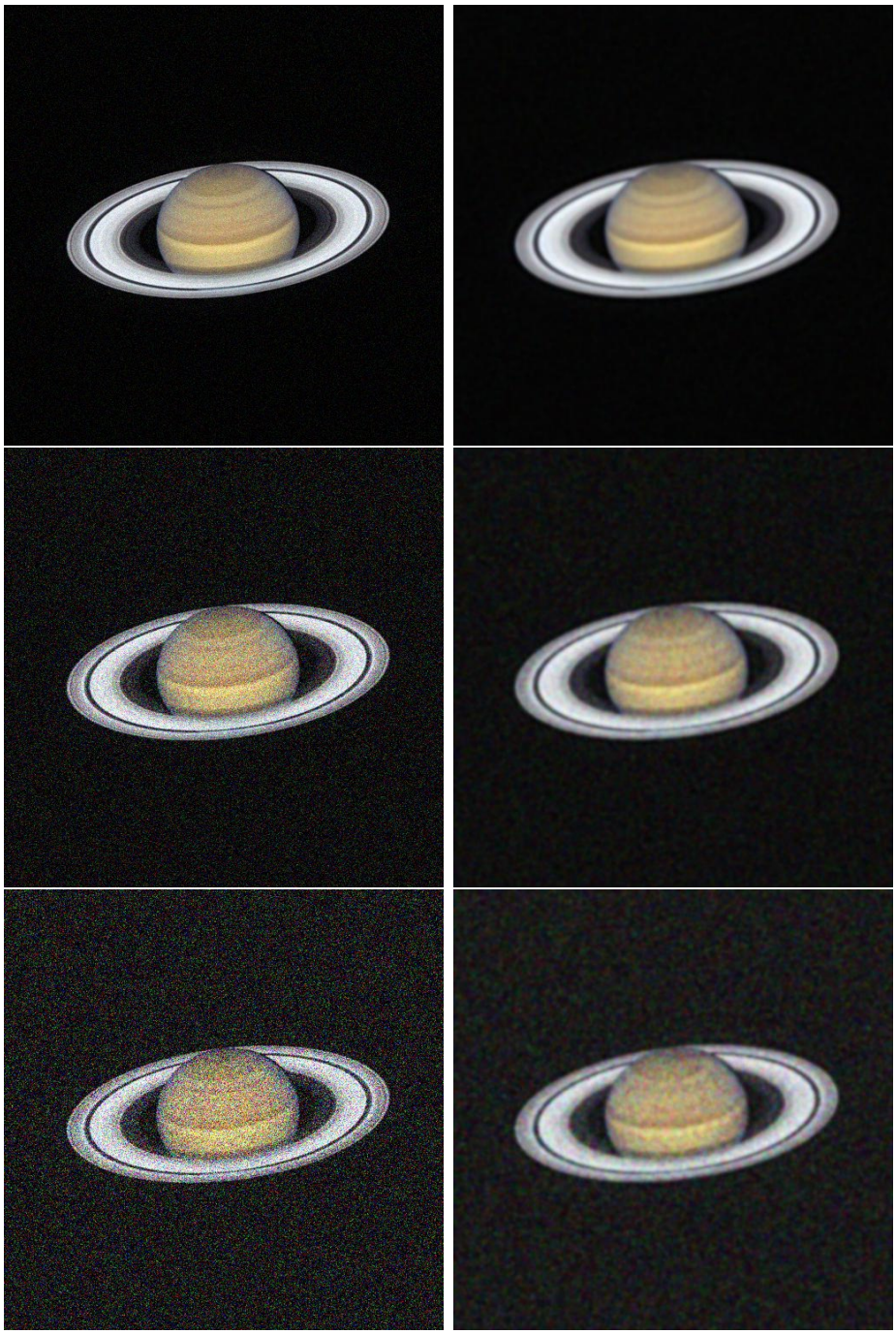
A Tabela 4.3 mostra as métricas da restauração obtida pela regularização Tikhonov-1. Nota-se que, pelas métricas, este método é levemente inferior ao Tikhonov-2.

Figura 4.1 - Resultados da imagem Westerlund com a regularização Tikhonov-2.

Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.2 - Resultados da imagem Saturno com a regularização Tikhonov-2.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Tabela 4.3 - Métricas para a regularização *Tikhonov-1*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.161	24.16	0.78	0.173	23.51	0.74	0.205	22.05	0.67
2	0.133	28.27	0.85	0.158	26.76	0.8	0.219	23.95	0.72
3	0.083	30.86	0.74	0.156	25.34	0.48	0.247	21.34	0.38
4	0.125	29.21	0.87	0.155	27.37	0.81	0.222	24.24	0.72
5	0.106	32.36	0.95	0.163	28.62	0.88	0.28	23.9	0.77
6	0.178	26.94	0.58	0.278	23.08	0.25	0.41	19.7	0.18
7	0.137	27.32	0.79	0.154	26.26	0.74	0.194	24.26	0.68
8	0.151	26.38	0.69	0.166	25.53	0.66	0.203	23.78	0.6

A Tabela 4.4 mostra as métricas da restauração obtida pela regularização Tikhonov-0. Neste caso, há degradação. Todas as métricas são inferiores à da imagem degradada.

Tabela 4.4 - Métricas para a regularização *Tikhonov-0*.

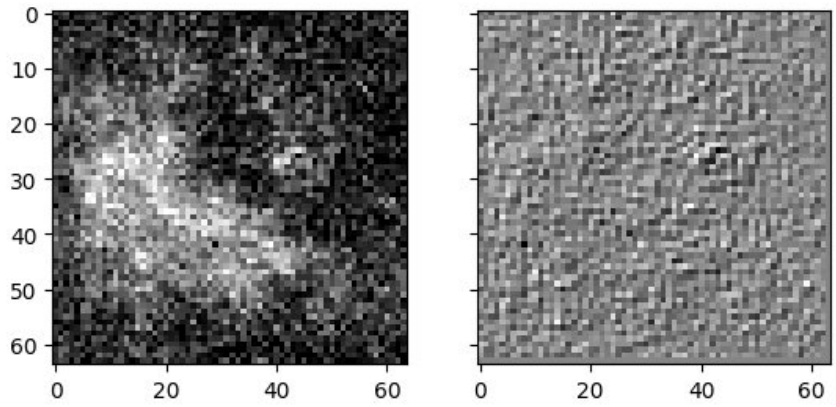
σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.838	9.83	0.23	0.839	9.81	0.22	0.841	9.8	0.22
2	0.836	12.31	0.27	0.837	12.3	0.26	0.836	12.31	0.27
3	0.834	10.77	0.66	0.837	10.74	0.59	0.842	10.69	0.47
4	0.836	12.71	0.29	0.838	12.7	0.28	0.837	12.71	0.28
5	0.835	14.4	0.32	0.835	14.4	0.34	0.833	14.42	0.36
6	0.837	13.5	0.81	0.841	13.46	0.66	0.849	13.38	0.48
7	0.837	11.57	0.24	0.839	11.56	0.23	0.838	11.56	0.22
8	0.838	11.49	0.21	0.84	11.47	0.2	0.839	11.48	0.19

4.1.2 Entropia

A não linearidade do operador de entropia aumenta o esforço computacional, pois é necessário resolver um problema de otimização com $M \times N$ variáveis (número de pixels da imagem). Para avaliar o método, a dimensão ou resolução da imagem foi reduzida para 64×64 pixels, sendo mapeada para tons de cinza. Similar a regularização de Tikhonov, a solução inversa no domínio espacial pode ser obtida com operadores entrópicos de diferentes ordens, que requer a identificação de distintos parâmetros de regularização, como em outros métodos deste tipo de estratégia.

Mesmo com a redução da resolução da imagem, os resultados não foram razoáveis com a entropia, testada com todas as ordens, utilizando o otimizador de gradiente conjugado sem precondicionamento implementado na função `optimize.fmin_cg` no pacote SciPy e a própria imagem degradada como condição inicial. A Figura 4.3 mostra uma tentativa de restauração por este método utilizando uma convolução da PSF utilizada na restauração pelo funcional de Tikhonov, com parâmetro de regularização $\alpha = 0,5$, onde a própria imagem degradada é usada como estimativa inicial. O elevado custo computacional também dificultou a experimentação numérica dos parâmetros livres.

Figura 4.3 - Tentativa de regularização por entropia.



À esquerda: image Westerlund degradada com ruído 15%. Direita: tentativa de restauração por regularização de entropia.

Fonte: Produção do autor.

4.1.3 TVR

Para este método, foi utilizada a implementação disponível no pacote *scikit-image*, na função `restoration.denoise_tv_chambolle`, com o algoritmo Chambolle. O valor do parâmetro de regularização $\alpha = 0.1$ – Equação 2.25 – com 200 iterações foi usado nos experimentos de restauração.

A Tabela 4.1.3 mostra os resultados para o método *TVR*. Conforme mencionado, este método foi proposto para melhorar a suavização que ocorre na regularização com operadores lineares como o de Tikhonov. É possível observar na Tabela 4.1.3 uma melhoria das métricas deste método em todas as imagens com ruído de até

15%, enquanto Tikhonov-2 apresenta métricas melhores para o caso com ruído de 25%.

Tabela 4.5 - Métricas para o método *TVR*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.095	28.7	0.88	0.127	26.24	0.81	0.236	20.85	0.53
2	0.099	30.86	0.9	0.14	27.86	0.83	0.328	20.44	0.46
3	0.061	33.47	0.77	0.148	25.81	0.49	0.283	20.17	0.27
4	0.095	31.62	0.9	0.138	28.35	0.83	0.308	21.39	0.48
5	0.061	37.14	0.97	0.144	29.66	0.89	0.363	21.64	0.53
6	0.092	32.64	0.62	0.241	24.31	0.27	0.417	19.55	0.15
7	0.112	29.08	0.82	0.138	27.22	0.77	0.275	21.25	0.47
8	0.133	27.51	0.74	0.154	26.17	0.7	0.283	20.91	0.45

As Figuras 4.4 e 4.5 mostram os resultados com a filtragem TVR. Nota-se que o ruído gaussiano é mais aparente em relação à Tikhonov-2. Entretanto, a imagem é mais nítida neste método enquanto a restauração por Tikhonov-2 introduz borramento excessivo na imagem.

