Universidade Federal do Rio de Janeiro
Centro de Ciências Matemáticas e da Natureza
Observatório do Valongo
Departamento de Astronomia

A Transformada de Wavelet Aplicada a Imagens Astronômicas

Projeto de Final de Curso Para Obtenção do Título de Astrônomo

Aluno
Daniel Nicolato Epitácio Pereira

Orientador
Dr. Carlos Roberto Rabaça
(Observatório do Valongo, UFRJ)

Agradecimentos

A Ana Beatriz, meu grande amor, minha companheira de todas as horas e fonte de toda a minha felicidade.

Por encher meu coração de alegria.

Por sua atenção, dedicação e afeto.

Pelo conforto nas horas difíceis e pelo seu amor, em todas as horas.

Que nossos caminhos permaneçam juntos por muitos e muitos anos.

Aos grandes amigos Paulo Fernando Penteado, Rodolfo Smiljanic, Rodolfo Braga e Leandro Guedes. Cada um deles, ao seu próprio modo, fez a minha vida um pouco melhor durante esses anos. A Thiago, Francisco e Márcio, grandes amigos há muito tempo, dos quais os estudos me afastaram temporariamente. Aos muitos outros amigos, colegas de curso ou não, que são muitos para que possam todos ser mencionados.

Aos amigos por natureza, minha família, pela educação, pelo conforto, pela compreensão e pela cumplicidade. Aos meus irmãos Lucas e Marta, ao meu pai José, e especialmente à minha mãe, Mara de Lisieux, que apesar de ter atravessado uma das etapas mais difíceis de sua vida nesses últimos anos, nunca deixou de me apoiar em minhas escolhas e de rezar pela minha felicidade. O amor e o carinho não passarão.

A outros amigos, que apesar de menos humanos, não são menos amigos. Por sua imensa alegria, capaz de invadir meu coração e nas horas mais difíceis me fazer sorrir. Pelo seu empenho em me mostrar o quanto sou querido, mesmo quando tão castigados pelo destino.

Ao meu orientador, Carlos Roberto Rabaça, pela oportunidade de desenvolver este trabalho, pelos inúmeros conselhos e recomendações, alguns dos quais se mostraram muito valiosos, e pela paciência e compreensão durante meus momentos difíceis.

Aos bons professores que me forneceram o conhecimento e a estrutura necessários à conclusão de mais essa etapa da minha formação.

Aos colegas de curso que em algum momento compartilharam comigo informações preciosas para a sobrevivência às disciplinas e àqueles que, ao me pedir ajuda na solução de algum problema, acabaram me ajudando ainda mais.

E, novamente, à minha amada Ana Beatriz, por seus talentos como digitadora e organizadora, que se mostraram vitais para a materialização deste documento.

Aos que fazem o bem e aos que não fazem, mas tentam fazer

E aos que não tentam, mas tentam tentar, tentando querer

E aos que não querem, nem tentam querer, mas ousam ouvir

E aos que não ousam, mas tentam parar, querendo escutar

E aos que não param, nem tentam parar, mas podem tentar

Abstract

Wavelets and other multiscale methods are applied to different areas of knowledge and overcome traditional techniques, like Fourier or Gabor, in dealing with several specific problems. Although the theory behind wavelets was developed almost thirty years ago, its impact in astronomy was not fully accessed until very recently. This work introduces a wavelet-based software written on the course of the last three years at the Observatório do Valongo/UFRJ. Applications developed include denoising, hierarchical detection of sources, and modeling of sources with arbitrary brightness profiles under non-ideal conditions. We also present results of controlled tests, and some applications to astronomy, such as to the structural analysis of Hickson's compact groups of galaxies and a multiscale vision of planetary nebulae.

Keywords: Wavelets, Astronomical Image Processing, Multiscalar Analysis, Noise, Source Detection, Image Reconstruction, Compact Groups, Galaxies, Planetary Nebulae

Resumo

Wavelets e outras formas de análise multiescalar têm sido amplamente empregadas em diversas áreas do conhecimento, sendo reconhecidamente superiores a técnicas mais tradicionais, como as análises de Fourier e de Gabor, em certas aplicações. Embora a teoria dos wavelets tenha começado a ser elaborada há quase trinta anos, seu impacto no estudo de imagens astronômicas tem sido pequeno até bem recentemente. Apresentamos um conjunto de programas desenvolvidos ao longo dos últimos três anos no Observatório do Valongo/UFRJ que possibilitam aplicar essa poderosa ferramenta a problemas comuns em astronomia, como a remoção de ruído, a detecção hierárquica de fontes e a modelagem de objetos com perfis de brilho arbitrários em condições não ideais. Mostramos também os resultados de testes controlados, além de aplicações astrofísicas. As aplicações estudadas incluem a análise da estrutura de grupos compactos de galáxias de Hickson e o estudo de nebulosas planetárias no espaço multiescalar.

Palavras-Chave: Wavelets, Processamento de Imagens Astronômicas, Análise Multiescalar, Ruído, Detecção de Fontes, Reconstrução de Imagens, Grupos Compactos, Galáxias, Nebulosas Planetárias

Índice

Prefácio

Este documento apresenta, sob a forma de um projeto final de curso de graduação, um pacote de programas para a análise multiescalar de imagens astronômicas que vem sendo desenvolvido há três anos no Observatório do Valongo, assim como aplicações a problemas astrofísicos.

O Capítulo 1 apresenta o tipo de problema em cuja solução o pacote pode ser empregado; introduz a transformada de wavelet, operação em que se baseia toda a operação dos programas; e lista as aplicações já desenvolvidas, apresentando a organização do pacote.

Os Capítulos 2 e 3 descrevem cada módulo desenvolvido em detalhe. O Capítulo 2 é reservado aos elementos fundamentais do pacote, como a transformada discreta de wavelet e a análise de ruído, que geram as informações necessárias a outros módulos. O Capítulo 3, por sua vez, trata dos programas avançados, que realizam as operações finais de remoção do ruído, detecção de fontes e reconstrução parcial da imagem. Cada módulo recebe uma descrição teórica, seguida de uma discussão sobre as técnicas adotadas em sua implementação.

Alguns testes realizados quanto à eficiência dos módulos são apresentados no Capítulo 4. Dois problemas astrofísicos e a forma como o pacote foi utilizado em seu estudo também são discutidos.

Finalmente, no Capítulo 5, discorremos sobre o progresso atingido até o momento e sobre o que ainda há para ser feito.

Três apêndices estão disponíveis para complementação didática: um sobre tópicos comuns em processamento de imagens, convolução e filtros; outro sobre as distribuições de probabilidade relevantes à teoria por trás dos módulos desenvolvidos e, por fim, um resumo da notação utilizada em fórmulas e expressões ao longo do documento, para consulta rápida.

Ao fim do documento, encontram-se uma seção dedicada às referências bibliográficas e um índice remissivo, destinado a auxiliar na localização da definição de termos específicos, ou de referências relevantes aos mesmos. Para facilitar sua localização, termos importantes, quando sendo definidos ou introduzidos, são marcados em negrito.

Os pesquisadores Silvia Lorenz-Martins, François Cuisinier, Cláudia Mendes de Oliveira e Cristiano da Rocha, assim como o estudante de graduação Rodrigo Muniz de Moura, colaboraram de forma especial no desenvolvimento de algumas etapas do projeto e, portanto, deixamos aqui nosso agradecimento.

Agradecemos também ao CNPq, que apoiou este projeto através da concessão de uma bolsa de iniciação científica PIBIC e à FAPERJ, que financiou a aquisição de parte dos equipamentos utilizados através de bolsa APQ-1 (processo E-26/171.085-99).

Algumas imagens utilizadas neste projeto foram obtidas do *Canadian Astronomy Data Center* (CADC), operado pelo *Dominion Astrophysical Observatory* para o Instituto de Astrofísica do Conselho Nacional de Pesquisa do Canadá.

1 Introdução: Visão Multiescalar

Uma nova forma de encarar antigos problemas

Uma das formas mais comuns de dados astronômicos é a imagem direta. Adquirida em geral por um CCD (*Charge-Coupled Device*), consiste em um conjunto de valores de intensidade associados a pontos (ou píxeis[1], freqüentemente espaçados de forma regular) referentes a direções discretas no céu. A representação computacional mais natural para esse tipo de informação é uma matriz bidimensional em que a cada elemento é associado um valor, em geral correspondendo à intensidade (ou contagem de elétrons) do ponto correspondente.

O tratamento de uma imagem como um conjunto de valores de píxeis é apropriado para operações fundamentais em seus dados, além de consistir na forma mais simples de armazená-los.

Muitas operações recorrentes em astronomia dependem, no entanto, de um tratamento da imagem como um conjunto de objetos físicos. Esse tipo de tratamento pode ser alcançado de várias formas diferentes. A transformada de Fourier consiste em uma boa ferramenta para esse tipo de análise, transportando a informação da imagem para uma base de freqüências e possibilitando, então, uma visão multifreqüência de seu conteúdo.

Um novo tipo de visão tem sido cada vez mais empregado, ora substituindo, ora complementando a visão multifreqüência de Fourier: a visão multiescalar. Obtida através de transformadas multiescalares, como a de wavelet, confere a perspectiva de um espaço que tem como bases tanto as freqüências locais quanto as posições espaciais,

---

[1] O píxel (do inglês "*píxel*", uma redução de "*picture element*" – "elemento de figura", segundo alguns autores ou de "*picture cell*" – "célula de figura", segundo outros) é a unidade fundamental de dados de uma imagem, representando uma região, em geral retangular, em que se verifica uma certa intensidade (ou um conjunto de intensidades, no caso em que haja mais que um canal, como em imagens coloridas). Em um dispositivo de CCD, um píxel corresponde a uma célula de armazenamento de elétrons, liberados através da interação entre uma camada de material fotossensível e fótons produzidos pelos objetos observados.

sendo a resolução da análise vinculada à freqüência analisada, apresentando-se sob a forma de um parâmetro que denominamos escala.

Neste capítulo apresentamos uma série de problemas multiescalares em astronomia, introduzimos as transformadas multiescalares e descrevemos em resumo o conteúdo do pacote que temos desenvolvido para o tratamento desses problemas.

1.1 Problemas multiescalares em astronomia

Uma imagem de natureza astronômica consiste tipicamente de uma soma de três componentes principais[1]: um conjunto de estruturas bem localizadas espacialmente; uma componente de fundo, que associamos ao céu, de variação relativamente suave ao longo da imagem; e uma componente ruidosa que cobre toda imagem, contendo flutuações aleatórias oriundas de processos diversos e, em geral, bastante intensa (em comparação com sua importância relativa em imagens terrestres, por exemplo).

As estruturas localizadas representam toda sorte de objetos celestes. Estrelas, os objetos mais simples do ponto de vista do imageamento direto, consistem, para efeitos práticos, em fontes pontuais de radiação. Sua representação nos píxeis da imagem terá a forma da função de espalhamento de pontos (ou PSF)[2] relativa às condições de observação. Objetos não estelares possuem uma distribuição de brilho própria, e sua representação na imagem consiste na convolução[3] dessa distribuição pela PSF atuante na imagem. Esses objetos podem ser, por exemplo, planetas, asteróides, galáxias, nebulosas planetárias, aglomerados estelares e aglomerados e grupos de galáxias.

---

[1] Após a redução básica dos artifícios de aquisição dos dados (erros instrumentais), naturalmente.

[2] PSF é sigla da forma inglesa "*Point Spread Function*". A PSF é uma função com a mesma dimensionalidade do sinal analisado que fornece a distribuição de intensidades registrada no detector para uma única fonte pontual com fluxo unitário. Em imagens astronômicas a PSF é formada principalmente pelo *seeing* (múltiplas refrações dos raios de luz entre células móveis de diferentes densidades na atmosfera terrestre, gerando desvios rápidos e aleatórios de sua direção de detecção), por deformações introduzidas pela ótica específica do coletor (telescópio, por exemplo), e pelo perfil de difração, também depende da montagem do coletor.

[3] Para uma breve revisão sobre a operação de convolução em suas formas contínua e discreta, consulte o Apêndice A.