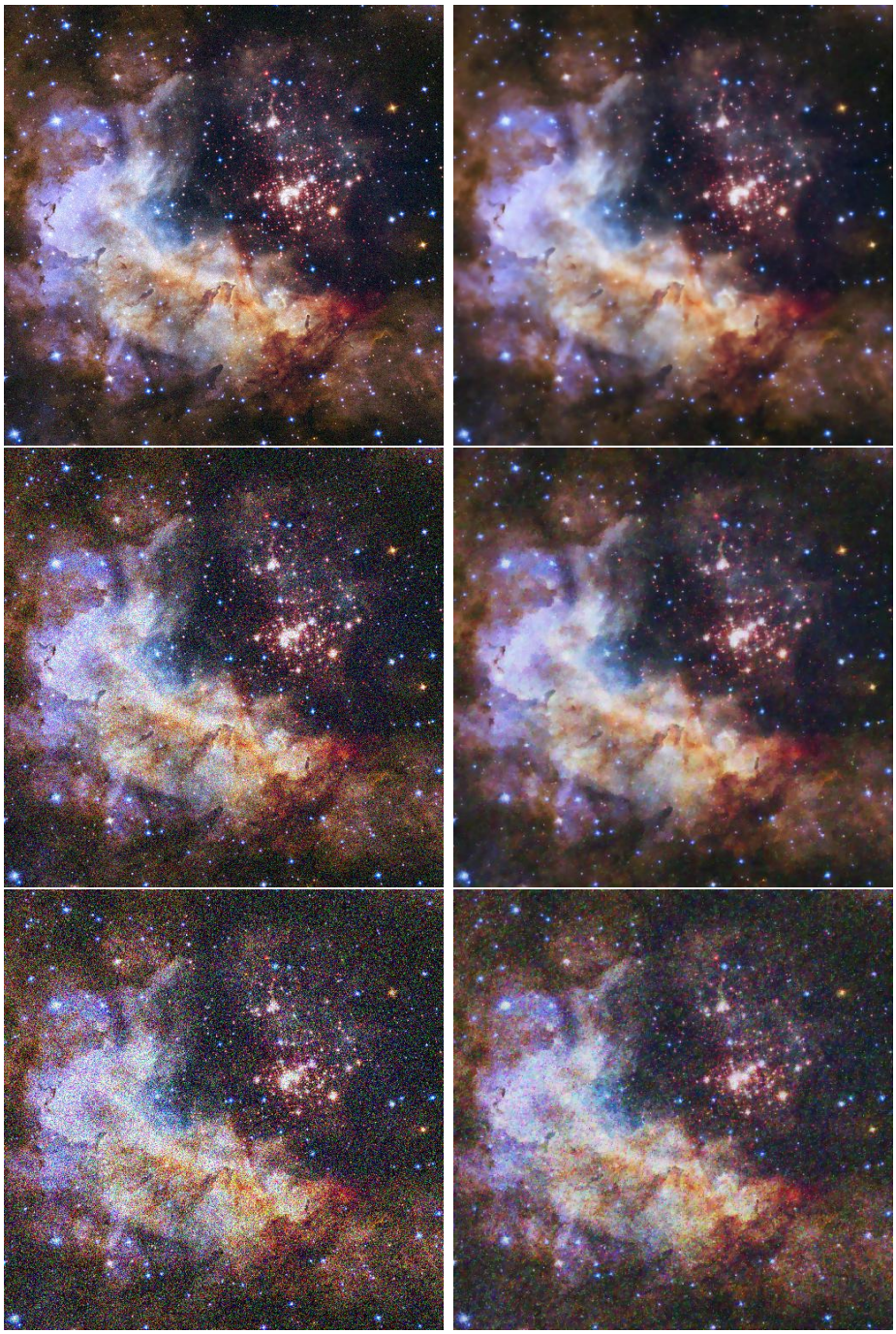
4.1.4 Filtro de Wiener

O filtro de Wiener é uma das técnicas mais difundidas e mais utilizadas em restauração de imagens. Sua popularidade é justificada devido a facilidade de implementação, boa qualidade da imagem restaurada e com muitas implementações atualmente disponíveis. o filtro de Wiener não supervisionada empregado foi da implementação do pacote *scikit-image*, disponível na função `restoration.unsupervised_wiener`. O valor de α foi estimado pelo método descrito em Orieux et al. (2010).

Os resultados com o filtro de Wiener correspondem à implementação de Wiener não-supervisionado como disponibilizado no pacote *scikit-image*. A não-supervisão significa que o parâmetro de regularização é estimado por um método automático, no caso *Gibbs Sampler* (ORIEUX et al., 2010).

As Figuras 4.6 e 4.7 mostram os resultados obtidos com esse filtro. Visualmente, os resultados são semelhantes ao filtro Tikhonov-2 para os casos de ruído 5% e 15%. Para o caso de ruído 25%, a restauração por Tikhonov-2 apresenta uma suavização maior da imagem, enquanto Wiener apresenta uma restauração mais discreta.

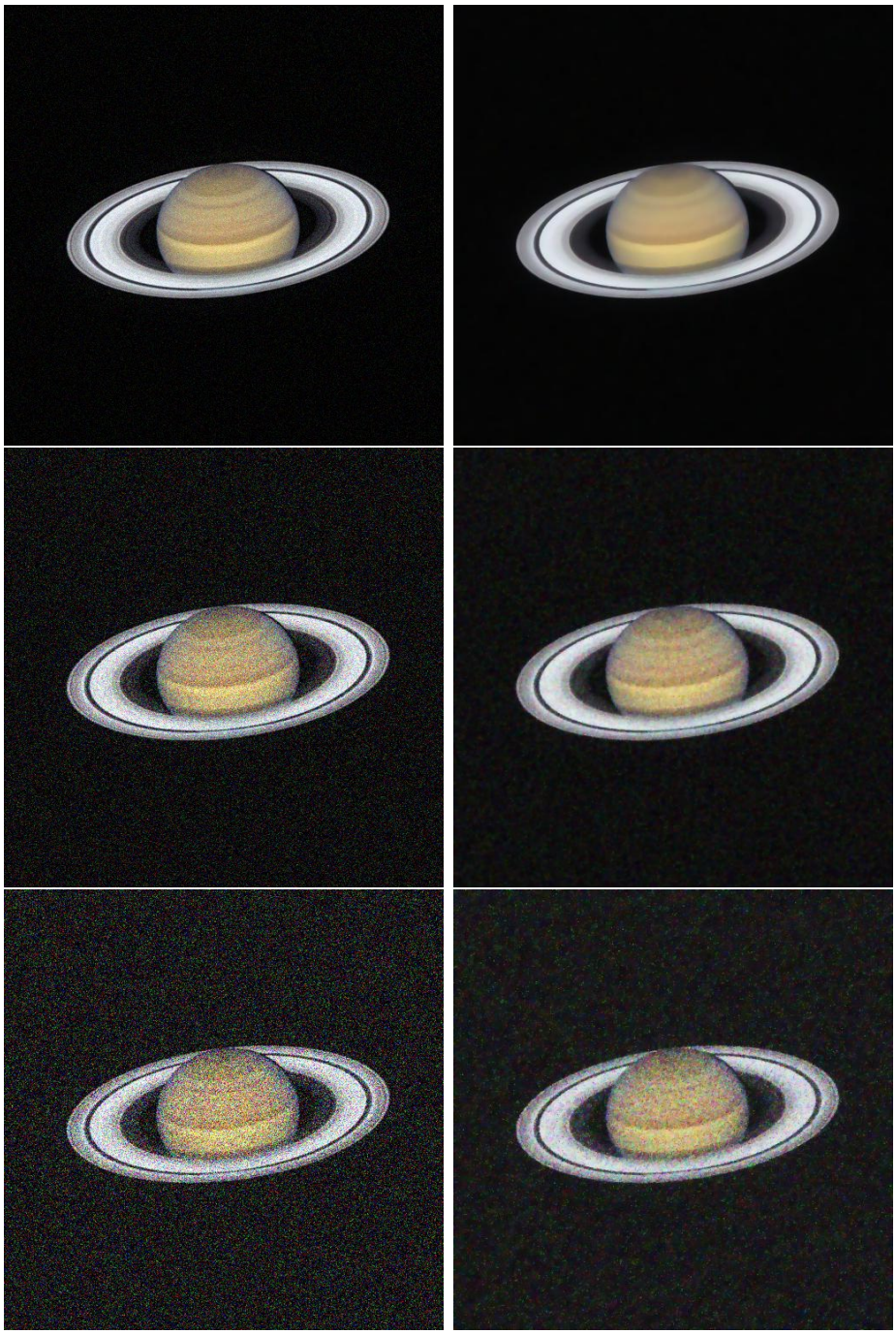
Figura 4.4 - Resultados da imagem Westerlund com o filtro TVR.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.5 - Resultados da imagem Saturno com o filtro TVR.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

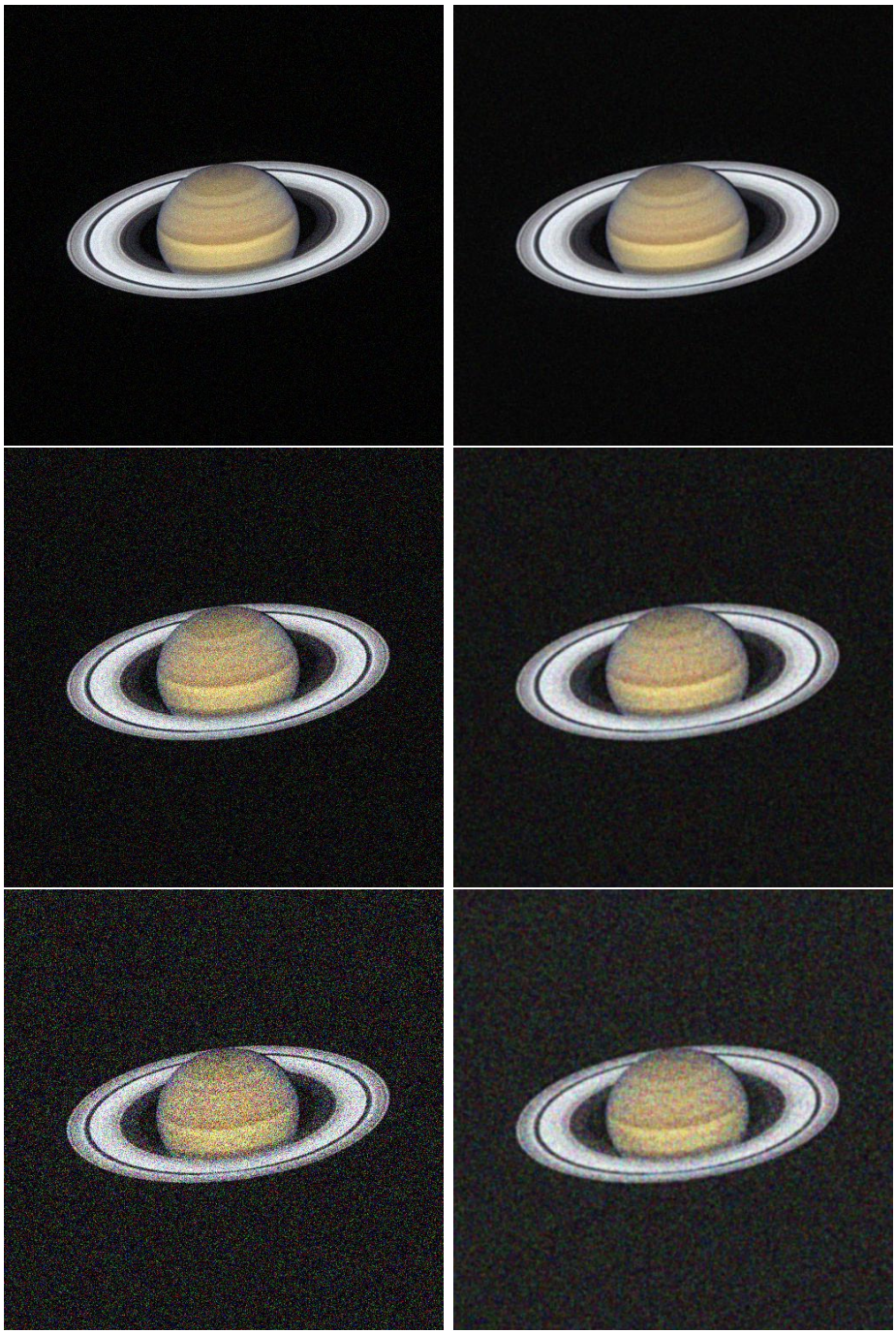
Fonte: Produção do autor.

Figura 4.6 - Resultados da imagem Westerlund com o filtro de Wiener.

Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.7 - Resultados da imagem Saturno com o filtro de Wiener.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

A Tabela 4.6 mostra as métricas dos resultados obtidos com o filtro de Wiener não-supervisionado. Nota-se que ele possui métricas melhores que o filtro Tikhonov-2 para ruídos baixos (5% e 15%), mas levemente piores para o nível de ruído 25%.

Tabela 4.6 - Métricas para o filtro *Wiener*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.092	29.0	0.88	0.15	24.76	0.73	0.204	22.11	0.61
2	0.086	32.05	0.91	0.16	26.68	0.76	0.245	22.97	0.63
3	0.059	33.73	0.77	0.16	25.11	0.46	0.262	20.84	0.34
4	0.098	31.38	0.9	0.17	26.57	0.74	0.258	22.94	0.6
5	0.079	34.92	0.94	0.183	27.57	0.78	0.317	22.82	0.62
6	0.1	31.99	0.6	0.254	23.87	0.25	0.407	19.75	0.17
7	0.104	29.72	0.86	0.16	25.92	0.7	0.225	23.0	0.58
8	0.122	28.19	0.79	0.174	25.12	0.65	0.233	22.6	0.54

4.2 Filtragem wavelet

A restauração por transformada *wavelet*, foi também utilizado rotinas do pacote *scikit-image*, que contém uma sub-rotina de restauração por transformada wavelet (*restoration.denoise_wavelet*) e os métodos de thresholding *VisuShrink* e *BayesShrink*. O *scikit-image* utiliza o pacote *PyWavelets* (LEE et al., 2019) para os algoritmos de decomposição wavelet. Todas as wavelets utilizadas neste trabalho provém desse pacote.

4.2.1 Referência

A transformada *wavelet* possui uma extensa seleção de parâmetros livres (tipo de *wavelet*, método de truncamento, níveis de decomposição, método de estimação do valor de truncamento, canal de cor, deslocamentos), que podem ser combinados de forma a maximizar a qualidade segundo as métricas apresentadas na Seção 3.3. Após experimentação com diferentes valores dos parâmetros (ver Seção 4.2.2), os parâmetros que maximizam a qualidade dos resultados são:

- *Wavelet*: Haar
- Método de truncamento: *soft*
- Níveis de decomposição: 3

- Canal de cor: YCbCr
- Deslocamentos: 7
- Estimação do valor de truncamento: BayesShrink

As Figuras 4.8 e 4.9 mostram o resultado da restauração com a transformada *wavelet* segundo os parâmetros definidos acima. Nota-se na Figura 4.8 uma boa qualidade da restauração até o nível de ruído 15% e uma perda de contraste acima desse nível. Na Figura 4.9, nota-se uma dificuldade em restaurar as áreas escuras da imagem.

A Tabela 4.7 mostra as métricas obtidas por essa técnica de restauração. Nota-se que esta combinação de parâmetros produz resultados superiores ao Filtro de Wiener, este sendo usado na literatura como referência de qualidade à qual os algoritmos devem ser comparados.

Tabela 4.7 - Métricas para a *Transformada Wavelet*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.065	32.03	0.94	0.122	26.53	0.84	0.174	23.46	0.77
2	0.066	34.32	0.95	0.125	28.8	0.87	0.195	24.96	0.8
3	0.053	34.67	0.78	0.146	25.91	0.49	0.237	21.71	0.41
4	0.071	34.14	0.94	0.127	29.09	0.87	0.195	25.36	0.82
5	0.052	38.5	0.98	0.124	30.96	0.94	0.25	24.88	0.87
6	0.086	33.3	0.61	0.234	24.58	0.27	0.38	20.36	0.2
7	0.085	31.44	0.89	0.13	27.74	0.8	0.172	25.33	0.75
8	0.115	28.75	0.8	0.151	26.36	0.69	0.184	24.64	0.66

4.2.2 Comparações com a referência

Nesta subseção, serão apresentadas as influências de cada parâmetro. As tabelas a seguir representam a diferença das métricas da imagem referência da subseção anterior, menos as métricas da transformada *wavelet* com os parâmetros modificados em cada tabela. Ou seja, o NRMSE negativo e as PSNR e SSIM positivas representam o quanto tal parâmetro contribui positivamente na restauração. Os demais parâmetros são mantidos como na transformada de referência.

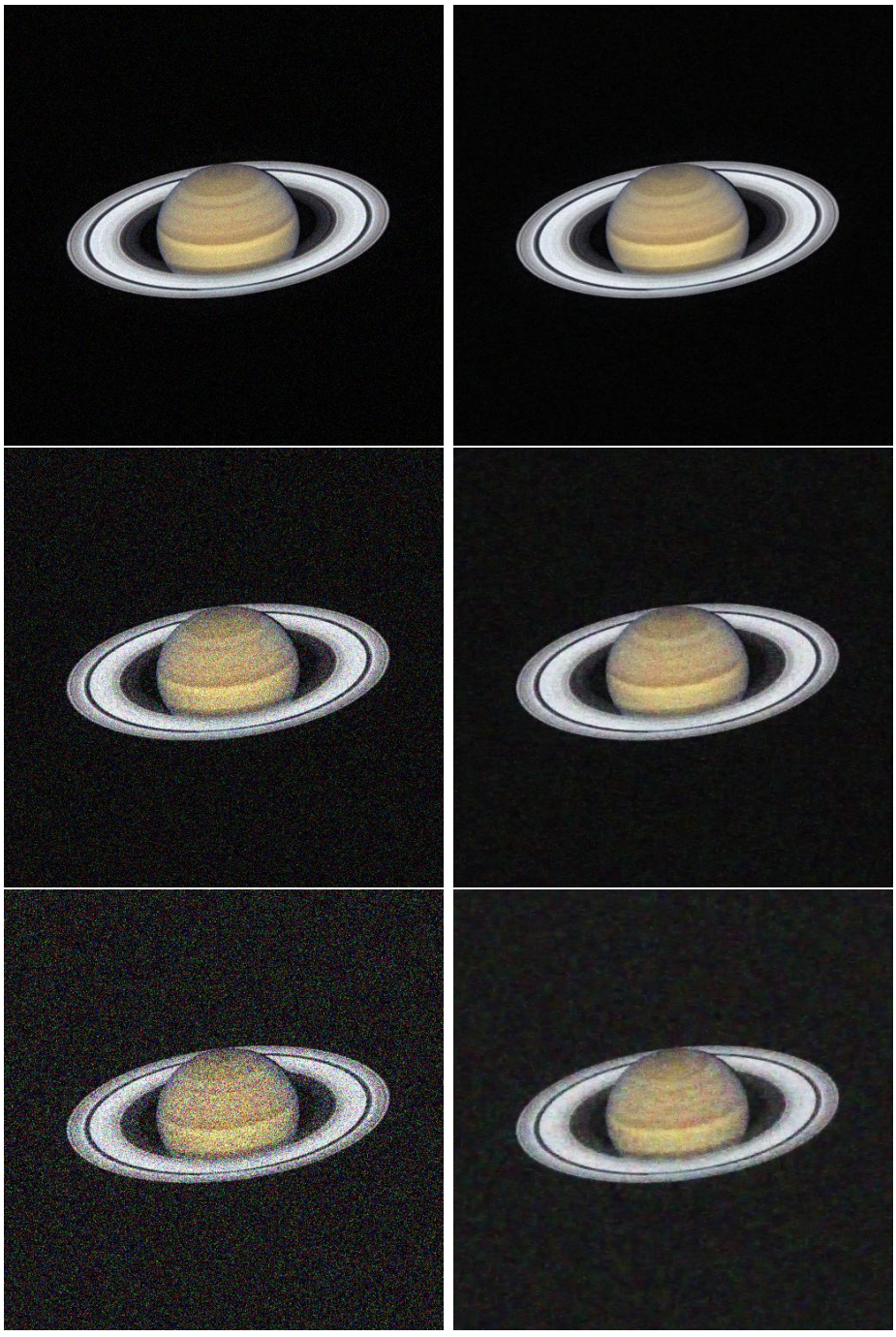
A Tabela 4.8 mostra a influência da conversão do canal de cor para YCbCr. Nota-se que em apenas um caso (imagem 1 - Westerlund - 5%) existe uma degradação nessa

Figura 4.8 - Resultados da imagem Westerlund com a transformada Wavelet Haar, parâmetros de referência.

Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.9 - Resultados da imagem Saturno com a transformada Wavelet Haar, parâmetros de referência.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

conversão. Nos demais casos, a contribuição fica entre 0.6 e 1dB na PSNR para o ruído de 5% e a contribuição diminui conforme o ruído aumenta.

Tabela 4.8 - Diferença entre a Transformada Wavelet com e sem a conversão para YCbCr, mantendo $s = 7$.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.001	-0.14	0.01	-0.008	0.54	0.03	-0.006	0.29	0.02
2	-0.005	0.6	0.01	-0.006	0.39	0.02	-0.004	0.18	0.01
3	-0.005	0.76	0.01	-0.004	0.23	0.01	-0.003	0.12	0.01
4	-0.005	0.6	0.01	-0.007	0.45	0.01	-0.004	0.19	0.01
5	-0.006	0.95	0.0	-0.012	0.79	0.01	-0.011	0.39	0.03
6	-0.01	0.9	0.01	-0.01	0.35	0.01	-0.009	0.2	0.01
7	-0.005	0.47	0.01	-0.006	0.38	0.01	-0.005	0.25	0.01
8	-0.013	1.01	0.05	-0.006	0.36	0.02	-0.003	0.16	0.01

A Tabela 4.9 mostra a contribuição do *cycle spin* com 7 deslocamentos. Nota-se que este possui uma influência mais considerável que a conversão do canal de cor, chegando a 2.44dB na PSNR da imagem 5 com ruído 5%. A influência diminui conforme o ruído aumenta, mas ainda assim é mais expressiva que a conversão do canal de cor para a maioria das imagens.

Tabela 4.9 - Diferença entre a Transformada Wavelet com e sem $s = 7$, mantendo a conversão YCbCr.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	-0.01	1.17	0.02	-0.018	1.14	0.06	-0.014	0.64	0.05
2	-0.013	1.41	0.02	-0.015	0.96	0.04	-0.01	0.44	0.03
3	-0.006	0.88	0.01	-0.012	0.64	0.04	-0.013	0.45	0.05
4	-0.009	0.94	0.02	-0.009	0.57	0.02	-0.005	0.23	0.02
5	-0.017	**2.44**	0.02	-0.019	1.24	0.03	-0.013	0.43	0.04
6	-0.004	0.4	0.01	-0.012	0.41	0.01	-0.017	0.36	0.02
7	-0.008	0.73	0.02	-0.009	0.57	0.03	-0.006	0.3	0.02
8	-0.007	0.58	0.02	-0.007	0.37	0.02	-0.004	0.19	0.01

A Tabela 4.10 mostra a diferença entre a trasformada Haar e a transformada Biortogonal 6.8 para a restauração das imagens. Nota-se que a db1 melhora os resultados

em todos os casos, em especial na imagem 5 onde existe uma perda d 6.89dB e 37% de similaridade na imagem 5 com ruído 15%. Uma explicação para este acontecimento é que pelas transformadas diretas e inversa biortogonais não serem ortogonais entre si, o ruído modelado pode ser diferente do tipo gaussiano.

Tabela 4.10 - Diferença entre a Transformada Wavelet a wavelet db1 e Biortogonal 6.8.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	-0.009	1.04	0.03	-0.056	3.17	0.22	-0.096	3.75	0.33
2	-0.017	1.87	0.04	-0.092	4.66	0.25	-0.152	4.92	0.39
3	-0.012	1.68	0.03	-0.051	2.49	0.12	-0.086	2.6	0.16
4	-0.023	2.3	0.06	-0.11	5.38	0.31	-0.177	5.62	0.47
5	-0.048	5.64	0.08	**-0.147**	**6.89**	**0.37**	-0.197	5.12	0.51
6	-0.008	0.69	0.01	-0.043	1.39	0.03	-0.08	1.6	0.05
7	-0.012	1.11	0.03	-0.086	4.34	0.24	-0.158	5.64	0.39
8	-0.008	0.64	0.02	-0.078	3.61	0.16	-0.151	5.18	0.29

A Tabela 4.11 mostra a diferença entre os métodos de estimação do valor de truncamento BayesShrink e VisuShrink. Nota-se que o método VisuShrink apresenta uma leve melhora nas imagens 3 e 6 (Júpiter e Saturno), que são imagens que possuem baixa variância e muitos pontos de brilho zero. No entanto, o caso geral é muito favorável ao BayesShrink, tendo uma melhoria de quase 5dB PSNR nas duas primeiras imagens com ruído 5%.

Tabela 4.11 - Diferença entre o método de estimação do valor de truncamento BayesShrink e VisuShrink.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	-0.055	4.93	0.1	-0.044	2.64	0.08	-0.026	1.19	0.05
2	-0.052	4.67	0.09	-0.027	1.66	0.06	-0.012	0.51	0.03
3	-0.005	0.79	0.01	0.011	-0.63	-0.05	0.011	-0.39	-0.07
4	-0.034	3.21	0.05	-0.014	0.89	0.02	-0.005	0.2	0.01
5	-0.014	1.98	0.0	-0.014	0.95	0.0	-0.007	0.26	0.0
6	0.006	-0.57	-0.02	0.01	-0.34	-0.03	0.008	-0.17	-0.04
7	-0.038	3.08	0.1	-0.013	0.82	0.03	-0.005	0.26	0.01
8	-0.044	3.09	0.16	-0.011	0.61	0.03	-0.003	0.13	0.01

A Tabela 4.12 mostra a influência do método de truncamento, *hard* ou *soft*. Como os métodos de restauração são baseados em *shrinkage* (encolhimento), são utilizados o método *soft*. Pela tabela percebe-se que o método *hard* degrada todas as imagens em quase todos os casos.

Outros métodos de estimação do valor de truncamento, à parte do *BayesShrink*, podem ser melhor adaptados ao método de truncamento *hard*. Mas demais métodos não foram testados.

Tabela 4.12 - Diferença entre o método de threshold *soft* e *hard*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	-0.015	1.61	0.05	-0.015	0.97	0.08	-0.002	0.08	0.04
2	-0.014	1.52	0.04	-0.002	0.12	0.01	-0.001	0.05	0.01
3	-0.032	3.85	0.07	-0.091	4.01	0.19	-0.111	3.25	0.21
4	-0.015	1.58	0.04	-0.001	0.05	0.02	0.001	-0.03	0.01
5	-0.014	2.0	0.02	-0.008	0.56	0.03	-0.015	0.49	0.06
6	-0.034	2.7	0.05	-0.102	2.98	0.08	-0.147	2.72	0.08
7	-0.008	0.72	0.02	0.002	-0.13	-0.0	0.002	-0.11	0.0
8	-0.008	0.65	0.0	0.0	-0.02	-0.02	0.002	-0.11	-0.01

A Tabela 4.13 mostra a contribuição da *wavelet* Symlet 4, que é ortogonal e quase simétrica. Ela apresenta uma leve melhora nos resultados das imagens 3 e 6 (Júpiter e Saturno), e e uma degradação mínima nas demais imagens. A Symlet 4, então, é uma opção muito considerável à db1 para restauração.

Tabela 4.13 - Diferença entre a Wavelet sym4 e db1.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	-0.001	0.08	0.0	-0.003	0.21	0.01	-0.001	0.03	0.01
2	0.0	-0.0	-0.0	-0.001	0.08	0.0	-0.003	0.12	0.01
3	0.003	-0.49	-0.01	0.003	-0.16	-0.01	0.002	-0.06	-0.01
4	-0.002	0.27	0.0	-0.003	0.23	0.01	-0.005	0.21	0.01
5	-0.001	0.21	0.0	-0.005	0.34	0.01	-0.004	0.13	0.01
6	0.004	-0.43	-0.01	0.004	-0.15	-0.01	0.003	-0.07	-0.01
7	-0.002	0.24	0.01	-0.003	0.18	0.01	-0.004	0.19	0.01
8	-0.002	0.17	0.01	-0.003	0.18	0.01	-0.005	0.22	0.01

4.3 Rede convolucional

Para a rede neural convolucional, foi utilizada uma modificação local da implementação disponível em (UCHIDA, 2018) e os pesos foram disponibilizados junto à implementação. A modificação consiste em adaptar a fonte do ruído e o formato de saída.

Considerando o modelo da rede neural já treinado com os parâmetros descritos na Seção 2.3, a restauração é realizada fazendo a inferência a partir da configuração final da rede. As Figuras 4.10 e 4.11 mostram o resultado visual desta rede neural convolucional utilizando imagens "limpas" (sem ruído) como alvo de treinamento. Nota-se que, visualmente, o resultado é muito bom, mesmo para o maior nível de ruído. Pode-se dizer que no caso do nível 25% o ruído gausiano é removido completamente ao custo de um leve borramento. Este foi o único método que conseguiu remover o ruído de fundo escuro da imagem Saturno.

A Tabela 4.14 mostra as métricas dos resultados da rede neural Noise2Clean. Esta rede supera os resultados obtidos pela Transformada Wavelet, chegando a uma média de 90% de similaridade em todas as imagens com ruído 5%, e ultrapassando os 40dB da PSNR nas imagens 3 e 6.

Tabela 4.14 - Métricas para a rede convolucional *Noise2Clean*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.065	31.98	0.94	0.102	28.09	0.87	0.129	26.1	0.82
2	0.06	35.25	0.96	0.102	30.57	0.9	0.129	28.53	0.85
3	0.027	40.53	0.97	0.05	35.26	0.91	0.076	31.56	0.76
4	0.065	34.94	0.95	0.098	31.35	0.9	0.12	29.56	0.87
5	0.046	39.53	0.98	0.069	36.09	0.97	0.095	33.27	0.95
6	0.026	43.81	0.97	0.05	37.92	0.89	0.093	32.59	0.66
7	0.078	32.21	0.91	0.113	28.94	0.83	0.133	27.57	0.79
8	0.095	30.38	0.87	0.137	27.23	0.73	0.153	26.28	0.69

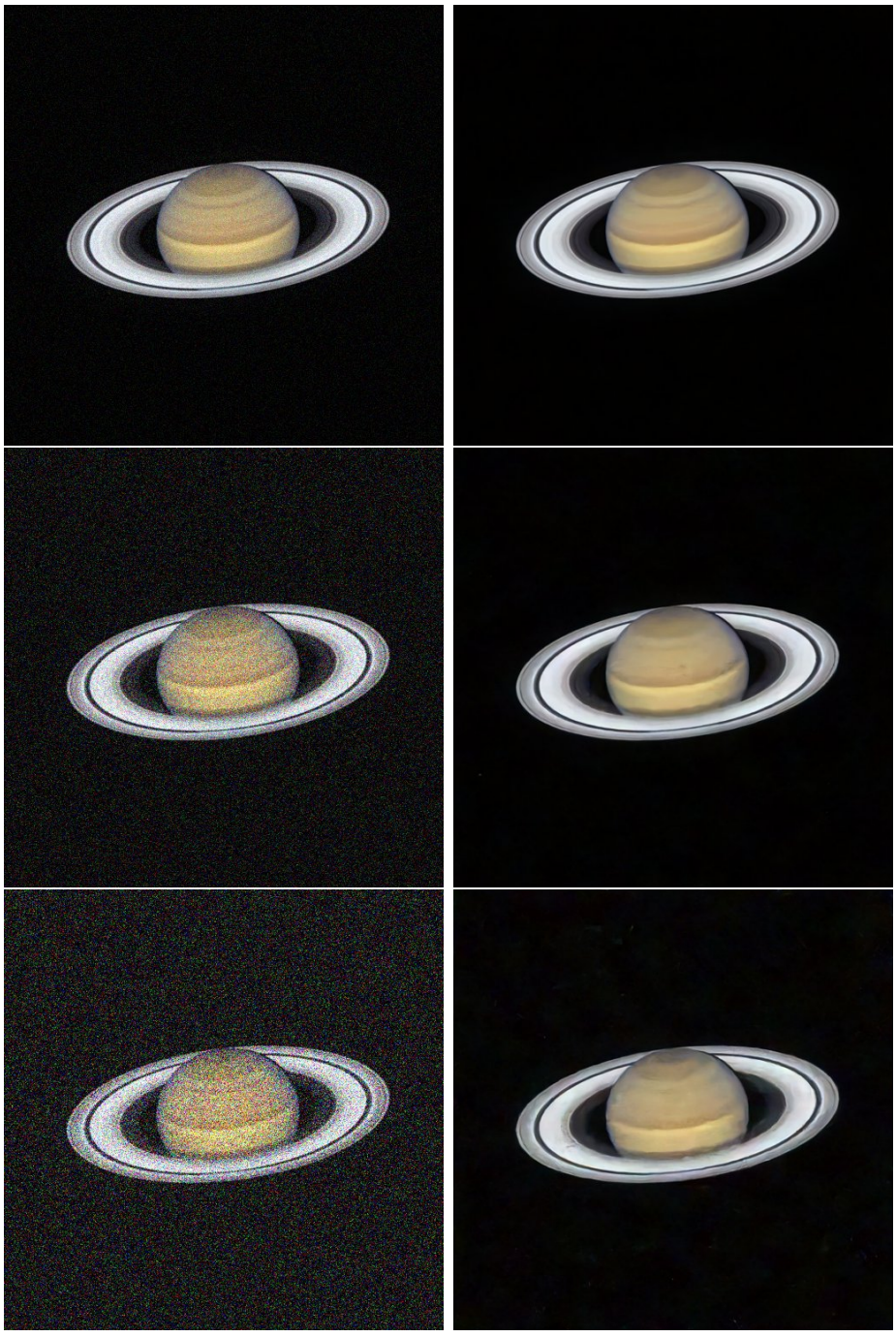
A Tabela apresenta as métricas dos resultados da rede convolucional treinada com dados ruidosos (Noise2Noise). Nota-se uma perda de qualidade considerável para o nível de ruído 5%, mas a qualidade permanece constante entre todos os níveis de ruído, sendo o resultado com ruído 25% comparável à rede Noise2Clean.

Figura 4.10 - Resultados da imagem Westerlund com a rede convolucional SRResNet Noise2Clean.

Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.11 - Resultados da imagem Saturno com a rede convolucional SRResNet Noise2Clean.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Tabela 4.15 - Métricas para a rede convolucional *Noise2Noise*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.114	27.17	0.84	0.113	27.19	0.84	0.113	27.19	0.84
2	0.116	29.45	0.87	0.116	29.5	0.87	0.117	29.42	0.87
3	0.1	29.16	0.6	0.101	29.15	0.6	0.101	29.09	0.6
4	0.117	29.82	0.86	0.116	29.84	0.86	0.116	29.87	0.86
5	0.101	32.79	0.93	0.1	32.8	0.93	0.1	32.79	0.93
6	0.151	28.39	0.4	0.151	28.36	0.4	0.15	28.43	0.4
7	0.125	28.1	0.8	0.125	28.07	0.8	0.125	28.07	0.8
8	0.147	26.63	0.72	0.147	26.62	0.72	0.147	26.63	0.72

4.4 Filtro neural multiescala

A implementação do filtro neural utilizada foi providenciada pela autora de (CASTRO et al., 2008) em MATLAB e executada no Octave. A autora distribuiu os pesos e o código de inferência. Após a inferência, foi gerada a Tabela 4.16.