Muitas vezes, alguns desses objetos são observados como constituintes de outros, como galáxias em um grupo e aglomerados de estrelas em uma galáxia. Outras vezes, objetos sem conexão física real estão inseridos no mesmo espaço na imagem, como no caso de estrelas situadas entre nós e uma galáxia, mostrando-se sobrepostas a esta em uma imagem. Uma característica dessa componente é que a intensidade nos píxeis contidos em uma dessas estruturas será, em geral, positiva, somando-se ao nível de fundo da imagem[1].

O nível de céu e a componente ruidosa não têm qualquer relação com os objetos físicos estudados e, por isso, suas propriedades não são, em geral, o objetivo final da análise de uma imagem. Apesar disso, seu comportamento precisa ser muito bem conhecido para que sua contribuição não seja levada em conta quando examinarmos as estruturas verdadeiras.

Muitos problemas em processamento de imagens astronômicas consistem em determinar o comportamento, e por vezes isolar, uma ou mais das componentes citadas presentes na imagem. A remoção da componente de ruído, por exemplo, é utilizada como etapa de pré-processamento em muitas aplicações de deconvolução e detecção de fontes. A componente de estruturas localizadas é o alvo dos métodos de detecção de fontes, nos quais se deseja subdividir esta componente em estruturas geradas por cada objeto astronômico em particular, em geral determinando algumas de suas propriedades, como fluxo, forma e posição central. Os algoritmos de modelagem e reconstrução de objetos tentam sintetizar uma imagem que contenha apenas alguns objetos desejados, eliminando não só a componente de ruído e a de fundo, como também parte da componente de estruturas. Esses últimos algoritmos possuem inúmeras aplicações em astrofísica, como por exemplo, a identificação e a medição de propriedades de elementos tênues em estruturas compostas ou a determinação das contribuições individuais de objetos sobrepostos, como veremos no Capítulo 4.

Vários métodos tradicionais para a identificação da componente intermediária (objetos) são baseados em uma simples operação de corte (*threshold*) da imagem no

---

[1] Apesar do que possa parecer a princípio, essa afirmação é muito relevante e nem um pouco óbvia. No caso de dados espectrais, por exemplo, linhas e bandas de absorção são estruturas negativas em relação ao contínuo e, mesmo em imagens diretas, é possível pensar em casos especiais em que objetos reais se apresentem como estruturas negativas, como, por exemplo, nebulosas de absorção em um campo rico em objetos luminosos.

espaço direto. O desvio padrão da componente ruidosa é determinado, e regiões da imagem com intensidade maior do esse valor multiplicado por um certo número (geralmente três) são identificadas como pertencentes a um objeto. Esse procedimento simples acaba por ignorar algumas as fontes extensas, e com baixo brilho superficial, que, mesmo com um fluxo integrado considerável, podem estar inteiramente abaixo do nível de corte (ou de detecção), como mostra a figura 1.1 a seguir.

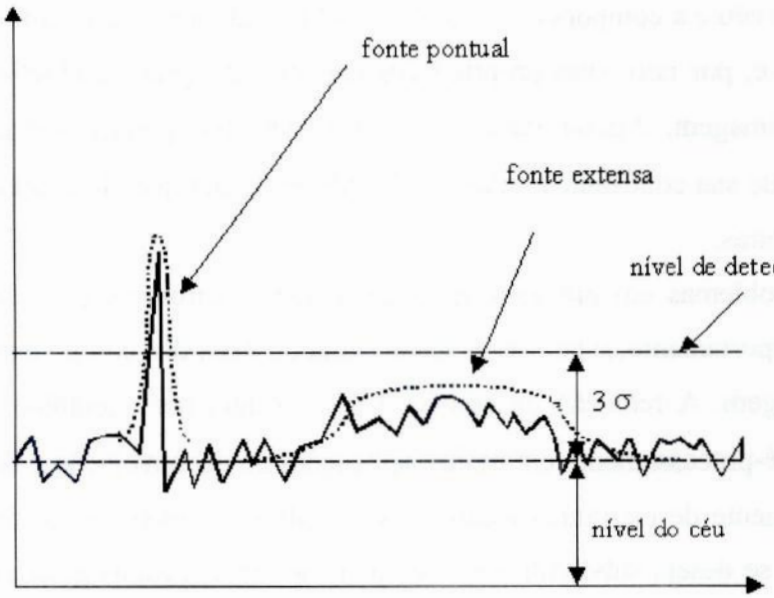


Figura 1.1: O problema da detecção de objetos baseada unicamente no valor dos píxeis no espaço direto. Fontes extensas com baixo brilho superficial podem ser erroneamente descartadas. No gráfico, σ representa o desvio padrão do ruído na imagem. Figura adaptada de Starck et. al (2000).

Outros métodos para a separação dessas três componentes operam através de uma visão multifreqüência. O ruído possui como característica marcante uma variação acentuada entre os valores de píxeis vizinhos, podendo ser considerado uma componente de alta freqüência espacial. O nível do céu varia suavemente ao longo da imagem, podendo então ser tomado como de baixa freqüência. As estruturas correspondentes aos objetos pertenceriam a freqüências intermediárias. Através da análise de Fourier (que definiremos adiante), pode-se separar essas componentes de freqüências diferentes e eliminar uma ou mais delas. No entanto, há diversos problemas relacionados a esse tipo de visão, sendo os principais:

1. Embora o ruído possa ser considerado uma estrutura em que predominam as altas freqüências, é incorreto dizer que não possua componentes de freqüências mais baixas, de forma que a supressão das altas freqüências não o elimina completamente. As estruturas correspondentes a objetos físicos também podem possuir componentes de freqüências muito altas, e que poderiam ser excluídas com o ruído quando este fosse suprimido.

2. Mesmo que se obtivesse uma boa supressão do ruído e da variação de nível de céu, a visão multifreqüência de Fourier não nos permite, em geral, distinguir dentro da componente de estruturas localizadas quais são pertencentes a cada objeto. Objetos com aproximadamente o mesmo tamanho, mas localizados em posições muito diferentes na imagem seriam incluídos em uma mesma componente de freqüência.

Esses problemas tornam mais apropriada uma visão baseada na separação de componentes de escalas de tamanhos diferentes (em lugar de freqüências) e que propicie uma análise local. Esse tipo de visão, denominado multiescalar, nos permite:

1. Identificar o ruído como uma componente de escala característica pequena, mas que pode estar presente em menor intensidade em escalas maiores, sendo que nessas pode se estabelecer um critério de ação local sobre a presença ou não de quantidade expressiva de ruído (ou critério de validade, ou ainda critério de significância). Como nossa análise se torna localizada, podemos apontar regiões em que informações pertinentes aos objetos reais são encontradas e as deixar intocadas ao remover o ruído.

2. Analisar localmente as estruturas em escalas de tamanho diferentes, assim podendo distinguir estruturas correspondentes a fontes distintas, mesmo que de mesmo tamanho aproximado. Podemos também analisar hierarquicamente as estruturas encontradas, estabelecendo conexões entre objetos. Essa visão proporciona um ponto de partida adequado para a reconstrução individual de objetos.

O fato de os problemas relacionados à supressão do ruído e à detecção e reconstrução de objetos poderem ser encarados com proveito através de uma visão multiescalar faz com que sejam reconhecidos como problemas multiescalares. Esses são

os problemas-alvo principais do nosso pacote de programas e as técnicas empregadas para suas soluções são discutidas em detalhe no segundo e no terceiro capítulo deste documento.

1.2 A transformada de wavelet

O método mais tradicional para a análise multiescalar de dados é a **transformada de wavelet**. A teoria acerca dessa transformada é um ramo importante na matemática moderna, atualmente contribuindo de forma expressiva em muitas áreas aplicadas, desde a astronomia e a física até as ciências contábeis e a ecologia. Sua origem é difícil de ser apontada de forma exata, mas muitos a atribuem a Morlet e Grossman, pelo fim da década de 1970. Morlet, então engenheiro de uma companhia de petróleo, a partir da idéia de Gabor para uma transformada de Fourier de curto tempo, construiu a função à qual seu nome seria dado posteriormente, e que permitiria a aplicação de uma transformada integral em que a resolução estivesse vinculada a freqüência – a transformada de wavelet. Grossman, um físico teórico, tendo conhecido o trabalho de Morlet, uniu-se a ele, querendo aplicar suas idéias à mecânica quântica. Deduziu a transformada inversa de wavelet e então os dois juntos investigaram uma série de aplicações da teoria. Daí seguiram-se diversas contribuições notáveis, por parte de Daubechies, Meyer, Lernairé, Battle, Mallat e muitos outros. Trabalhos que precederam o de Morlet e Grossman e que exploraram alguns elementos constituintes da teoria dos wavelets que viria a surgir foram realizados por Haar, Gabor, Alaksen e Klauder e Calderón. Para uma descrição detalhada da história do desenvolvimento da teoria dos wavelets, veja Daubechies (1994).

Iniciaremos nossa discussão sobre a transformada de wavelet dando a definição formal da sua versão contínua e listando suas propriedades. Alguns conceitos serão apresentados. Em seguida mostraremos como a transformada de Fourier relaciona-se com a transformada de wavelet, reproduzindo aproximadamente os passos de Morlet em sua descoberta. Finalmente exploraremos o aspecto mais prático (e mais interessante para nós) da transformada: sua aplicação a funções discretas e de domínio finito, como imagens digitais.

Embora a teoria dos wavelets possua diversos aspectos teóricos complexos, sendo área de grande interesse para a matemática pura, nos concentraremos em mostrar suas propriedades relevantes para as aplicações desenvolvidas. Para introduções à teoria dos wavelets sob pontos de vista um pouco diferentes, incluindo incursões mais profundas em suas peculiaridades matemáticas, veja Strang (1989) ou Daubechies (1996).

1.2.1 Definição e propriedades da transformada de wavelet

A definição de Morlet-Grossman para a **transformada contínua de wavelet** de uma função unidimensional $f(x)$[1] é

$$W(a,b)\,[f] = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(x)\, \psi^* \left(\frac{x-b}{a} \right) dx, \tag{1.1}$$

onde ψ é uma função de wavelet, cuja natureza é discutida adiante, e o asterisco (*) indica a utilização de seu conjugado complexo. Como indicado, a função resultante será dependente de duas variáveis: *b*, relacionada com a posição, e *a*, relacionada com a escala.

Essa transformada possui as três seguintes propriedades:

1. linearidade:

$$W(a,b)\,[k_1 f_1 + k_2 f_2] = k_1\, W(a,b)\,[f_1] + k_2\, W(a,b)\,[f_2], \tag{1.2}$$

onde f_1 e f_2 são duas funções arbitrárias quaisquer e k_1 e k_2 são duas constantes quaisquer. Em palavras, *a transformada de wavelet de uma combinação linear de duas (ou mais, conseqüentemente) funções será a combinação linear das transformadas individuais dessas funções, segundo as mesmas proporções.*

---

[1] Como aqui, em muitas ocasiões, consideraremos em nosso formalismo casos unidimensionais, para simplicidade.

2. covariância a translações:

$$W(a,b)\,[f\,(x-x_o)] = W(a,b-x_o)\,[f(x)], \tag{1.3}$$

onde x_0 é uma constante qualquer. Ou, *a transformada de wavelet da versão deslocada de uma função será igual a uma versão deslocada em sua dimensão espacial correspondente (em mesma quantidade) da transformada de wavelet da função original.*

3. covariância a dilatações:

$$W(a,b)\,[f\,(s.x)] = \frac{1}{\sqrt{s}}\,W(sa,sb)\,[f(x)] \tag{1.4}$$

onde s é uma constante qualquer. Ou seja, *a transformada de wavelet da versão dilatada de uma certa função é uma versão dilatada em todas as suas dimensões (e da mesma quantidade) da transformada de wavelet da função original, multiplicada ainda por um fator de renormalização (a raiz quadrada do inverso do fator de dilatação).*

Essas três condições não apenas são conseqüências diretas de (1.1), mas são equivalentes a essa definição, ou seja, se uma transformação obedece a essas três propriedades, então é uma transformada de wavelet e pode ser dada por (1.1) (Bijaoui e Rué 1996).