Tabela 4.16 - Métricas para a rede MLP *filtro multiescala*.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
1	0.147	24.86	0.56	0.159	24.18	0.56	0.221	21.32	0.56
2	0.151	27.07	0.55	0.169	26.13	0.55	0.263	22.28	0.55
3	0.107	28.54	0.44	0.131	26.73	0.41	0.207	22.77	0.36
4	0.155	27.28	0.5	0.175	26.26	0.51	0.275	22.32	0.51
5	0.161	28.63	0.52	0.187	27.31	0.5	0.315	22.78	0.48
6	0.175	26.97	0.23	0.204	25.64	0.21	0.295	22.44	0.17
7	0.152	26.32	0.5	0.168	25.44	0.51	0.253	21.9	0.51
8	0.166	25.49	0.41	0.18	24.76	0.41	0.257	21.71	0.41

Este filtro é comparável à regularização Tikhonov-2. Porém, nota-se que o filtro multiescala é mais tolerante ao nível de ruído, tendo pouca variação de 5% a 25%. Os resultados também diferem pouco entre as imagens, a rede restaura todas as imagens com aproximadamente a mesma qualidade. Os resultados evidenciam uma propriedade que foi pensada no projeto desta rede: a restauração é realizada independente do nível de ruído (CASTRO et al., 2008).

Observa-se que a combinação de hiperparâmetros com os quais a rede foi treinada

não é necessariamente a combinação ótima, portanto, esta rede ainda pode ser aprimorada.

As Figuras 4.12 e 4.13 mostram as imagens degradas e restauradas. Este filtro apresenta uma diminuição do ruído sem introduzir novos artefatos (como borramento) na imagem.

4.5 Comparação final

Considerando os resultados obtidos das imagens 1 a 8 em cada método, as Tabelas 4.17 e 4.18 mostram as médias e desvios padrão de restauração de todas as imagens. A rede convolucional Noise2Clean obteve a melhor média de resultados em todos os casos. A filtragem *wavelet* é o segundo melhor método. A Tabela 4.18 mostra o desvio padrão entre a restauração de todas as imagens. Para o caso do valor de desvio padrão, as duas técnicas baseadas em redes neurais obtiveram o menor desvio, ou seja, as médias obtidas com estes métodos estão mais agrupadas em torno da média, o que pode significar melhor representatividade da média. O filtro multiescala apresentou menor desvio nos níveis de ruído mais baixos, enquanto a rede convolucional apresentou menor desvio nos níveis de ruído mais altos.

Tabela 4.17 - Médias entre os resultados de todas as imagens juntas.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
T2	0.129	28.565	0.788	0.167	25.925	0.674	0.247	22.9	0.596
T1	0.134	28.188	0.781	0.175	25.809	0.67	0.248	22.902	0.59
T0	0.836	12.073	0.379	0.838	12.055	0.348	0.839	12.044	0.311
TV	0.094	31.378	0.825	0.154	26.952	0.699	0.312	20.775	0.418
FW	0.092	31.372	0.831	0.176	25.7	0.634	0.269	22.129	0.498
W+	0.074	33.394	0.861	0.145	27.496	0.721	0.223	23.838	0.66
RC	**0.058**	**36.079**	**0.944**	**0.09**	**31.931**	**0.875**	**0.116**	**29.432**	**0.799**
RM	0.152	26.895	0.464	0.172	25.806	0.458	0.261	22.19	0.444

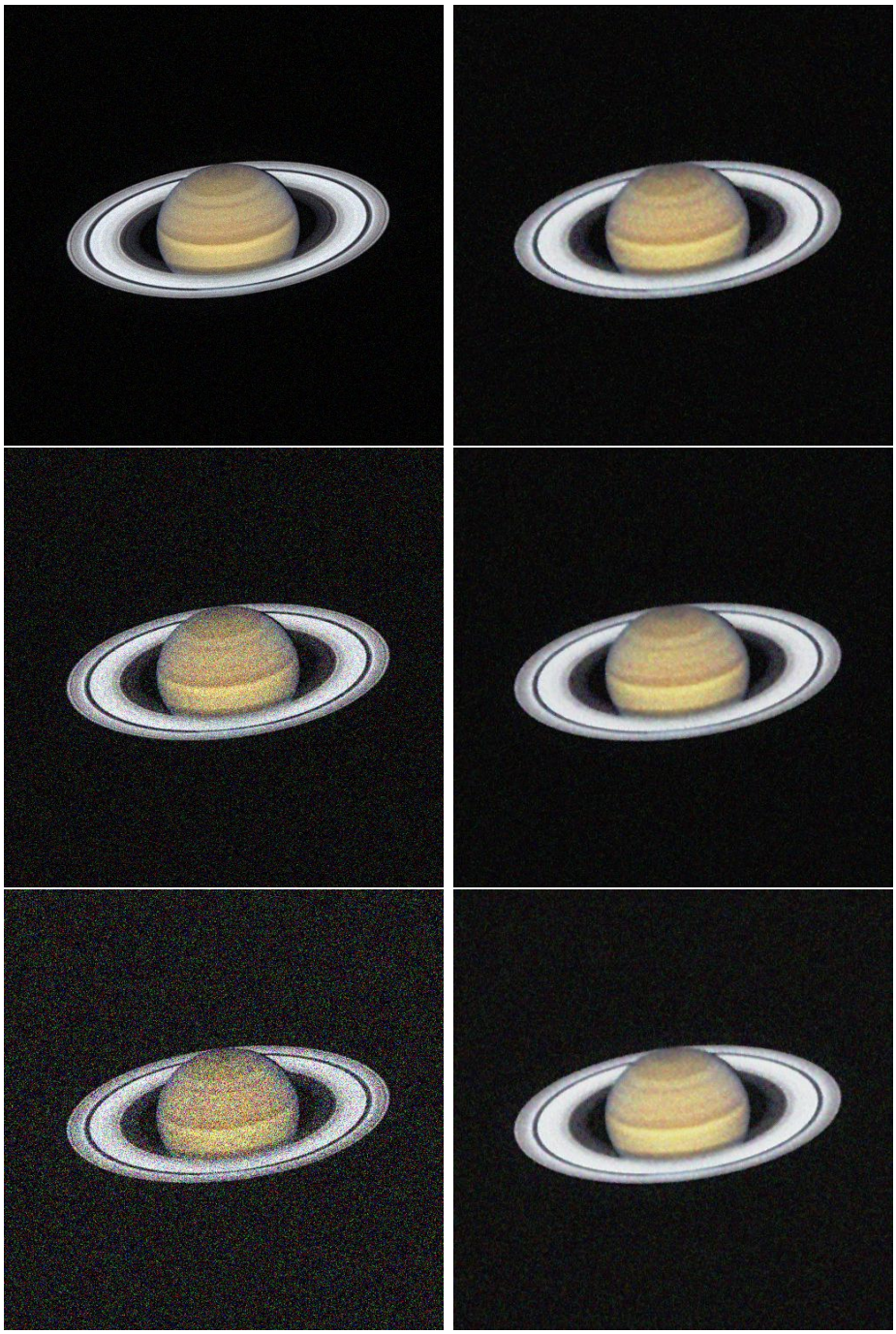
Métodos: T2 = Tikhonov-2, T1=Tikhonov-1, T0=Tikhonov-0, TV = Regularização por Variação Total, FW = Filtro de Wiener, W+ = Filtragem Wavelet, RC = Rede convolucional Noise2Clean, RM = Filtro neural multiescala.

Figura 4.12 - Resultados da imagem Westerlund com a rede MLP Filtro Neural.

Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Figura 4.13 - Resultados da imagem Saturno com a rede MLP Filtro Neural.



Esquerda: imagem degradada com ruídos gaussiano 5%, 15% e 25%, direita: restauração correspondente.

Fonte: Produção do autor.

Tabela 4.18 - Desvio padrão entre os resultados de todas as imagens juntas. T2 = Tikhonov-2, T1=Tikhonov-1, T0=Tikhonov-0, TV = Regulariação por Variação Total, FW = Filtro de Wiener, W+ = Filtragem Wavelet, RC = Rede convolucional Noise2Clean, RM = Filtro neural multiescala.

σ	5%			15%			25%		
	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM	NRMSE	PSNR	SSIM
T2	0.028	2.607	0.107	0.041	1.693	0.191	0.064	1.485	0.187
T1	0.028	2.434	0.107	0.039	1.749	0.195	0.067	1.578	0.191
T0	**0.001**	1.38	0.212	**0.002**	1.383	0.166	0.004	1.386	0.106
TV	0.023	2.892	0.105	0.034	1.565	0.199	0.053	0.648	0.127
FW	0.017	2.164	0.103	0.031	1.128	0.173	0.061	1.126	0.174
W+	0.019	2.654	0.117	0.035	1.922	0.215	0.065	1.743	0.219
RC	0.022	4.44	0.035	0.029	3.746	**0.066**	**0.024**	2.612	**0.089**
RM	0.019	**1.242**	**0.1**	0.02	**0.957**	0.107	0.033	**0.48**	0.121

Métodos: T2 = Tikhonov-2, T1=Tikhonov-1, T0=Tikhonov-0, TV = Regularização por Variação Total, FW = Filtro de Wiener, W+ = Filtragem Wavelet, RC = Rede convolucional Noise2Clean, RM = Filtro neural multiescala.

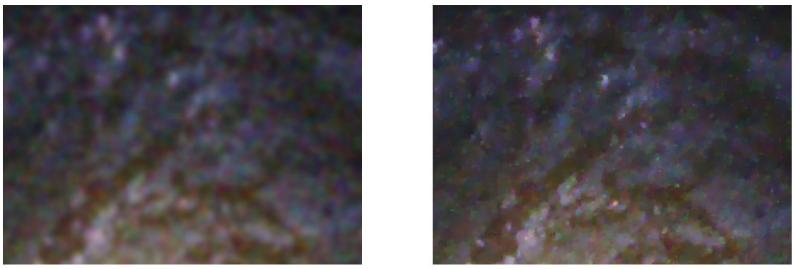
Após a análise de generalização dos métodos, é realizada uma comparação visual nas regiões de bordas, para verificar o comportamento da restauração de alguns métodos em um nível mais detalhado da imagem.

A Figura 4.14 compara uma região de borda entre os métodos Tikhonov-2 e TVR para a imagem 2. Nota-se que devido à menor suavização no TVR, detalhes como estrelas podem ser restaurados com maior precisão. A Figura 4.15 compara a contribuição do *cycle spin* em uma região da imagem 5. A técnica diminui o efeito pixelado na restauração. Já o VisuShrink, como visto na Figura 4.16, tende a causar um efeito de desfoque. Neste caso, como a subbanda diagonal possui ruído nas duas direções, o valor de truncamento é alto em relação às subbandas direcionais e isto acaba eliminando muito sinal e causando o efeito observado.
Alguns métodos apresentam melhor desempenho quando altos níveis de ruído estão presentes, como Tikhonov-2 e VisuShrink – como visto nas Tabelas 4.2, 4.1.3 e 4.11.

Por fim, os métodos são comparados por pelo tempo de CPU na Tabela 4.19. A tabela mostra o tempo de execução apenas da imagem 1 com ruído 15% por todos os métodos na mesma máquina. O tempo é medido com o comando *time* disponível nos sistemas baseados em Unix e possui as seguintes medidas (M):

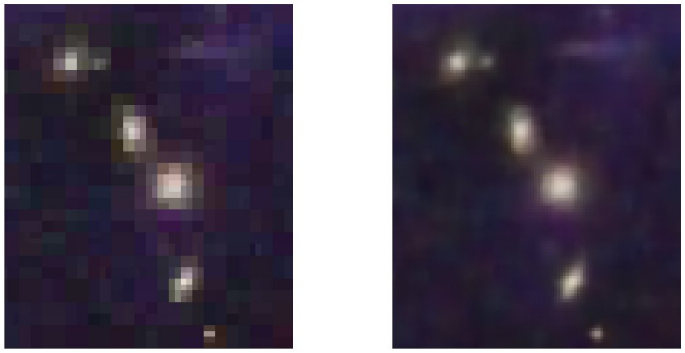
- *Real*: Tempo do início da execução ao final do programa.

Figura 4.14 - Comparação entre os métodos Tikhonov-2 e TVR.



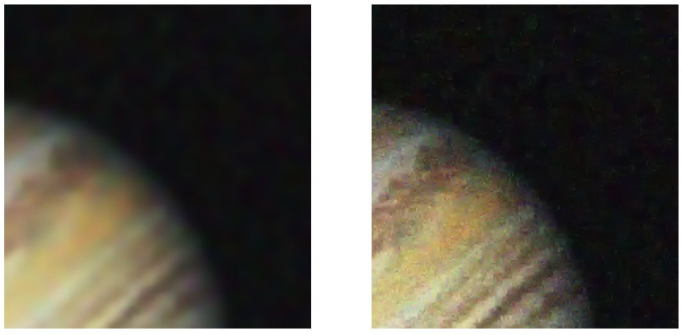
Esquerda: Regularização Tikhonov-2. Direita: TVR. Ruído: 15%.
Fonte: Produção do autor.

Figura 4.15 - Resultados com e sem deslocamentos na transformada *Wavelet*.



Esquerda: Sem deslocamentos. Direita: Com deslocamentos. Ruído: 15%
Fonte: Produção do autor.

Figura 4.16 - Comparação entre os resultados com VisuShrink e BayesShrink na transformada *wavelet*.



Esquerda: VisuShrink. Direita: BayesShrink. Ruído: 15%.
Fonte: Produção do autor.

- *User*: Tempo gasto na execução do processo, sem chamadas de sistema.
- *Sys*: Tempo de execução gasto em chamadas de sistema.

Tabela 4.19 - Comparação de tempo de CPU entre os métodos utilizados, onde "M" é o tipo de medida.

M	Tikh1	Tikh2	Entropia	TVR	UW	Wavelet	RC	FNME
real	1m1s	1m1s	85m07s	**6s**	14s	10,50s	36,0s	4m45s
user	1m0s	1m0s	131m22s	**6s**	14s	27,10s	1m33,0s	4m45s
sys	2s	2s	95m56s	**1s**	1s	1,85s	7,4s	4m45s

Com exceção do método da regularização por entropia, o desempenho foi medido considerando a imagem em tamanho 512×512 com três canais de cor. No caso da entropia a imagem teve de ser redimensionada para 64×64 e convertida em uma imagem em tons de cinza. Apesar disso, o método ainda é muito mais lento que os demais e possui um tempo considerável gasto em chamadas de sistema.

O algoritmo de menor tempo de CPU computacional foi o Chambolle (TVM), seguido do filtro de Wiener.

O desempenho da Transformada Wavelet foi medido considerando a técnica *cycle spin* com 7 deslocamentos. Apesar do código também ser otimizado, como os deslocamentos são feitos em todas as direções (4 direções) por 7 vezes, a restauração é executada 28 vezes, o que acarreta uma diferença maior em relação aos demais métodos.

Para a avaliação de desempenho das redes neurais, foi considerada apenas a inferência, sem a parte de treinamento. Lehtinen et al. (2018) menciona que a rede convolucional levou 13 horas para ser treinada utilizando uma placa GPU NVIDIA Tesla P100. Já a rede FNME pode ser treinada em uma máquina comum (CPU *multi-core multi-thread*).

5 CONCLUSÕES E TRABALHOS FUTUROS

O método Tikhonov é computacionalmente eficiente e apresenta um bom resultado para a regularização de 2ª ordem. Não há diferença de desempenho computacional entre as ordens na solução em domínio da frequência. O algoritmo também é de fácil implementação. O método de Tikhonov-2 apresenta melhores resultados para ruídos mais altos.

Diferente do método de Tikhonov de passo único, onde o problema direto linear permite uma solução explícita para o funcional de regularização, a não-linearidade do operador de entropia não permite soluções deste tipo. A aplicação de métodos de otimização iterativos requer um alto esforço computacional, devido a alta dimensão do espaço de busca – para uma imagem com N^2 *pixels*, mesmo em baixa resolução de 64×64 *pixels* implica num espaço de busca de dimensão de **4096**, para imagens de resolução mais alta, por exemplo com $O(10^3)$ *pixels*, o número de pixels desconhecidos torna-se da ordem de 10^6. Mesmo reduzindo a dimensão da imagem e canais de cor, este método se mostrou desproporcionalmente lento em comparação aos demais.

O método TVR implementado pelo algoritmo Chambolle se mostrou comparável aos métodos Tikhonov-2 e Wiener. Comparando com o filtro de Wiener, este método apresenta resultados melhores em todos os casos apenas com nível de ruído 15%, possui métricas inferiores nos demais níveis de ruído e apresentou o melhor desempenho computacional entre todas as metodologias de restauração testadas – estimado pelo comando *time*.

O filtro de Wiener é um dos métodos mais conhecidos de restauração de imagens e utilizado como referência de qualidade em comparação com outros métodos. Entretanto, os resultados com filtragem *wavelet* e pela rede convolucional obtiveram melhores métricas de restauração. Contudo, o filtro de Wiener tem um custo computacional menor que os métodos de *wavelet* e rede neural – ver Tabela 4.19.

A combinação de parâmetros utilizada na restauração por transformada *wavelet* de referência surpreendeu positivamente. As métricas foram comparáveis à rede *Noise2Clean*, que foi o método com melhores métricas de restauração entre os testados. Entretanto, a técnica de *wavelet* tem desempenho pior nas imagens 3 e 6, que possuem uma grande região de área escura que também foi contaminada por ruído.

Os métodos baseados em regularização – exceto entropia, para os casos analisados

– são os que menos apresentam limitações. São computacionalmente eficientes e dependem apenas de um parâmetro livre (α) e podem restaurar mais tipos de ruído. No entanto, os métodos baseados em filtragem *wavelet* e redes neurais conseguem gerar resultados melhores.

Uma limitação da transformada *wavelet*, dependendo da família de funções escolhida, é a aplicação apenas em ruídos do tipo gaussiano, visto que o valor de truncamento é proporcional ao desvio padrão do ruído. Os métodos baseados em regularização são capazes de restaurar também degradação do tipo borramento, e a rede neural pode ser treinada para restaurar qualquer tipo de degradação. Um outro aspecto referente a métodos de filtragem por meio de transformadas *wavelet*, que as diferencia dos demais métodos, é a grande quantidade de opções para a escolha das família de funções wavelet, do número de níveis recomendável para as análises, métodos adaptativos de truncamento e de estimação do valor de truncamento, entre outros parâmetros específicos de cada formulação. Neste trabalho técnicas consagradas foram testadas, no entanto este segue sendo um tópico com grande potencial para investigações futuras.

O maior esforço computacional para as redes neurais reside no processo de treinamento. Dependendo da quantidade e dimensão dos dados, o processo computacional para a etapa de treinamento requer uso de processamento paralelo (vários processadores em rede), como a solução tipo *cluster*. No entanto, o resultado entregue pela rede é de alta qualidade com esforço computacional competitivo, em que a inferência pode ser feita em computadores domésticos ou até em outros tipos de *hardware*, como co-processadores do tipo GPU e FPGA.

5.1 Trabalhos futuros

O tema de restauração de imagem é tópico de ampla pesquisa, para o qual várias metodologias são propostas e avaliadas sistematicamente devido à importância das aplicações. Algumas das técnicas de restauração foram descritas e aplicadas nesta dissertação de mestrado. Contudo, há outras técnicas que têm produzido bons resultados em restauração de imagens, como a técnica de morfologia matemática, com aplicações em imagens de satélites (BANON; CANDELAS, 1993) e em ciência dos materais (CIDADE et al., 2012), e o método de regularização baseada na distância de Bregman (CIDADE et al., 2012) – ou *q-discrepância*. Além destas, uma outra alternativa interessante a ser considerada é a regularização via entropia não

extensiva[1] (CAMPOS VELHO et al., 2006)[2]. Destaca-se também a possibilidade de desenvolvimento de versões em arquiteturas de processamento paralelo (STUTZ; SILVA NETO, 2011) para as técnicas aqui trabalhadas.

O método de variação total (TVR) pode ser formulado como uma variante de mínimos quadrados (ASTER et al., 2018, Sec 7.3), ou como pertencente à classe dos métodos de restauração de imagens por solução de equações diferenciais parciais (EDP) (RUDIN et al., 1992; CHAMBOLLE, 2004). Contudo, há uma enorme literatura para restauração e processamento de imagens baseada em solução de EDP, em particular, os métodos variacionais (POSIRCA et al., 2011; CHEN, 2013).

A transformada discreta *Wavelet* associada a métodos de filtragem foi testada com um conjunto limitado de métodos de estimação do valor de truncamento (*VisuShrink* e *BayesShrink*). Outros métodos adaptativos que se mostraram eficientes para diferentes aplicações merecem ser considerados na restauração de imagens astronômicas, como por exemplo *NeighShrink* (CHEN et al., 2005).

Além das técnicas citadas para determinar o truncamento da expansão *wavelet*, é de interesse aplicar/testar o método da discrepância de Morozov (MOROZOV; STESSIN, 1993; MUNIZ et al., 1999), que pode ser generalizado para ruídos não gaussianos (SHIGUEMORI et al., 2004).

Uma outra possiblididade de estudo futuro é a utilização da transformada DTCWT (*Dual-Tree Complex Wavelet Transform*), uma vez que nesta dissertação apenas foram consideradas implementações para transformadas *wavelet* do pacote de software em (LEE et al., 2019), no qual não consta a formulação DTCWT.

A rede neural filtro multiescala foi implementada em MATLAB sem o uso de ferramentas de software auxiliares. Trabalhos futuros para esta rede podem ter várias abordagens. Uma delas é aplicar uma formulação automática para identificar o conjunto ótimo de hiper-parâmetros para a rede na aplicação de interesse (ANOCHI et al., 2019; AKIBA et al., 2019). Outra estratégia é organizar o conjunto de treinamento para aplicar em *frameworks* de aprendizado profundo (KRIZHEVSKY et al., 2012; DAHL et al., 2014).

Todas as imagens neste trabalho foram padronizadas no tamanho 512×512, uma resolução média, para reduzir o custo computacional. Supõe-se que alguns métodos

[1] A técnica de regularização pelo princípio da máxima entropia é bem estabelecida na literatura para restauração de imagens, mas a impletação não logrou êxito e, por óbvio, precisa ser revista.