A função ψ é qualquer membro de uma família de funções denominadas **funções de wavelet** ou, simplesmente, **wavelets**. Wavelets são funções localizadas (condição de **compacticidade**) e com integral por todo o domínio igual a zero (condição de **admissibilidade**[1]), o que em geral lhes concede o aspecto de uma onda confinada a um

[1] Ou **aceitabilidade**.

domínio finito, sendo essa a causa de seu nome[1]. A escolha apropriada do wavelet a ser usado em cada aplicação é de crucial importância para a obtenção de bons resultados. Veremos agora alguns dos wavelets mais empregados para o tratamento de funções contínuas.

O **wavelet de Haar**, o mais simples existente, é dado pela expressão

$$\psi(x) = \begin{cases} 1 & ; 0 \le x < \frac{1}{2} \\ -1 & ; \frac{1}{2} \le x < 1 \\ 0 & ; x < 0 \ ou \ x \ge 1 \end{cases} . \qquad (1.5)$$

Esse wavelet é também o mais antigo a ser utilizado, tendo suas propriedades como função analisadora descritas várias décadas antes que a teoria matemática dos wavelets tivesse sido construída. Sua forma pode ser vista na figura 1.2. Veremos mais sobre esse wavelet adiante.

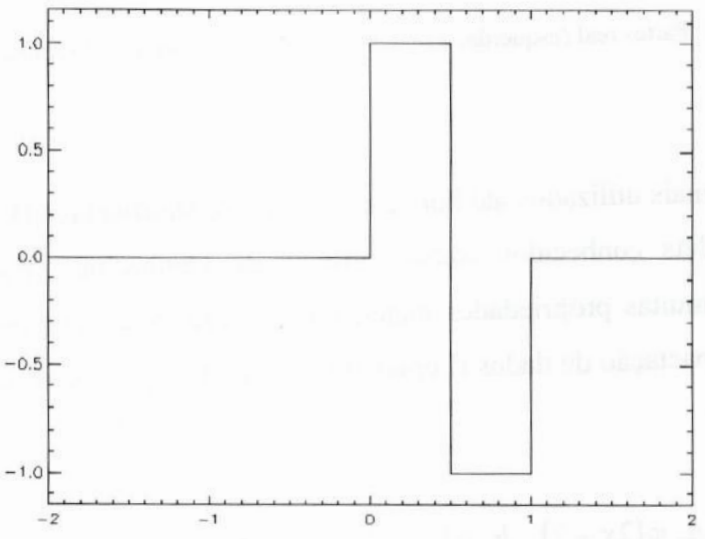


Figura 1.2: Wavelet de Haar.

[1] Do inglês "*wave*" + "*let*", pequena onda. Na França, onde grande parte da teoria dos wavelets foi desenvolvida, é comum o uso da forma "*ondelete*". A esse exemplo, alguns autores de língua portuguesa utilizam a expressão "ondaleta", embora a forma inglesa ainda seja a mais difundida no Brasil.

O **wavelet de Morlet**, um dos pais da teoria dos wavelets, é um wavelet complexo, expresso por:

$$\psi(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2} + i(2\pi v_o x)}, \qquad (1.6)$$

onde v_0 é uma constante, tipicamente escolhida como sendo maior que 0,8. Suas partes real e imaginária podem ser vistas na figura 1.3.

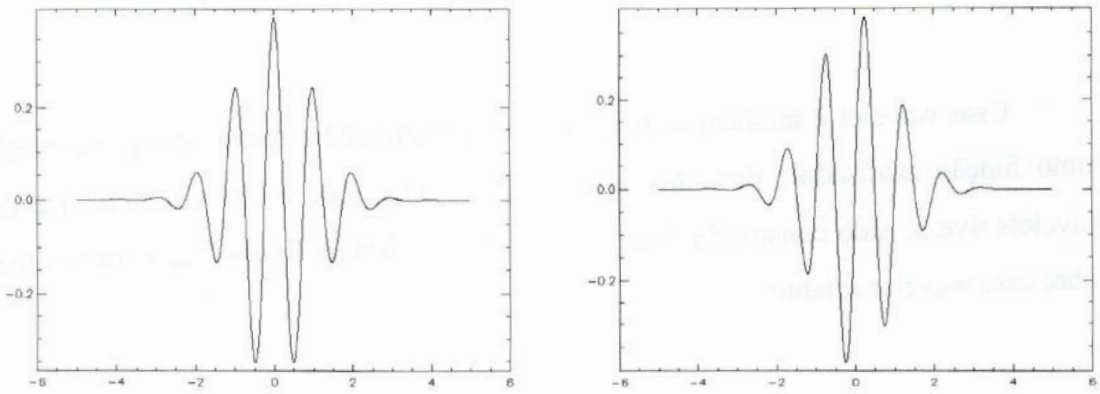


Figura 1.3: Partes real (esquerda) e imaginária (direita) do wavelet de Morlet com $v_0 = 1$.

Um dos mais utilizados até hoje é o **wavelet de Daubechies D_4**, membro de um grupo de wavelets conhecidos como wavelets de Daubechies[1] (D_2, D_4, D_6 etc.), possuidores de muitas propriedades matemáticas interessantes (Strang 1989) e muito aplicados à compactação de dados (Lepley e Forkert 1997, por exemplo). O wavelet D_4 é expresso como

$$\psi(x) = h_3\,\varphi(2x+2) - h_2\,\varphi(2x+1) + h_1\,\varphi(2x) - h_0\,\varphi(2x-1), \qquad (1.7)$$

[1] Na verdade, o wavelet D_2 é o próprio wavelet de Haar, redescoberto por Daubechies mais de 70 anos depois de sua criação.

onde $\varphi(x)$ é a solução da equação de dilatação[1]:

$$\varphi(x) = h_0\,\varphi(2x) + h_1\,\varphi(2x-1) + h_2\,\varphi(2x-2) + h_3\,\varphi(2x-3) \qquad (1.8)$$

e os coeficientes h_0, h_1, h_2 e h_3 possuem os valores, nessa ordem, $\frac{1+\sqrt{3}}{4}$, $\frac{1-\sqrt{3}}{4}$, $\frac{3+\sqrt{3}}{4}$ e $\frac{3-\sqrt{3}}{4}$.

O aspecto bem curioso dessa função é mostrado na figura 1.4. Na verdade, esse wavelet é ainda menos suave do que o que aparenta ser na figura. Se examinássemos uma pequena porção de sua curva, veríamos que reproduz a forma básica da curva como um todo, o que faz com que seja, por vezes, citado como um wavelet fractal.

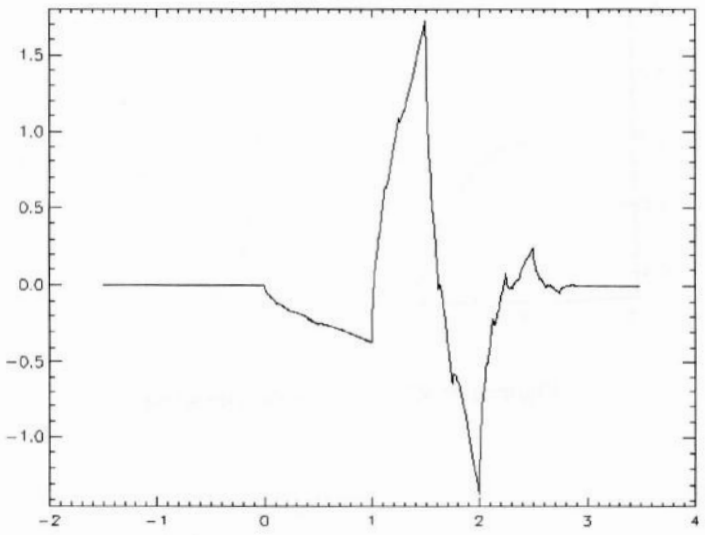


Figura 1.4: Wavelet de Daubechies D_4.

---

[1] Equações de dilatação são equações que possuem a forma

$$\varphi(x) = \sum_k c_k\,\varphi(a\cdot x - k),$$

onde a é uma constante qualquer (no caso mais comum é igual a 2), c_k são coeficientes constantes e $\varphi(x)$ é uma função, em geral a ser determinada.

A teoria matemática a respeito dessas equações está intimamente ligada à dos wavelets. Para mais informações sobre o assunto veja Strang (1989).

Finalmente, o wavelet conhecido como **Chapéu Mexicano** (assim nomeado por causa de seu aspecto, que pode ser visto na figura 1.5) é historicamente um dos mais aplicados a problemas multiescalares astronômicos, devido à sua forma simétrica (que garante uma análise isotrópica). Possui a forma da derivada segunda da função gaussiana:

$$\psi(x) = (1 - x^2)\, e^{\frac{-x^2}{2}} . \tag{1.9}$$

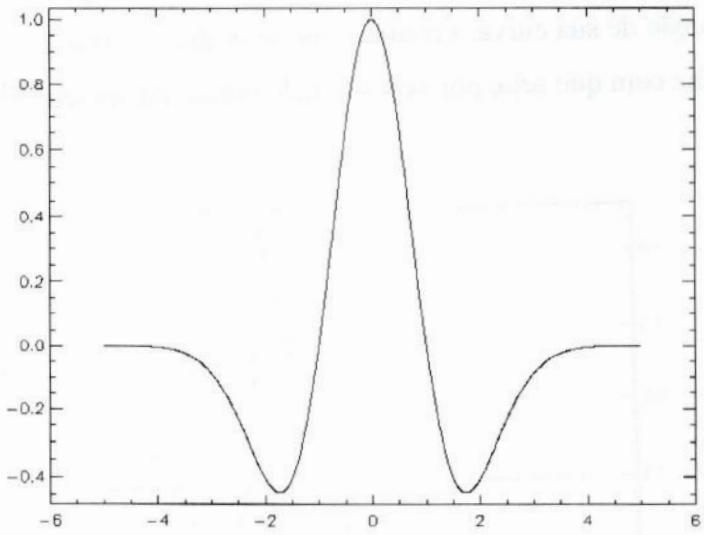


Figura 1.5: Wavelet Chapéu Mexicano

1.2.2 Da análise de Fourier à análise multiescalar

Um método tradicional para o estudo das estruturas de diferentes tamanhos em um conjunto de dados é a **análise de Fourier**, criada pelo matemático francês Jean-Baptiste Joseph Fourier, nos primórdios do século XIX. A análise de Fourier é uma ferramenta de grande sucesso em praticamente todas as ciências exatas, puras ou aplicadas (para um estudo introdutório detalhado sobre o assunto, procure, por exemplo, Hsu 1970 ou Spiegel 1977). Veremos agora como, a partir de algumas ligeiras alterações no formalismo de Fourier e passando pelo de Gabor, chegamos a uma transformada apropriada para a análise multiescalar.

Estando os dados originais representados em um espaço de coordenadas posicionais[1], que chamaremos **espaço direto** daqui em diante, sua **transformada de Fourier** fornece uma representação em freqüências nessas coordenadas, num **espaço de Fourier**. Essa transformada é dada, em sua forma unidimensional, por

$$F(\omega)[f] = \int_{-\infty}^{+\infty} f(x)\, e^{-i\, 2\pi\, \omega\, x} dx\,. \qquad (1.10)$$

Como indicado, a transformada de Fourier de uma função de uma variável é também função de uma única variável, ω. Esta se relaciona com a freqüência de variação da função original ao longo do espaço direto.

Freqüências diferentes devem, em geral, ser associadas a estruturas de tamanhos diferentes, de forma que, nesse espaço, as diferentes escalas de tamanho estariam separadas.

Há um grave problema nesse tipo de tratamento, no entanto. Como analisa estruturas em termos de freqüências, o método de Fourier pressupõe que haja repetição espacial das mesmas por todo o domínio considerado, o que não ocorre em muitos casos, especialmente em imagens. A análise de Fourier não é adaptada, portanto, para a análise de estruturas finitas e localizadas ou sinais não estacionários, além de não conservar as informações sobre as posições dos elementos presentes nos dados originais.

Uma solução para o problema seria modificar a transformada de Fourier para que sua ação se tornasse localizada. Isso pode ser feito com a modulação[2] da função analisadora, no caso um composto complexo de seno e cosseno[3], por uma curva

---

[1] No caso de imagens diretas, essas coordenadas correspondem às espaciais. No estudo de séries temporais, que constitui o emprego mais clássico da análise de Fourier, temos o tempo como coordenada.