[2] Há similaridade entre a métrica de distância de Bregman e entropia não extensiva.

possam apresentar melhores ou piores resultados dependendo do tamanho da imagem. Por exemplo, a estimação do valor de truncamento pelo "VisuShrink" é sensível ao tamanho da imagem, portanto este método tende a suavizar mais as imagens maiores. A comparação da mesma técnica em imagens de tamanhos diferentes não foi realizada neste trabalho. Desta forma, realizar experimentos com imagens de alta resolução, como costumam ser as imagens astronômicas, é um item a ser avaliado para confirmar os resultados de desempenho obtidos nesta dissertação.

REFERÊNCIAS BIBLIOGRÁFICAS

ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANé, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIéGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; ZHENG, X. **TensorFlow: large-scale machine learning on heterogeneous systems**. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. 59

ABDELNOUR, A. F.; SELESNICK, I. W. Symmetric nearly orthogonal and orthogonal nearly symmetric wavelets. **Arabian Journal for Science and Engineering**, v. 29, n. 2, p. 3–16, 2004. 21

AKIBA, T.; SANO, S.; YANASE, T.; OHTA, T.; KOYAMA, M. **Optuna: a next-generation hyperparameter optimization framework**. 2019. Disponível em: <https://arxiv.org/abs/1907.10902>. 91

ANOCHI, J. A.; CAMPOS VELHO, H. F.; Hernández Torres, R. Two geoscience applications by optimal neural network architecture. **Pure and Applied Geophysics**, v. 176, 2019. Disponível em: <https://doi.org/10.1007/s00024-019-02386-y>. 91

ASADZADEH, M.; HASHEMI, E.; KOZAKEVICIUS, A. On efficiency of combined daubechies wavelets and statistical parameters applied in mammography. **Applied Computational Mathematics**, v. 12, n. 3, 2012. 3

ASTER, R. C.; BORCHERS, B.; THURBER, C. H. **Parameter estimation and inverse problems**. [S.l.]: Elsevier, 2018. 14, 91

BANON, G. J. F.; CANDELAS, A. L. B. Restauração de imagens NOAA por morfologia matemática. In: SIMPÓSIO BRASILEIRO DE COMPUTAÇãO GRÁFICA E PROCESSAMENTO DE IMAGENS, Recife, Brazil. **Proceedings...** [S.l.], 1993. p. 139–146. 90

BARRI, A.; DOOMS, A.; SCHELKENS, P. The near shift-invariance of the dual-tree complex wavelet transform revisited. **Journal of Mathematical**

Analysis and Applications, v. 389, n. 2, p. 1303–1314, May 2012. ISSN 0022-247X. Disponível em: <http://dx.doi.org/10.1016/j.jmaa.2012.01.010>. 32

BERTERO, M. **Introduction to inverse problems in imaging**. [S.l.]: CRC Press, 1998. 5-9 p. 8

BOUMAN, K. L.; JOHNSON, M. D.; ZORAN, D.; FISH, V. L.; DOELEMAN, S. S.; FREEMAN, W. T. Computational imaging for VLBI image reconstruction. In: THE IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. **Proceedings...** 2016. p. 913. Disponível em: <https://arxiv.org/abs/1512.01413>. 1

CAMPOS VELHO, H. F. **Problemas inversos em pesquisa espacial**. [S.l.]: SBMAC-Notas de Matemática Aplicada, 2019. 7

CAMPOS VELHO, H. F.; RAMOS, F. M. Numerical inversion of two-dimensional geoelectric conductivity distributions from magnetotelluric data. **Revista Brasileira de Geofísica**, v. 15, n. 2, p. 133–144, 1997. 12

CAMPOS VELHO, H. F.; SHIGUEMORI, E. H.; RAMOS, F. M.; CARVALHO, J. C. A unified regularization theory: the maximum non-extensive entropy principle. **Computational & Applied Mathematics**, v. 25, n. 2-3, p. 307–330, 2006. 12, 91

CASTRO, A. P. A.; DRUMMOND, I. N.; SILVA, J. D. S. A multiscale neural network for image restoration. **TEMA**, v. 9, n. 1, p. 41–50, 2008. 42, 44, 45, 82

CASTRO, A. P. A. de. **Restauração de imagens por operadores modelados por redes neurais artificiais**. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2009. 43

CHAMBOLLE, A. An algorithm for total variation minimization and applications. **Journal of Mathematical Imaging and Vision**, v. 20, n. 1, p. 89–97, 2004. 14, 91

CHAMBOLLE, A.; LIONS, P.-L. Image recovery via total variation minimization and related problems. **Numerische Mathematik**, v. 76, n. 2, p. 167–188, 1997. 14

CHANG, S. G.; YU, B.; VETTERLI, M. Adaptive wavelet thresholding for image denoising and compression. **IEEE Transactions On Image Processing**, v. 9, n. 9, p. 1532–1546, 2000. 29, 30

CHEN, G. Y.; BUI, T. D.; KRZYZAK, A. Image denoising using neighbouring wavelet coefficients. In: IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING. **Proceedings...** [S.l.], 2005. v. 12, n. 1, p. 99–107. 91

CHEN, H.; ZHANG, Y.; KALRA, M. K.; LIN, F.; CHEN, Y.; LIAO, P.; ZHOU, J.; WANG, G. Low-dose ct with a residual encoder-decoder convolutional neural network. **IEEE Transactions on Medical Imaging**, v. 36, n. 12, p. 2524–2535, 2017. 3

CHEN, K. Introduction to variational image-processing models and applications. **International Journal of Computer Mathematics**, v. 90, n. 1, p. 1–8, 2013. 91

CHENG, X.; ZHANG, Z. Denoising method of heart sound signals based on self-construct heart sound wavelet. **Aip Advances**, v. 4, n. 8, p. 087108, 2014. 24, 25

CIDADE, G. A. G.; SILVA NETO, A. J.; ROBERTY, N. C. **Restauração de imagens com aplicações em biologia e engenharia**. [S.l.]: SBMAC-Notas de Matemática Aplicada – republicação do livro original de 2003, 2012. 13, 90

COIFMAN, R. R.; DONOHO, D. L. Translation-invariant de-noising. **Lecture Notes in Statistics**, p. 125–150, 1995. ISSN 0930-0325. Disponível em: <https://doi.org/10.1007/978-1-4612-2544-7_9>. 31, 32

CUN, X.; PUN, C.-M. **Split then refine: stacked attention-guided ResUNets for blind single image visible watermark removal**. 2020. Disponível em: <https://arxiv.org/abs/2012.07007>. 1

DAHL, G. E.; JAITLY, N.; SALAKHUTDINOV, R. **Multi-task neural networks for qsar predictions**. 2014. Disponível em: <https://arxiv.org/abs/1406.1231>. 91

DAUBECHIES, I. **Ten lectures on wavelets**. [S.l.]: Siam, 1992. 9, 18, 19, 23, 26, 27

DAVIS, P. J. **Circulant matrices**. [S.l.]: Wiley, 1979. 11

DOMINGUES, M. O.; KAIBARA, M. K. Wavelets biortogonais. **Revista Brasileira de Ensino de Física**, v. 34, p. 1 – 16, 09 2012. ISSN 1806-1117. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&

pid=S1806-11172012000300019&nrm=iso>. Acesso em: 20 jan. 2021. 17, 18, 24, 26

DONOHO, D. L.; JOHNSTONE, J. M. Ideal spatial adaptation by wavelet shrinkage. **Biometrika**, v. 81, n. 3, p. 425–455, 1994. 30

DURAN, J.; COLL, B.; SBERT, C. Chambolle's Projection Algorithm for Total Variation denoising. **Image Processing On Line**, v. 3, p. 311–331, 2013. https://doi.org/10.5201/ipol.2013.61. 13, 15

FOWLER, J. E. The redundant discrete wavelet transform and additive noise. **IEEE Signal Processing Letters**, v. 12, n. 9, p. 629–632, 2005. 31

GONZALEZ, R. C.; WOODS, R. E. **Processamento de imagens digitais**. [S.l.]: Blucher, 2010. 233 p. 17

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. Disponível em: <http://www.deeplearningbook.org>. 38

GULL, S. F.; DANIELL, G. J. Image reconstruction from incomplete and noisy data. **Nature**, v. 272, n. 5655, p. 686–690, 1978. 12

HADARMARD, J. Sur les problèm aux dérivées partielles et leur signification physicque. **Bulletin Princeton University**, v. 13, p. 49–52, 1902. 8

HAMILA, O.; RAMANNA, S.; HENRY, C. J.; KIRANYAZ, S.; HAMILA, R.; MAZHAR, R.; HAMID, T. **Fully automated 2d and 3d convolutional neural networks pipeline for video segmentation and myocardial infarction detection in echocardiography**. 2021. Disponível em: <https://arxiv.org/abs/2103.14734>. 39

HANISCH, R. J. Image restoration for the Hubble Space Telescope. In: CRAWFORD, D. L.; CRAINE, E. R. (Ed.). **Proceedings...** SPIE, 1994. v. 2198, p. 1349 – 1356. Disponível em: <https://doi.org/10.1117/12.176820>. 2

HANSEN, P. **Rank-Deficient and discrete ill-posed problems: numerical aspects of linear inversion**. SIAM, 1998. ISBN 9780898719697. Disponível em: <https://books.google.com.br/books?id=0-WHmLCofBOC>. 9

HANSEN, P.; JENSEN, S. Fir filter representations of reduced-rank noise reduction. **IEEE Transactions on Signal Processing**, v. 46, p. 1737–1741, 06 1998. Disponível em: <https://doi.org/10.1109/78.678511>. 9, 17

HAYKIN, S. **Neural networks: a comprehensive foundation**. [S.l.]: Prentice Hall PTR, 1999. 35, 36, 37, 38

HE, K.; ZHANG, X.; REN, S.; SUN, J. **Deep residual learning for image recognition**. 2015. Disponível em: <https://arxiv.org/abs/1512.03385>. 41

HOERL, A. E.; KENNARD, R. W. Ridge regression: Biased estimation for nonorthogonal problems. **Technometrics**, v. 12, n. 1, p. 55–67, 1970. Disponível em: <https://doi.org/10.1080/00401706.1970.10488634>. 10

HÖGBOM, J. Aperture synthesis with a non-regular distribution of interferometer baselines. **Astronomy and Astrophysics Supplement Series**, v. 15, p. 417, 1974. 1

HUBBLESITE. **HubbleSite**. 2020. Disponível em: <https://hubblesite.org/>. Acesso em: 15 maio 2020. 47

IIZUKA, S.; SIMO-SERRA, E. DeepRemaster: Temporal Source-Reference Attention Networks for Comprehensive Video enhancement. **ACM Transactions on Graphics**, v. 38, n. 6, p. 1–13, 2019. 4

IMAGENET. **Image database organized according to the WordNet hierarchy**. 2020. Disponível em: <http://image-net.org/>. 41

KINGMA, D. P.; BA, J. **Adam: a Method for Stochastic Optimization**. 2014. Disponível em: <https://arxiv.org/abs/1412.6980>. 38

KOLMOGOROV, A. N. **Stationary sequences in Hilbert space**. [S.l.]: John Crerar Library National Translations Center, 1978. 15

KOZAKEVICIUS, A.; SCHMIDT, A. A. Wavelet transform with special boundary treatment for 1d data. **Computational and Applied Mathematics**, p. 447–457, 2013. 27

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. **Advances in Neural Information Processing Systems**, v. 25, p. 1097–1105, 2012. 91

KRUITHOF, M. C.; EEKEREN, A. W. M. van; DIJK, J.; SCHUTTE, K. Single image super resolution via sparse reconstruction. In: AHMAD, F. (Ed.). **Proceedings...** SPIE, 2012. v. 8365, p. 84 – 90. Disponível em: <https://doi.org/10.1117/12.919036>. 4