[2] A **modulação** de uma função periódica por uma outra função (chamada moduladora, e não necessariamente periódica) é a operação de modificar a sua forma de modo a fazer com que sua amplitude a cada ponto seja igual ao valor da moduladora naquele mesmo ponto. Na prática, se a função periódica possui amplitude constante e igual a 1, a modulação pode ser feita pela simples multiplicação pela função moduladora.

[3] Pela fórmula de Euler, $e^{ix} = \cos(x) + i\,\mathrm{sen}(x)$ para x qualquer.

gaussiana deslocada de forma a seu máximo estar localizado em uma certa posição b e com largura proporcional a um certo valor a, ou seja,

$$e^{-\frac{1}{2}\left(\frac{x-b}{a}\right)^2}, \tag{1.11}$$

gerando uma nova transformação:

$$F'(\omega,a,b)[f] = \int_{-\infty}^{+\infty} f(x)\, e^{-\frac{1}{2}\left(\frac{x-b}{a}\right)^2} e^{-i\,2\pi\,\omega\,x} dx\,. \tag{1.12}$$

Essa transformação (exceto por um fator de normalização) é conhecida como **transformada de Fourier de curto tempo**[1], (Gabor 1946). Note que acrescentamos mais duas dimensões àquela originalmente presente no espaço de Fourier, de forma que nossa representação fornece agora informações em excesso. Em geral, a resolução de uma dada característica do sinal, dada por a, está estritamente relacionada com sua freqüência, dada por ω. Podemos então vincular os dois parâmetros através de

$$a = \frac{C}{\omega} \quad \longrightarrow \quad \omega = \frac{C}{a}\,, \tag{1.13}$$

onde C é uma constante arbitrária, de forma a obtermos uma transformada que leve a uma representação em que estejam presentes apenas a posição e o parâmetro a, que rege tanto resolução quanto freqüência e que, em geral, é chamado de **escala**:

---

[1] É muito comum na teoria de processamento de sinais partir do princípio de que o sinal analisado constitui em uma série de valores associados a tempos diferentes, daí o uso clássico da expressão "curto tempo". Em um conjunto de dados ordenados espacialmente, como uma imagem, poder-se-ia utilizar uma expressão como "transformada de Fourier localizada no espaço", por exemplo, ou simplesmente "transformada localizada de Fourier", o que seria uma forma mais geral. Essa transformada é por vezes chamada de "transformada de Gabor", em homenagem a seu criador. É comum haver uma pequena confusão entre essa transformada e a chamada transformada rápida de Fourier (ou FFT – *Fast Fourier Transform*), devido a problemas de tradução. Essa última, na verdade, trata-se apenas de um algoritmo computacional eficiente para a aplicação numérica da transformada de Fourier comum.

$$F''(a,b)[f] = \int_{-\infty}^{+\infty} f(x)\, e^{-\frac{1}{2}\left(\frac{x-b}{a}\right)^2} e^{-i\,2\pi\,\frac{C\,x}{a}}\, dx. \qquad (1.14)$$

Para garantir que a análise seja feita de forma homogênea pelo domínio do sinal, podemos ainda forçar a fase da função analisadora a acompanhar a posição central da gaussiana, obtendo

$$F'''(a,b)[f] = \int_{-\infty}^{+\infty} f(x)\, e^{-\frac{1}{2}\left(\frac{x-b}{a}\right)^2} e^{-i\,2\pi\,C\left(\frac{x-b}{a}\right)} dx. \qquad (1.15)$$

Os dois últimos passos foram os utilizados por Morlet para obter, a partir da transformada de Fourier de curto tempo, uma transformada que fosse verdadeiramente multiescalar, ou seja, que contivesse apenas posição (ou tempo) e escala. Pode-se facilmente verificar que a expressão obtida corresponde à definição da transformada de wavelet vista em (1.1) com a função (1.6) como wavelet, com $C = -v_0$, e com a introdução de um fator de normalização dependente da escala a.

1.2.3 Aplicação a funções discretas

Diversas técnicas foram desenvolvidas para aplicação da transformada de wavelet a funções discretas, limitadas e bidimensionais, como imagens. A **transformada de Mallat**, de natureza anisotrópica, e a **transformada de Feauveau** são alguns exemplos de técnicas tradicionais (Starck, Murtagh e Bijaoui 1998).

O wavelet de Haar também pode ser aplicado a funções discretas de qualquer dimensionalidade através de uma técnica razoavelmente simples e que ilustra o funcionamento da transformada discreta de wavelet. Essa técnica também fornece uma introdução alternativa para a própria transformada de wavelet, já que atua de forma praticamente independente do formalismo de sua versão contínua (embora compartilhe muitas de suas propriedades). Vejamos um exemplo de sua aplicação:

Seja uma função unidimensional discreta $f(x)$, como a mostrada na figura 1.6. Introduzimos a **função discreta de Haar**, dada por

$$\psi(x) = \begin{cases} 1 \; ; \; x = 0 \\ -1 \; ; \; x = 1 \\ 0 \; ; \; \textit{em outro caso} \end{cases} \tag{1.16}$$

e uma função complementar, chamada **função de escala**:

$$\varphi(x) = \begin{cases} 1 \; ; \; x = 0 \\ 0 \; ; \; \textit{em outro caso} \end{cases} . \tag{1.17}$$

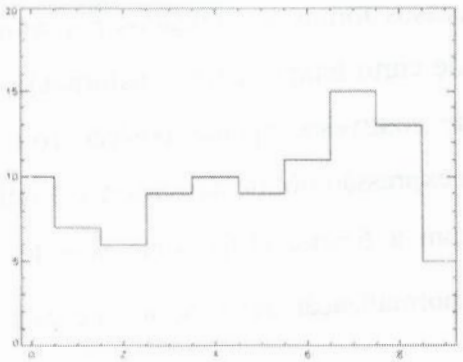


Figura 1.6: Exemplo de função discreta

As funções $\varphi(x)$ e $\psi(x)$ são mostradas na figura 1.7 abaixo:

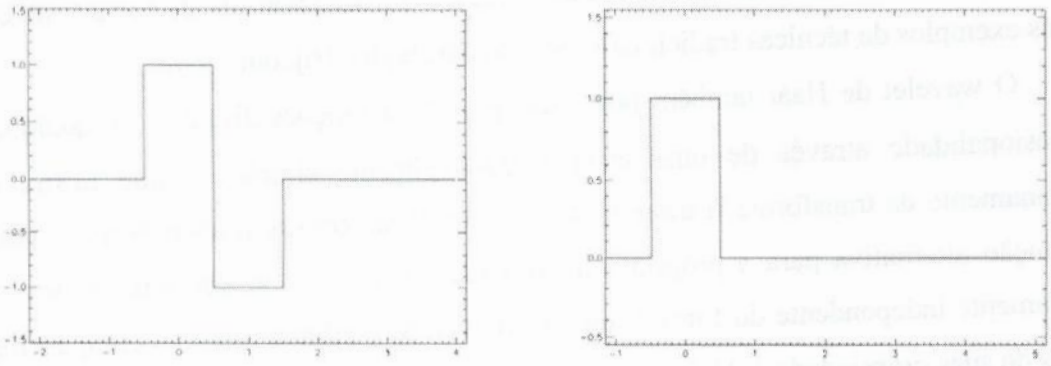


Figura 1.7: Função discreta de Haar (esquerda) e função de escala correspondente (direita)

Definimos agora outras funções, freqüentemente chamadas funções filhas da função de escala, versões dilatadas e deslocadas de φ :

$$\varphi_{a,b}(x) = \varphi\left(\frac{x}{a+1} - b\right), \tag{1.18}$$

onde a divisão realizada é inteira, ou seja, a parte decimal do resultado é desprezada. Algumas dessas funções são mostradas na figura 1.8. Pode-se verificar que $\varphi(x) = \varphi_{0,0}(x)$. Temos que $\varphi_{0,b}$ é um conjunto de funções que possuem o mesmo aspecto de φ, mas deslocadas da origem cada uma por um número inteiro. Se visualizarmos todas ao mesmo tempo em um gráfico, as veremos como uma série de retângulos justapostos, todos com altura e largura 1. Não é difícil perceber que podemos representar com exatidão a nossa função $f(x)$ através de uma combinação linear dessas funções $\varphi_{0,b}$, simplesmente multiplicando cada função com índice b pelo valor de $f(b)$. Ou seja,

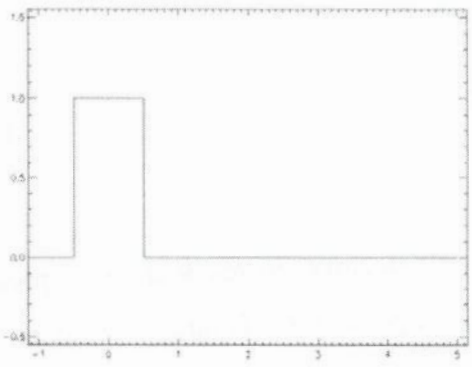
$$f(x) = \sum_b c_{0,b}\ \varphi_{0,b}(x)\,, \tag{1.19}$$

onde

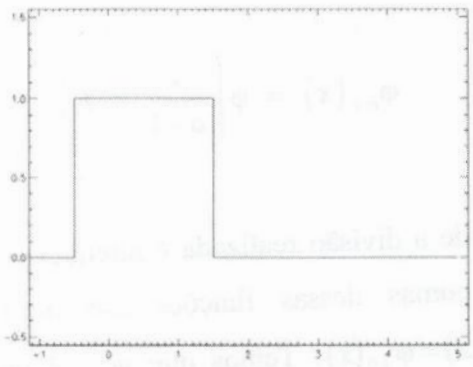
$$c_{0,b} = f(b)\,. \tag{1.20}$$

Dizemos que estamos representando $f(x)$ na base formada pelas funções $\varphi_{0,b}$. Os coeficientes $c_{0,b}$ dessa representação são chamados **coeficientes de escala** de nível 0 (ou escala 0).

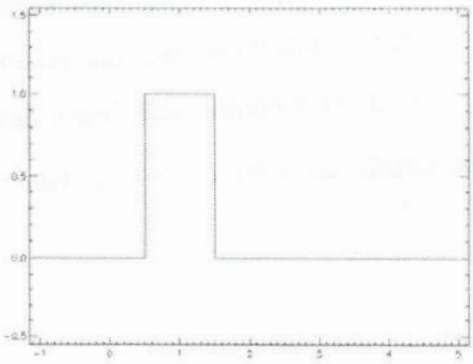
$a = 0, b = 0$



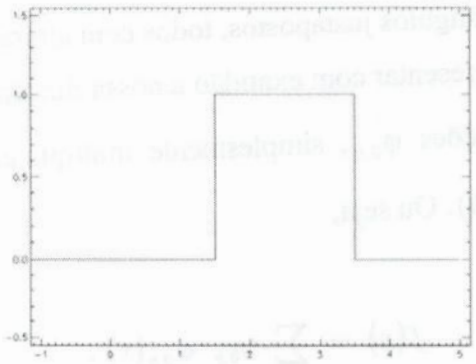
$a = 1, b = 0$



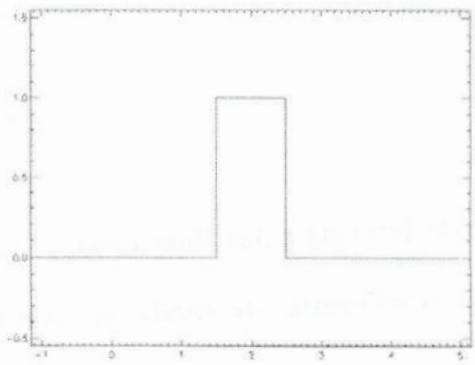
$a = 0, b = 1$



$a = 1, b = 1$



$a = 0, b = 2$



$a = 1, b = 2$

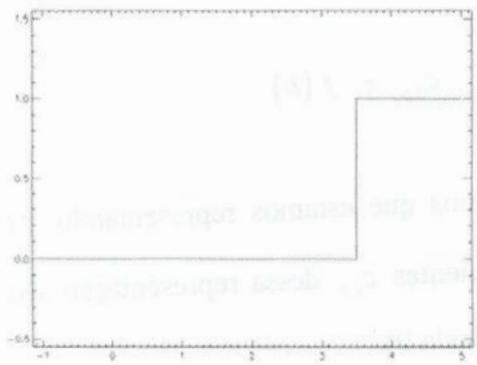


Figura 1.8: Funções filhas da função de escala. À esquerda, funções com a igual a 0, à direita a igual a 1. O valor de b é 0 para as funções em cima, 1 para as do meio e 2 para as de baixo.

Se tentarmos agora realizar a mesma operação, mas tendo como base as funções $\varphi_{1,b}$, iremos nos deparar com um pequeno problema: essas funções estão justapostas como as primeiras, mas sua largura é 2 e, portanto, estão deslocadas de inteiros pares, o que não nos permite representar exatamente $f(x)$ a partir dessa base. O melhor que podemos obter é uma representação aproximada, fazendo cada coeficiente de escala de nível 1 de índice b igual à média entre os valores de $f(2b)$ e $f(2b+1)$:

$$f(x) \cong \sum_b c_{1,b} \varphi_{1,b}(x) , \tag{1.21}$$

onde

$$c_{1,b} = \frac{f(2b)+f(2b+1)}{2} = \frac{c_{0,2b}+c_{0,2b+1}}{2} . \tag{1.22}$$

A função resultante dessa aproximação pode ser vista na figura 1.9. Note que o número de coeficientes em um certo intervalo de valores de x caiu pela metade, ao passo que a resolução da representação também foi degradada de um fator de dois. A representação exata da função pode ser obtida a partir desses coeficientes, se utilizarmos informações adicionais. Repetindo aproximadamente o que fizemos para a função de escala com a função discreta de Haar, obtemos:

$$\psi_{a,b}(x) = \psi\left(\frac{x}{a} - b\right), \tag{1.23}$$

de onde podemos verificar que $\psi(x) = \psi_{1,0}(x)$.

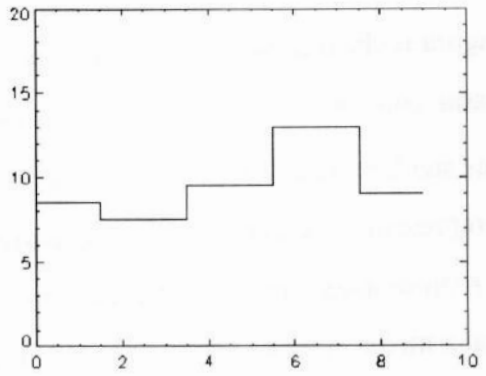


Figura 1.9: Versão suavizada da função discreta original obtida através de (1.21).

Agora podemos utilizar as funções $\psi_{1,b}$ como bases adicionais às $\varphi_{1,b}$ para obter uma representação exata de $f(x)$. A cada função $\psi_{1,b}$ associamos um coeficiente que deverá se igual à metade da diferença entre os valores $f(2b)$ e $f(2b+1)$:

$$f(x) = \sum_b c_{1,b}\,\varphi_{1,b}(x) + \sum_b w_{1,b}\,\psi_{1,b}(x)\,, \tag{1.24}$$

onde

$$w_{1,b} = \frac{f(2b) - f(2b+1)}{2} = \frac{c_{0,\,2b} - c_{0,\,2b+1}}{2}\,. \tag{1.25}$$

Esses coeficientes $w_{1,b}$ são denominados **coeficientes de wavelet** de nível 1 (ou escala 1).

O mesmo raciocínio pode ser aplicado às dilatações maiores das funções de Haar e de escala, de forma que obtemos representações exatas da imagem com $\psi_{1,b}$, $\psi_{2,b}$ e $\varphi_{2,b}$, ou ainda com $\psi_{1,b}$, $\psi_{2,b}$, $\psi_{3,b}$ e $\varphi_{3,b}$ etc. De uma forma geral, podemos escrever:

$$f(x) = \sum_{a=1}^{N} \left(\sum_b w_{a,b}\,\psi_{a,b}(x) \right) + \sum_b c_{N,b}\,\varphi_{N,b}(x)\,, \tag{1.26}$$

onde N é um número inteiro arbitrário,

$$c_{a+1,\,b} = \frac{c_{a,\,2b} + c_{a,\,2b+1}}{2}, \tag{1.27}$$

$$w_{a+1,\,b} = \frac{c_{a,\,2b} - c_{a,\,2b+1}}{2}, \tag{1.28}$$

e os coeficientes $c_{0,b}$ têm seus valores dados por (1.20).

Como visto, para obtermos os coeficientes de escala $c_{a+1,b}$, podemos tomar a média de coeficientes de escala $c_{a,b}$ consecutivos. Da mesma forma, para obtermos os coeficientes de wavelet $w_{a+1,b}$ tomamos metade da diferença dos coeficientes de escala $c_{a,b}$ consecutivos. Essas duas operações podem ser escritas como:

$$c_{a+1} = H\left[c_a\right] \tag{1.29}$$

e

$$w_{a+1} = G\left[c_a\right], \tag{1.30}$$

onde H e G representam, respectivamente, filtros associados à função de escala (média entre dois elementos) e à função discreta de wavelet (metade da diferença entre dois elementos)[1]. c_a e w_a representam conjuntos de coeficientes de escala e de wavelet, respectivamente, com mesmo um mesmo nível a, e podem ser eles mesmos tratados como funções discretas (do parâmetro de deslocamento b). Esses conjuntos são também chamados de **planos de escala** e **planos de wavelet**, respectivamente.

Dessa forma, podemos obter os planos de wavelet através da aplicação sucessiva de filtros aos coeficientes originais da imagem (contidos em c_0), como mostra a figura 1.10. O conjunto de todos os planos de wavelet obtidos é conhecido como **estrutura de wavelet**, ou ainda representação da função no **espaço de wavelet** (espaço que tem

[1] Para uma breve revisão sobre filtros digitais, veja o Apêndice A.

como bases as funções $\psi_{a,b}$). Essa estrutura nos fornece uma versão discreta da informação obtida com a transformada descrita em (1.4) e o processo de sua obtenção é a **transformada discreta de wavelet** pela técnica de Haar. Cada plano irá conter a informação pertinente às estruturas de uma certa escala de tamanho na função original. Do processo de decomposição resta ainda uma **função residual** c_N, que irá conter, para um valor de N suficientemente alto, apenas informação sobre o valor médio da função original. A partir dos planos de wavelet e dessa função residual, é possível obter a função original através da **transformada inversa de wavelet**, dada pela expressão (1.26).

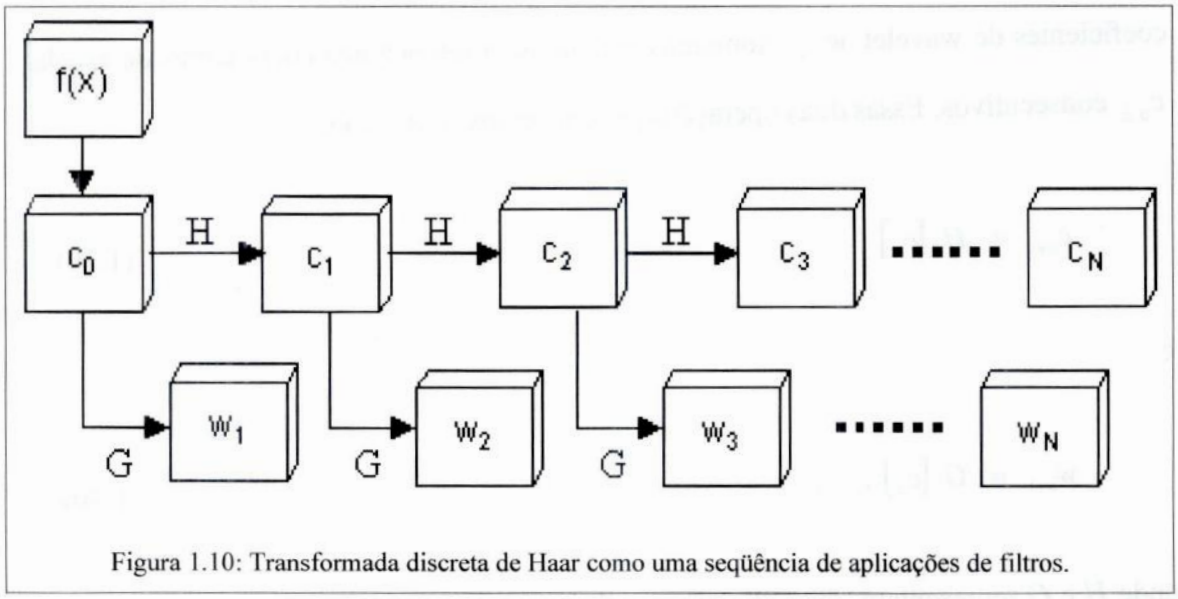


Figura 1.10: Transformada discreta de Haar como uma seqüência de aplicações de filtros.

A diminuição verificada do número de coeficientes com o aumento da escala é chamada **decimação**[1]. A forma mais natural de armazenamento dos coeficientes de uma transformada que envolve decimação, é chamada **piramidal**: os coeficientes da menor escala são representados por um vetor com o mesmo tamanho do domínio do sinal analisado, os da escala seguinte com metade desse tamanho, os da próxima com um quarto etc. Seja, por exemplo, uma função discreta definida no intervalo $x = [0\ ,\ 7]$, por exemplo, seus coeficientes de escala podem ser armazenados na forma

[1] Ou **dizimação**.

$$\begin{array}{cccccccc} (c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} & c_{0,4} & c_{0,5} & c_{0,6} & c_{0,7}) \\ & & (c_{1,0} & c_{1,1} & c_{1,2} & c_{1,3}) & & \\ & & & (c_{2,0} & c_{2,1}) & & & \\ & & & & (c_{3,0}) & & & \end{array} \tag{1.31}$$

e seus coeficientes de wavelet de forma semelhante.

Existe sempre a opção, no entanto, de manter as dimensões da função original ao longo de todo o espaço de coeficientes, no que chamamos **representação redundante**, o que facilita uma série de operações nos planos de wavelet em uma análise posterior, ao custo de mais espaço de armazenamento. Para o exemplo citado acima temos

$$\begin{array}{cccccccc} (c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} & c_{0,4} & c_{0,5} & c_{0,6} & c_{0,7}) \\ (c_{1,0} & c_{1,0} & c_{1,1} & c_{1,1} & c_{1,2} & c_{1,2} & c_{1,3} & c_{1,3}) \\ (c_{2,0} & c_{2,0} & c_{2,0} & c_{2,0} & c_{2,1} & c_{2,1} & c_{2,1} & c_{2,1}) \\ (c_{3,0} & c_{3,0} & c_{3,0} & c_{3,0} & c_{3,0} & c_{3,0} & c_{3,0} & c_{3,0}) \end{array} \tag{1.32}$$

para os coeficientes de escala e uma estrutura semelhante para os coeficientes de wavelet.

Uma outra forma de aplicação discreta da transformada de wavelet de desenvolvimento relativamente recente é a transformação *à trous*. Esta é muito semelhante à técnica discreta de Haar, mas possui alguns pontos inovadores que a tornam uma das melhores técnicas já desenvolvidas para a tarefa de discretização da transformada e a nossa escolha na elaboração dos nossos programas. Essa técnica é detalhada na próxima seção.

1.2.4 A técnica *à trous*

Com a técnica ***à trous***, o problema da transformação discreta pode ser visto como uma seqüência de aplicação de filtros, como na técnica de Haar, mas há duas diferenças importantes entre as duas: i) Os filtros utilizados com a técnica *à trous* são geralmente baseados em *splines* e ii) A composição da imagem (transformada inversa) a partir dos coeficientes é independente das formas da função discreta de wavelet e da

função de escala utilizadas (na verdade, não é preciso nem mesmo definir formalmente essas duas funções, cujas formas estarão vinculadas ao conjunto de filtros utilizados).