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. **Neural Computation**, v. 1, n. 4, p. 541–551, 1989. Disponível em: <https://doi.org/10.1162/neco.1989.1.4.541>. 39

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998. 40

LEDIG, C.; THEIS, L.; HUSZAR, F.; CABALLERO, J.; CUNNINGHAM, A.; ACOSTA, A.; AITKEN, A.; TEJANI, A.; TOTZ, J.; WANG, Z.; SHI, W. **Photo-realistic single image super-resolution using a generative adversarial network**. 2017. Disponível em: <https://arxiv.org/abs/1609.04802>. 41

LEE, G.; GOMMERS, R.; WASELEWSKI, F.; WOHLFAHRT, K.; O'LEARY, A. Pywavelets: a python package for wavelet analysis. **Journal of Open Source Software**, v. 4, n. 36, p. 1237, 2019. Disponível em: <https://doi.org/10.21105/joss.01237>. 27, 59, 72, 91

LEHTINEN, J.; MUNKBERG, J.; HASSELGREN, J.; LAINE, S.; KARRAS, T.; AITTALA, M.; AILA, T. **Noise2Noise: learning image restoration without clean data**. 2018. Disponível em: <https://arxiv.org/abs/1803.04189>. 1, 40, 41, 88

LIAN, N.-X.; ZAGORODNOV, V.; TAN, Y.-P. Color image denoising using wavelets and minimum cut analysis. **IEEE Signal Processing Letters**, v. 12, p. 741 – 744, 12 2005. Disponível em: <https://doi.org/10.1109/LSP.2005.856865>. 31

LU, Z.-M.; GUO, S.-Z. **Lossless information hiding in images**. [S.l.]: Syngress, 2016. 1

MALLAT, S. **A wavelet tour of signal processing**. 3. ed. [S.l.]: Academic Press, 2008. 832 p. 17, 19, 20, 21, 24, 27, 29

MAO, X.-J.; SHEN, C.; YANG, Y.-B. **Image restoration using convolutional auto-encoders with symmetric skip connections**. 2016. Disponível em: <https://arxiv.org/abs/1606.08921>. 41

MATHWORKS. **Least asymmetric wavelet and phase**. 2020. Disponível em: <https://www.mathworks.com/help///wavelet/ug/least-asymmetric-wavelet-and-phase.html>. Acesso em: 15 maio 2020. 23

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, v. 5, n. 4, p. 115–133, 1943. 35

MONEGO, V. S.; KOZAKEVICIUS, A. J.; CAMPOS VELHO, H. F. Restoration of astronomical images by wavelet techniques. In: IBERIAN LATIN-AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGINEERING, 2020. **Proceedings...** [S.l.], 2020. 34

MOROZOV, V. A. **Methods for solving incorrectly posed problems**. [S.l.]: Springer Science & Business Media, 2012. 9

MOROZOV, V. A.; STESSIN, M. **Regularization methods for ill-posed problems**. Boca Raton: CRC Press, 1993. 91

MUNIZ, W. B.; CAMPOS VELHO, H. F.; RAMOS, F. M. A comparison of some inverse methods for estimating the initial condition of the heat equation. **Journal of Computational and Applied Mathematics**, v. 103, n. 1, p. 145–163, 1999. 91

NIU, Y.; SHEN, L. Wavelet denoising using the pareto optimal threshold. **International Journal of Computer Science and Network Security**, v. 7, n. 1, p. 30, 2007. 29

ORIEUX, F.; GIOVANNELLI, J.-F.; RODET, T. Bayesian estimation of regularization and point spread function parameters for wiener-hunt deconvolution. **Journal of the Optical Society of America A**, v. 27, n. 7, p. 1593, Jun 2010. ISSN 1520-8532. Disponível em: <https://doi.org/10.1364/josaa.27.001593>. 67

PODER, E. **CNN-based search model underestimates attention guidance by simple visual features**. 2021. Disponível em: <https://arxiv.org/abs/2103.15439>. 39

POSIRCA, I.; CHENA, Y.; BARCELOS, C. Z. A new stochastic variational pde model for soft mumford-shah segmentation. **Journal of Mathematical Analysis and Applications**, v. 384, p. 104–114, 2011. ISSN 0930-0325. Disponível em: <https://doi.org/10.1007/978-1-4612-2544-7_9>. 91

PYWAVELETS. **Wavelet browser**. 2020. Disponível em: <http://wavelets.pybytes.com/>. Acesso em: 15 maio 2020. 22, 25

RAJNI, R.; ANUTAM, A. Image denoising techniques - an overview. **International Journal of Computer Applications**, v. 86, 12 2013. Disponível em: <https://doi.org/10.5120/15069-3436>. 29

RAMOS, F. R.; CAMPOS VELHO, H. F.; CARVALHO, J. C.; FERREIRA, N. J. Novel approaches on entropic regularization. **Inverse Problems**, v. 46, n. 5, p. 1139–1148, 1999. Disponível em: <https://doi.org/10.1088/0266-5611/15/5/302>. 12

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: NAVAB, N.; HORNEGGER, J.; WELLS, W. M.; FRANGI, A. F. (Ed.). **Proceedings...** Cham: Springer, 2015. p. 234–241. ISBN 978-3-319-24574-4. 40

RUDIN, L. I.; OSHER, S.; FATEMI, E. Nonlinear total variation based noise removal algorithms. **Physica D: Nonlinear Phenomena**, v. 60, n. 1-4, p. 259–268, 1992. 13, 14, 91

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533–536, 1986. Disponível em: <https://doi.org/10.1038/323533a0>. 35

SHEN, L.; SUN, Q. Biorthogonal wavelet system for high-resolution image reconstruction. **IEEE Transactions on Signal Processing**, v. 52, p. 1997 – 2011, 08 2004. Disponível em: <https://doi.org/10.1109/TSP.2004.828939>. 26

SHIGUEMORI, A. P.; DANTAS, M. S.; SHIGUEMORI, E. H.; KOZAKEVICIUS, A. J.; MONEGO, V. S.; RUIZ, R. S. R.; STRIEDER, C.; CAMPOS VELHO, H. F. Methods for astronominal image restoration. In: IBERIAN LATIN-AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGINEERING. **Proceedings...** [S.l.], 2017. 61, 62

SHIGUEMORI, E. H.; CAMPOS VELHO, H. F.; SILVA, J. D. S. da. Generalized discrepancy principle. In: COLAÇO, M. J.; ORLANDE, H. R. B.; DULIKRAVICH, G. S. (Ed.). **Inverse problems, design and optimization**. [S.l.]: E-papers, 2004. v. 1, p. 145. 91

SILVEIRA, T. da; KOZAKEVICIUS, A. **Transformada wavelet de Haar: conceitos, formulações e aplicações**. [S.l.]: SBM, 2016. 70 p. 20

SOUIBGUI, M. A.; KESSENTINI, Y. De-gan: A conditional generative adversarial network for document enhancement. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, p. 1–1, 2021. ISSN 1939-3539. Disponível em: <http://dx.doi.org/10.1109/TPAMI.2020.3022406>. 1

SRIDHAR, S.; KUMAR, P. R.; RAMANAIAH, K. Wavelet transform techniques for image compression-an evaluation. **International Journal of Image, Graphics and Signal Processing**, v. 6, n. 2, p. 54, 2014. 21

STANHILL, D.; ZEEVI, Y. Y. Two-dimensional orthogonal filter banks and wavelets with linear phase. **IEEE Transactions on Signal Processing**, v. 46, n. 1, p. 183–190, 1998. 22

STUTZ, D.; SILVA NETO, A. J. **Fundamentos de computação paralela para a restauração de imagens de microscopia de força atômica**. [S.l.]: SBMAC-Notas de Matemática Aplicada, 2011. 91

SUN, B.; ZHANG, T.; SU, J.; SHA, H. **GnetDet: object detection optimized on a 224mW CNN accelerator chip at the speed of 106FPS**. 2021. Disponível em: <https://arxiv.org/abs/2103.15756>. 39

TÓTH, L. Phone recognition with deep sparse rectifier neural networks. In: IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING. **Proceedings...** [S.l.], 2013. p. 6985–6989. 36

UCHIDA, Y. **An unofficial and partial Keras implementation of 'Noise2Noise: learning Image Restoration without Clean Data'**. 2018. Disponível em: <https://github.com/yu4u/noise2noise>. Acesso em: 6 out. 2020. 41, 79

VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; WALT, S. J. van der; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES, E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VANDERPLAS, J.; LAXALDE, D.; PERKTOLD, J.; CIMRMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; MULBREGT, P. van; SciPy 1.0

Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. **Nature Methods**, v. 17, p. 261–272, 2020. Disponível em: <https://doi.org/10.1038/s41592-019-0686-2>. 59

WALT, S. van der; SCHÖNBERGER, J. L.; NUNEZ-IGLESIAS, J.; BOULOGNE, F.; WARNER, J. D.; YAGER, N.; GOUILLART, E.; YU, T. Scikit-image: Image processing in python. **PeerJ**, v. 2, p. e453, jun 2014. Disponível em: <https://doi.org/10.7717/peerj.453>. 59

WANG, Z.; BOVIK, A. C.; SHEIKH, H. R.; SIMONCELLI, E. P. Image quality assessment: From error visibility to structural similarity. **IEEE Transactions on Image Processing**, v. 13, n. 4, p. 600–612, 2004. 58

WIENER, N. **Extrapolation, Interpolation and Smoothing of Stationary Time Series: With Engineering Applications**. [S.l.]: MIT press Cambridge, 1950. 15

WIKIPEDIA. **Black hole**. 2020. Disponível em: <https://en.wikipedia.org/w/index.php?title=Black_hole&oldid=957000029>. Acesso em: 28 maio 2020. 2

_____. **Convolutional neural network**. 2020. Disponível em: <https://en.wikipedia.org/w/index.php?title=Convolutional_neural_network&oldid=956628497>. Acesso em: 15 maio 2020. 39, 40

_____. **Discrete wavelet transform**. 2020. Disponível em: <https://en.wikipedia.org/w/index.php?title=Discrete_wavelet_transform&oldid=952525557>. Acesso em: 15 maio 2020. 21

_____. **Hubble Space Telescope**. 2020. Disponível em: <https://en.wikipedia.org/w/index.php?title=Hubble_Space_Telescope&oldid=959234715>. Acesso em: 28 maio 2020. 2

_____. **Abdominal aortic aneurysm**. 2021. Disponível em: <https://en.wikipedia.org/w/index.php?title=Abdominal_aortic_aneurysm&oldid=1014555429>. Acesso em: 9 abril 2021. 3

_____. **Residual neural network**. 2021. Disponível em: <https://en.wikipedia.org/w/index.php?title=Residual_neural_network&oldid=1016523304>. Acesso em: 8 abril 2021. 41

WILLIAMS, J. R.; AMARATUNGA, K. Introduction to wavelets in engineering. **International Journal for Numerical Methods in Engineering**, v. 37, n. 14, p. 2365–2388, 1994. 17

YE, S.; LONG, Y.; CHUN, I. Y. **Momentum-net for low-dose CT image reconstruction**. 2020. Disponível em: <https://arxiv.org/abs/2002.12018>. 40

ZHANG, S.; ZHANG, S.; WANG, Y. **Biorthogonal wavelets in image compression**. [s.n.], 2012. 590-593 p. ISBN 978-1-4577-2144-1. Disponível em: <https://doi.org/10.1109/ICICIP.2012.6391511>. 26

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.