Para um entendimento mais fácil do procedimento, vamos iniciar com o conjunto mais simples possível de filtros, que chamamos **filtros de escala lineares**[1]. Definimos um filtro H_1 dado por

$$H_1 \longleftrightarrow \left(\frac{1}{4} \quad \frac{1}{2} \quad \frac{1}{4}\right), \tag{1.33}$$

segundo a notação esclarecida na seção sobre filtros digitais do Apêndice A. Sua aplicação a uma função discreta $f(x)$ pode ser escrita explicitamente como

$$H_1\,[f(x)] = \frac{1}{4} f(x+1) + \frac{1}{2} f(x) + \frac{1}{4} f(x-1)\,. \tag{1.34}$$

Podemos gerar uma série de outros filtros, H_2, H_3, etc., fazendo mais esparso[2] o vetor relacionado a H_1. A aplicação desses filtros será dada então por

$$H_n\,[f(x)] = \frac{1}{4} f\left(x+2^{n-1}\right) + \frac{1}{2} f(x) + \frac{1}{4} f\left(x-2^{n-1}\right). \tag{1.35}$$

O filtro G_1, por sua vez, é dado por:

$$G_1 \longleftrightarrow \left(-\frac{1}{4} \quad \frac{1}{2} \quad -\frac{1}{4}\right), \tag{1.36}$$

---

[1] Note que, embora o filtro seja, de fato, um filtro linear, seu nome nada tem a ver com este fato. A palavra "linear" no nome deste filtro refere-se ao comportamento de seus coeficientes ao se aproximarem do elemento central (coeficiente associado ao ponto $H_1(0,0)$ na função que representa o filtro).

[2] Esse procedimento é que é responsável pelo nome da técnica. A expressão francesa "*à trous*" significa, em português, "com buracos".

e, como fizemos para H_1, podemos definir uma família de filtros G_n de aplicação dada por

$$G_n\left[f(x)\right] = -\frac{1}{4}f\left(x+2^{n+1}\right) + \frac{1}{2}f(x) - \frac{1}{4}f\left(x-2^{n-1}\right). \tag{1.37}$$

Os filtros H_n são chamados de filtros **passa-baixas**, por suavizarem a função à qual são aplicados, eliminando as estruturas de alta freqüência e conservando as de baixa freqüência. Os filtros G_n são filtros **passa-altas**, pois acentuam detalhes, privilegiando altas freqüências em detrimento das baixas.

A partir dessas definições, pode-se extrair os planos de wavelet e a função residual de forma semelhante ao procedimento da técnica de Haar, como mostrado na figura 1.11, mas na verdade, existe um meio mais eficiente computacionalmente de se obter essas informações. A aplicação dos filtros G_n não é estritamente necessária para a geração dos coeficientes, já que, como pode ser visto facilmente, para uma função qualquer $y(x)$,

$$G_n\left[y(x)\right] = y(x) - H_n\left[y(x)\right]. \tag{1.38}$$

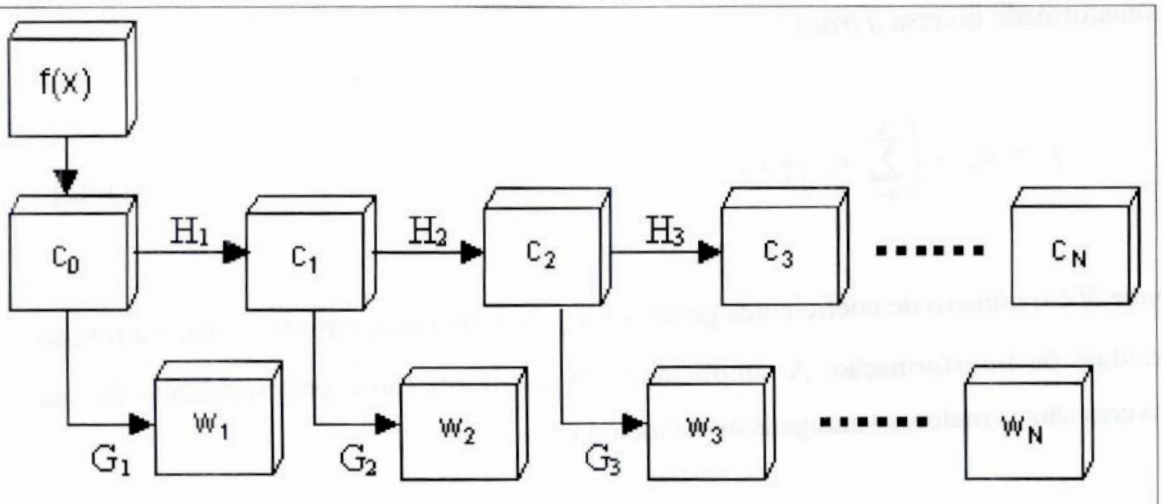


Figura 1.11: Transformada *à trous* como uma seqüência de aplicações de filtros.

Assim, podemos calcular os coeficientes de wavelet em dois passos. Primeiramente, calculamos os coeficientes de escala pela lei recursiva

$$c_{n+1}(x) = H_{n+1}\left[c_n(x)\right], \tag{1.39}$$

onde c_0 corresponde à própria função original, como na técnica de Haar:

$$c_0(x) = f(x) . \tag{1.40}$$

Cada função c_n obtida dessa forma é uma suavização da função original. A geração dessas imagens pode ser visualizada esquematicamente na figura 1.12. Podemos aplicar essa lei para gerarmos quantas funções suavizadas desejarmos. Então calculamos os planos de wavelet:

$$w_{n+1} = c_n - c_{n+1} . \tag{1.41}$$

Esse processo é a **transformada discreta de wavelet** pela técnica *à trous*, e a estrutura de wavelet (composta por todos os planos de wavelet) é a representação de $f(x)$ no espaço de wavelet. De (1.40) e (1.41), pode-se obter facilmente a fórmula da transformada inversa *à trous* :

$$f = c_0 = \left(\sum_{n=1}^{N} w_n\right) + c_N , \tag{1.42}$$

onde N é o número de coeficientes gerados e c_N, a última imagem suavizada, é a função residual da transformação. A simplicidade da transformação e principalmente da sua inversa são as maiores vantagens desse método.

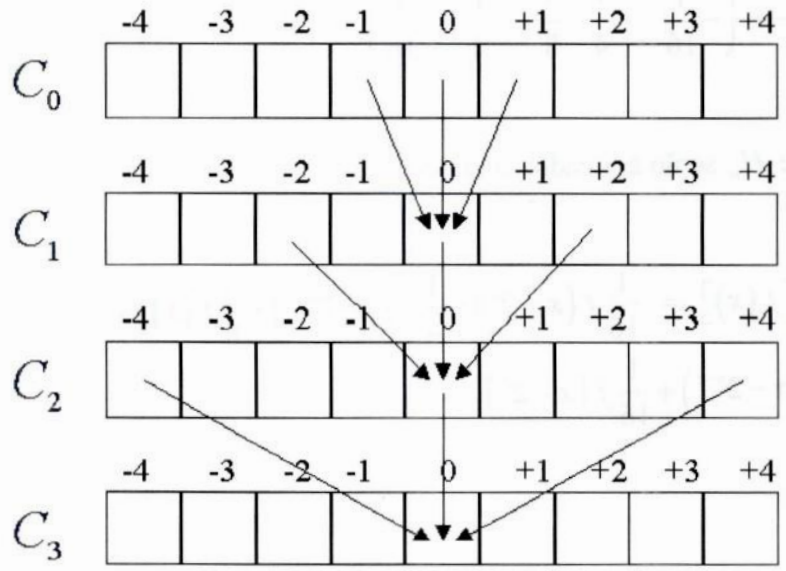


Figura 1.12: Geração de funções suavizadas a partir de filtros de escalas lineares na transformada *à trous*. Os valores dos pontos das imagens com um certo nível de suavização são obtidos a partir de frações dos valores dos pontos vizinhos à posição correspondente na imagem anterior. As frações são dadas por (1.35).

Os conjuntos de coeficientes de escala e os planos de wavelet assim gerados possuem o mesmo número de elementos que a função original e podem ser tratados como funções com o mesmo domínio de $f(x)$ o que torna esse método intrinsecamente redundante. Uma estrutura piramidal de armazenamento pode ser imposta através de uma subamostragem após cada aplicação de um filtro passa-baixas, mas implica muitos outros problemas adicionais, como uma dificuldade de inversão da transformada, por exemplo.

Uma outra escolha possível para o conjunto de filtros, em geral fornecendo resultados um pouco melhores (entre outras coisas, por proporcionar uma maior isotropia quando aplicado em duas dimensões) é dada pelo filtro ***spline*** $\mathbf{B_3}$:

$$H_1 \longleftrightarrow \left(\frac{1}{16} \quad \frac{1}{4} \quad \frac{3}{8} \quad \frac{1}{4} \quad \frac{1}{16}\right) \tag{1.43}$$

e o filtro G_1, ainda obedecendo a (1.38), será dado por

$$G_1 \longleftrightarrow \left(-\frac{1}{16} \quad -\frac{1}{4} \quad \frac{5}{8} \quad -\frac{1}{4} \quad -\frac{1}{16}\right). \tag{1.44}$$

Os filtros G_n e H_n serão aplicados segundo

$$H_n\left[f(x)\right] = \frac{1}{16}f\left(x+2^n\right)+\frac{1}{4}f\left(x+2^{n-1}\right)+\frac{3}{8}f(x)+ \frac{1}{4}f\left(x-2^{n-1}\right)+\frac{1}{16}f\left(x-2^n\right) \tag{1.45}$$

e

$$G_n\left[f(x)\right] = -\frac{1}{16}f\left(x+2^n\right)-\frac{1}{4}f\left(x+2^{n-1}\right)+\frac{5}{8}f(x)- \frac{1}{4}f\left(x-2^{n-1}\right)-\frac{1}{16}f\left(x-2^n\right) \tag{1.46}$$

1.3 Outras transformadas multiescalares

Existem outros tipos de transformadas multiescalares além da de wavelet. Como vimos na seção 1.2.4, na descrição da transformada *à trous*, a aplicação da transformada discreta de wavelet pode ser feita a partir de imagens suavizadas geradas através da aplicação de filtros lineares especiais H_n. Os planos de wavelet são gerados então através da subtração de suavizações consecutivas. Podemos substituir os filtros lineares utilizados pelo método *à trous* por outros filtros de suavização (ou redução de detalhes), não necessariamente lineares.

1.3.1 Mediana multiescalar

Seja um conjunto de um número ímpar n de quaisquer valores ordenáveis. Se associarmos a cada valor um índice entre 1 e n, de forma que o menor valor receba o índice 1, o segundo menor receba o índice 2 e assim por diante até o maior, que deverá

receber o índice n, a **mediana** desse conjunto será o valor do elemento de índice igual a $(n+1)/2$, ou seja, o elemento intermediário do conjunto.

É possível adotar um método para se calcular a mediana de conjuntos com número de elementos par, como o uso da média entre os valores centrais, de índices $n/2$ e $n/2+1$. Embora esse seja, provavelmente, o caminho mais adotado para o tratamento de conjuntos pares, as propriedades da mediana seriam mais bem conservadas se fosse tomado um dos dois valores centrais inalterados. A escolha de qual valor tomar poderia ser feita com base na proximidade do valor médio do grupo, por exemplo.

A aplicação de um **filtro mediano** corresponde à substituição do valor de cada ponto p na imagem original pela mediana dos pontos em uma região de largura (ou diâmetro[1]) arbitrária l em torno de p. Escreve-se

$$f_F(x) = FM_l[f(x)]. \tag{1.47}$$

Esse filtro possui muitas propriedades interessantes. O valor tomado para cada ponto é necessariamente o valor de um dos elementos da imagem original, de forma que se a imagem for composta de valores inteiros, por exemplo, então sua versão filtrada também o será. O filtro mediano costuma preservar muito bem os contornos dos objetos na imagem, ao contrário das convoluções, que tendem a tornar os objetos mais difusos. Além disso, estruturas de intensidade muito alta ou muito baixa e com poucos píxeis de tamanho, como alguns artifícios provenientes de falhas no CCD desaparecem completamente com a filtragem, como visto no exemplo visto na figura 1.13. Em imagens astronômicas, essa propriedade pode ser aproveitada para a remoção de raios cósmicos[2] também.

---

[1] O formato da região em sinais com mais do que uma dimensão é, em princípio, arbitrário, mas consideraremos, para todos os efeitos, o uso de uma região circular.

[2] Raios cósmicos são partículas de alta energia que, vindas do espaço, podem atravessar a atmosfera terrestre e estimular píxeis do CCD, causando o surgimento de pontos com altíssima contagem na imagem.

Figura 1.13: À esquerda, imagem com alguns defeitos em píxeis e colunas (introduzidos artificialmente). A mesma imagem processada com um filtro mediano com largura de 3 píxeis pode ser vista à direita.

A partir do filtro mediano, pode-se estabelecer uma transformada multiescalar discreta, de forma semelhante ao que é feito pela técnica *à trous*:

$$c_{n+1} = FM_{l=2n+3}[c_n] \tag{1.48}$$

e

$$w_{n+1} = c_n - c_{n+1}\,, \tag{1.49}$$

onde c_0 é a representação exata da imagem original e as funções (ou planos) w_n são conhecidos como planos multiescalares (uma generalização dos planos de wavelet). Essa transformada é chamada de **mediana multiescalar** (ou simplesmente **multimediana**, por alguns autores). A mediana multiescalar pertence a uma classe de transformações conhecidas como **transformações morfológicas multiescalares**, caracterizadas pelo emprego de filtros, no procedimento de transformação, que determinam para cada ponto na imagem final um valor presente na vizinhança do ponto correspondente na imagem processada, selecionado segundo algum critério especial, mas com seu valor inalterado (e, como vimos, o filtro mediano possui essa propriedade).

Da mesma forma que demonstrado para a transformação *à trous*,

$$f = c_0 = \left(\sum_{n=1}^{N} w_n \right) + c_N \tag{1.50}$$

nos fornece a transformação inversa.

A transformada mediana multiescalar tem como uma de suas principais vantagens a não propagação das estruturas de tamanho reduzido por todos os planos multiescalares. Outra grande vantagem é que não há criação de estruturas negativas nos coeficientes multiescalares, como acontece com a técnica *à trous*, exceto quando os objetos na imagem são eles mesmos negativos. Infelizmente, a não linearidade do filtro mediano complica muito certas operações envolvidas nas aplicações multiescalares comuns, principalmente no que concerne à propagação do ruído no espaço multiescalar (Bijaoui e Rué 1996).

1.3.2 Transformada MinMax

Outra transformada multiescalar pode ser obtida através dos operadores da **morfologia matemática**, filtros especiais que atuam diretamente na forma dos objetos:

- **operador de dilatação**: dá a cada ponto na imagem final o valor máximo da vizinhança imediata[1] do ponto correspondente na imagem processada.

- **operador de erosão**: dá a cada ponto na imagem final o valor mínimo da vizinhança imediata do ponto correspondente na imagem processada.

Seguimos os mesmos procedimentos utilizados na técnica *à trous* e na mediana multiescalar, utilizando como filtro suavizador de escala n a aplicação de n erosões seguida de n dilatações[2]. Assim, temos

---

[1] A definição do que seria a vizinhança do ponto para esse propósito é arbitrária, em princípio. Aqui consideramos um quadrado de 3 por 3 píxeis com o píxel em questão no centro.

[2] Essa operação é freqüentemente chamada **abertura** de ordem n.

$$c_{n+1} = D^{n+1}E^{n+1}\left[c_n\right], \tag{1.51}$$

$$w_{n+1} = c_n - c_{n+1} \tag{1.52}$$

e, como de costume,

$$f = c_0 = \left(\sum_{n=1}^{N} w_n\right) + c_N \tag{1.53}$$

Essa transformada é conhecida como **transformada MinMax** e, segundo a definição apresentada anteriormente, pertence à classe das transformadas morfológicas. Além de possuir algumas vantagens em comum com a transformada mediana, a MinMax possui a característica de eliminar completamente os coeficientes negativos, o que pode ser interessante em análises baseadas em estruturas sempre positivas. O mesmo problema da não-linearidade inerente à transformada mediana pode ser visto aqui. Essa transformada é aplicada a problemas de localização de aglomerações de pontos, como os de classificação de objetos a partir de um mapa de parâmetros físicos[1].

Essas duas transformadas multiescalares são apenas dois exemplos de transformadas não-wavelet possíveis. Seguindo essa mesma linha, é possível definir centenas de outras, partindo de diferentes filtros de suavização. Embora esses dois exemplos sejam de transformações redundantes, em ambos é possível impor uma representação piramidal. Algumas transformadas multiescalares que fogem completamente a esse modelo de aplicações de filtros também já foram definidas, mas uma investigação destas foge ao propósito deste documento.

---

[1] A classificação de asteróides em famílias a partir de suas propriedades observadas, por exemplo, pertence a essa classe de problemas.

1.4 OV_WAV: Um pacote IDL para a análise multiescalar

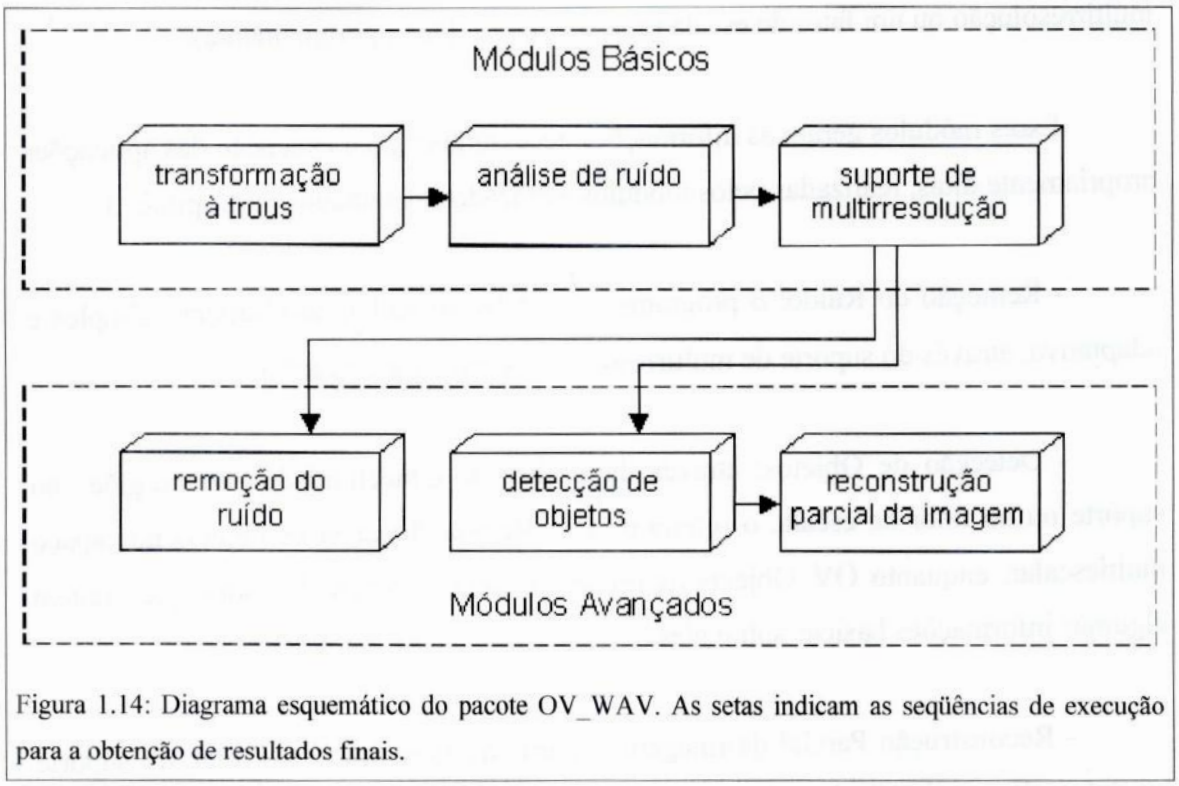


Figura 1.14: Diagrama esquemático do pacote OV_WAV. As setas indicam as seqüências de execução para a obtenção de resultados finais.

Nosso trabalho consistiu principalmente no desenvolvimento de uma série de programas computacionais para utilização em ambiente IDL[1] que possibilitassem aplicar a transformada de wavelet à análise multiescalar de vários problemas astronômicos. O pacote de programas, batizado como OV_WAV segue o esquema mostrado na figura 1.14.

Os módulos básicos, explicados em detalhe no Capítulo 2, são:

- Transformação *à trous*: inclui o programa OV_Atrous, responsável pela decomposição em planos de wavelet da imagem original.

- Análise de ruído: que contém os programas OV_Noise e OV_Anscombe, responsáveis, respectivamente, pela determinação do desvio padrão do ruído gaussiano em uma imagem e pela conversão de ruídos de Poisson ou misto em ruído gaussiano.

[1] A escolha do ambiente IDL será discutida mais adiante, na seção 2.1.

- Suporte de Multirresolução: com o programa OV_Support, que mapeia a significância de cada ponto no espaço de wavelet, sob a forma de um suporte de multirresolução ou um fator de escala (que serão definidos oportunamente).

Esses módulos geram as informações necessárias para a execução das aplicações propriamente ditas, realizadas pelos módulos avançados, detalhados no Capítulo 3:

- Remoção do Ruído: o programa OV_DeNoise realiza as filtragens simples e adaptativa, através do suporte de multirresolução ou do fator de escala.

- Detecção de Objetos: através da análise de conectividade entre regiões no suporte ou no fator de escala, o programa OV_Regions localiza os objetos no espaço multiescalar, enquanto OV_Objects os reúne em uma estrutura de dados que contém algumas informações básicas sobre eles.

- Reconstrução Parcial da Imagem: a partir da estrutura de wavelet, do suporte ou fator de escala, e dos dados gerados pela detecção de objetos, o programa OV_Reconstruct cria imagens de objetos isolados através de um método iterativo de inversão.

Existem ainda outros programas contidos no pacote, como várias subrotinas utilizadas pelos programas principais (algumas das quais podem ser úteis por si só) e ferramentas simples de visualização dos dados. No futuro o pacote deverá dispor ainda de programas que proporcionem uma interface gráfica entre os programas e o usuário e de alguns *scripts* pré-definidos para facilitar a execução de tarefas comuns. Essas perspectivas, além de outras, são discutidas no Capítulo 5 deste documento.

O desenvolvimento de nossos programas foi, em grande parte, baseado em um trabalho semelhante realizado por Starck, Murtagh e Bijaoui (1998), que originou o pacote de análise multiescalar MR. Esse pacote, no entanto, não está disponível de forma aberta à comunidade científica como o OV_WAV estará em breve. O pacote MR é de caráter bastante geral, com elementos de emprego específico em várias áreas além da astronomia, enquanto no OV_WAV todos os procedimentos são adaptados e otimizados para o uso em imagens astronômicas. Além disso, algumas técnicas sofisticadas utilizadas por nossos programas não estão disponíveis no MR.

2 Módulos Básicos

As primeiras pedras de um edifício

Neste capítulo discorremos sobre as etapas iniciais do projeto, discutindo sobre a escolha da plataforma IDL como ambiente de desenvolvimento e detalhando os módulos básicos, que também foram os primeiros a serem desenvolvidos. Esses módulos tratam, basicamente, da criação de informações e estruturas de dados utilizadas por outros módulos.

2.1 O ambiente de programação

Antes que o desenvolvimento do pacote fosse iniciado, era necessário determinar em que ambiente esse desenvolvimento seria realizado. A utilização de uma linguagem de programação tradicional, como C, C++ ou FORTRAN, foi considerada a princípio, mas, além dos bem conhecidos problemas de portabilidade dessas linguagens (apesar de todo o esforço pela padronização) haveria uma grande perda de tempo com o desenvolvimento de sub-rotinas de baixo nível para alocação de memória, exibição de imagens etc.

O ambiente IDL (*Interactive Data Language*) desenvolvido pela RSI (*Research Systems Inc.*), que tem crescido muito em popularidade entre os pesquisadores de diversas áreas, se mostrou especialmente atraente para a tarefa, por trabalhar com código extremamente portável (com um mínimo de cuidado, pode-se produzir programas utilizáveis sem modificações em Windows, MacOS, VMS, Unix e Linux) e por incorporar uma série de rotinas direcionadas à visualização e manipulação de imagens. A produção de interfaces gráficas (GUIs)[1] portáveis também é incrivelmente facilitada. As duas principais desvantagens da IDL são o fato de o código ser interpretado, o que reduz um pouco a velocidade de execução (embora isso possa ser

[1] GUI é sigla de "*Graphical User Interface*" – "interface gráfica com o usuário".

amenizado através do uso inteligente de construções especiais incorporadas à linguagem) e a não-gratuidade do interpretador, o que força o usuário a adquirir uma cópia do programa junto ao fabricante.

A linguagem IDL é altamente adaptada à manipulação de dados em grandes estruturas, como imagens. Nada melhor que um pequeno exemplo para ilustrar essa qualidade da linguagem. Temos duas imagens, *A* e *B*, com o mesmo tamanho, sendo *x* sua largura e *y* sua altura. Desejamos somar ponto-a-ponto essas imagens em uma nova imagem *C*. Queremos ainda que a soma não seja feita nos pontos em que *B* seja negativo; nesses casos, o resultado deve ter o valor de *A* naquela posição. Um algoritmo para tal façanha na linguagem C, por exemplo, pode ser escrito como[1]:

```
for (i = 0; i < x; i++)
for (j = 0; j < y; j++)
{ if (B[i][j] >= 0) C[i][j] = A[i][j] + B[i][j];
  else C[i][j] = A[i][j]; }
```

Outra forma um pouco mais compacta possível em C é:

```
for (i = 0; i < x; i++)
for (j = 0; j < y; j++)
C[i][j] = (B[i][j] >= 0) ? (A[i][j] + B[i][j]) : (A[i][j]);
```

Podemos escrever um código semelhante em IDL:

```
for i = 0, x - 1 do $
for j = 0, y - 1 do $
if B[i,j] GE 0 then C[i,j] = A[i,j] + B[i,j] $
else C[i,j] = A[i,j]
```

Mas a IDL nos dá a opção de escrevermos simplesmente:

```
C = A + (B > 0)
```

---

[1] A linguagem C foi escolhida para esse exemplo por conter as estruturas de código presentes em praticamente todas as linguagens tradicionais. Um programador com uma certa experiência em qualquer dessas linguagens não deve ter muito problema em compreender o procedimento seguido.

O operador ">" não possui em IDL o significado de "maior que". Ele significa "tome o maior entre esses dois (para cada ponto)". Assim, o resultado de "*(B > 0)*" terá o valor de *B* nos pontos em que este for maior que zero e terá o valor zero nos outros pontos. A soma dessa matriz com *A* será feita ponto-a-ponto automaticamente, de forma que *C* terá, para os pontos em que *B* seja maior que zero, o valor de $A + B$, e, quando *B* for negativo, terá o valor de $A + 0 = A$, que é exatamente o que queríamos.

Como vimos, IDL é uma linguagem que permite realizar operações com imagens de forma mais simples que o usual. Por outro lado, a programação eficiente em IDL requer algumas estratégias que não são naturais a outras linguagens em geral. As duas construções mostradas acima, ao contrário do que se possa pensar a princípio, não são executadas com a mesma performance. Como IDL é uma linguagem interpretada, a utilização de *loops* e iterações retarda em muito o processamento, pois requer a decodificação em tempo real de uma grande quantidade de procedimentos. A segunda estratégia deixa apenas uma linha de código a ser interpretada, deixando todo o trabalho de iteração por conta da biblioteca interna e pré-compilada do ambiente.

2.2 Transformação *à trous*

Dentre todas as possibilidades de transformadas multiescalares disponíveis, a transformação *à trous*, introduzida na subseção 1.2.4, foi selecionada como base para nossos programas. Como lidamos com problemas relacionados a imagens bidimensionais, precisamos de uma versão bidimensional da técnica. Nesse caso, definimos H_1 através de uma matriz, e não de um vetor, como fizemos anteriormente. Aqui a escolha exata do filtro a ser utilizado continua em aberto, mas as duas escolhas mais naturais são:

$$H_1 \longleftrightarrow \begin{pmatrix} 1/4 \\ 1/2 \\ 1/4 \end{pmatrix} \times \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{pmatrix}, \tag{2.1}$$

baseado no filtro de escala linear, e

$$H_1 \longleftrightarrow \begin{pmatrix} 1/16 \\ 1/4 \\ 3/8 \\ 1/4 \\ 1/16 \end{pmatrix} \times \begin{pmatrix} \frac{1}{16} & \frac{1}{4} & \frac{3}{8} & \frac{1}{4} & \frac{1}{16} \end{pmatrix}, \tag{2.2}$$

baseado no filtro *spline* B_3. Esses filtros são, por construção, separáveis (segundo definição dada no Apêndice A). Os filtros subseqüentes H_2, H_3 etc. são gerados de forma similar, a partir de versões esparsas do vetor que origina H_1. Por sua separabilidade, no entanto, é possível otimizar o processo de aplicação desses filtros, fazendo aplicações das versões unidimensionais mostradas na subseção 1.2.4 a cada linha da imagem, gerando uma imagem intermediária, na qual se aplicam os mesmos filtros a cada coluna, gerando a imagem filtrada. O resultado é idêntico a uma aplicação bidimensional direta (usando os filtros bidimensionais dados acima e a expressão (A.6)), mas o processo toma muito menos tempo de processamento.

2.2.1 Vantagens da técnica *à trous* para imagens astronômicas

A transformação *à trous* apresenta diversas vantagens que a tornam ideal para o processamento de imagens astronômicas modernas:

1. Imagens astronômicas modernas são de natureza digital, de forma que o uso de transformadas discretas, como a *à trous* se torna mais adequado. Com isso se evita uma quantidade não desprezível de erros introduzidos ao se discretizar funções analisadoras contínuas.

2. A natureza dos objetos e funções de espalhamento mais comuns em astronomia é isotrópica, o que torna inadequados os tratamentos anisotrópicos

fornecidos pela transformada de Mallat e pela técnica discreta de Haar[1]. A transformação *à trous* é altamente isotrópica, especialmente se o filtro *spline* B_3 for empregado.

3. A transformação *à trous*, como qualquer transformada verdadeira de wavelet, é uma operação linear, de forma que se ruído de natureza aditiva estiver presente na imagem original, por exemplo, sua representação no espaço multiescalar será também aditiva. Essa propriedade, que não é encontrada nas transformadas morfológicas, por exemplo, torna possível o estudo de imagens em que a presença do ruído é significativa, como as astronômicas.

4. Ao contrário do que ocorre com muitos modelos de discretização da transformada de wavelet, a técnica *à trous* não só é simples de ser implementada, como sua transformada inversa também o é. A transformada *à trous* inversa pode ser obtida de forma exata, o que também não é comum a todas as técnicas discretas de wavelet.

2.2.2 Sobre a notação

Como a transformação *à trous* foi a única técnica de análise multiescalar utilizada no desenvolvimento de todos os módulos que descrevemos, a partir daqui, a notação

$$w = W[f(x,y)] \tag{2.3}$$

significará que w é a estrutura de wavelet (que é o conjunto dos planos de wavelet) obtida através da aplicação da transformação *à trous* à imagem $f(x,y)$. Como $f(x,y)$ possui a mesma informação contida em c_0, as consideraremos idênticas. Da mesma forma, indicaremos a transformada inversa por

---

[1] Com a análise de Mallat são estudadas de forma independente as estruturas de orientações horizontal, vertical e diagonal. A transformada discreta de Haar (assim como a contínua) introduz uma anisotropia na análise através da não paridade do próprio wavelet utilizado.

$$f(x,y) = W^{-1}[w] = \left(\sum_{n=1}^{N} w_n\right) + c_N . \tag{2.4}$$

Note que embora possamos dizer que

$$W^{-1}W = 1 \tag{2.5}$$

ou seja,

$$W^{-1}W[f] = W^{-1}[W[f]] = f , \tag{2.6}$$

para uma imagem qualquer *f*, não podemos dizer, em geral, que

$$W W^{-1} = 1 \tag{2.7}$$

para um conjunto de imagens *w* qualquer. Isso porque a transformada *à trous* inversa (2.4) é a inversa também de várias outras operações (como visto na seção 1.3, por exemplo). A expressão acima só é válida se *w* for realmente a transformação *à trous* de uma imagem.

2.2.3 Implementação da transformação *à trous*

A transformação *à trous* precisa ser realizada antes de todos os outros programas do pacote, independentemente de qual o tipo de problema tratado. O programa **OV_Atrous**, responsável pela tarefa, foi construído de forma a tirar o máximo proveito possível dos recursos de operação de matrizes da IDL, alcançando uma performance satisfatória. A chamada básica ao programa é feita através da linha de comando IDL:

```
wv = ov_atrous(data, n)
```

onde `data` é uma matriz contendo a imagem original, `n` é o número de panos de wavelet a serem gerados e `wv` irá conter uma matriz de três dimensões, correspondente à

estrutura de wavelet calculada (a terceira dimensão será a escala, ou nível, do plano de wavelet). Um exemplo de aplicação é mostrado na figura 2.1.

A aplicação dos filtros passa-baixas é feita de forma separada a linhas e colunas para uma melhor performance.

O programa permite optar entre o filtro de escala linear e o *spline* B_3 para as operações de suavização através de parâmetros opcionais de entrada. Se for requisitado, o programa retorna, além da estrutura de wavelet, a imagem residual do processo.

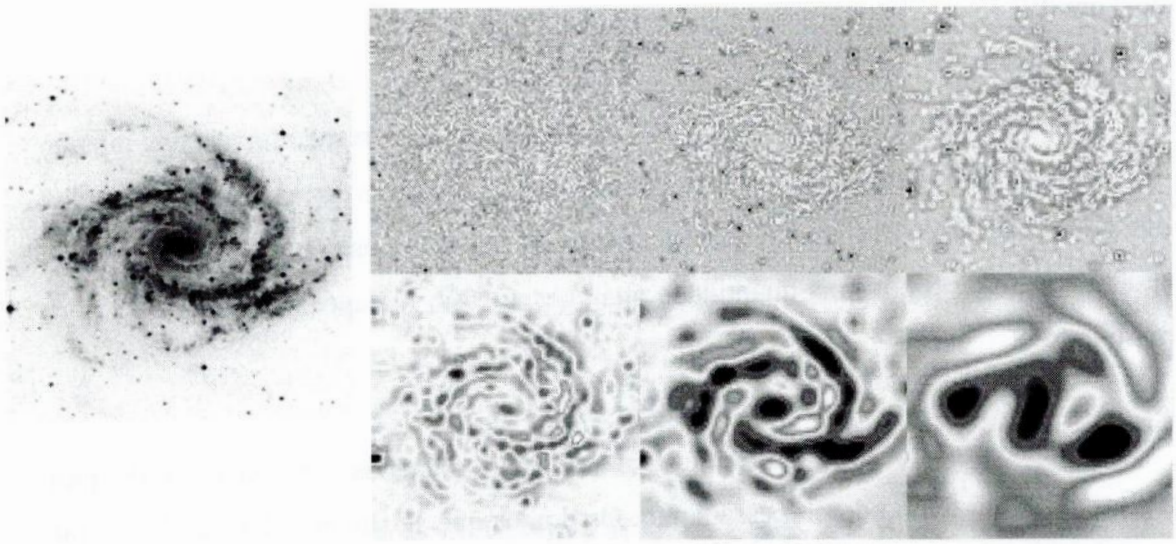


Figura 2.1: Exemplo de aplicação do programa OV_Atrous a uma imagem astronômica. À esquerda uma imagem da galáxia NGC 2997. À direita, ordenados da esquerda para a direita e de cima para baixo os 6 primeiros planos de wavelet obtidos com o programa, com filtro de escala linear. A escala de tons foi escolhida de forma a facilitar a visualização das estruturas.

Um tópico importante na implementação de filtros lineares é como substituir o valor dos pontos que se situam fora da imagem. Como o valor de um dado ponto na imagem filtrada depende dos pontos em uma certa vizinhança do ponto correspondente na imagem original, pontos próximos às bordas da imagem podem depender do valor de pontos que, na realidade, não estão definidos, situando-se além das bordas da imagem. Os métodos que pretendem lidar com esse problema são conhecidos como **métodos de tratamento de bordas**, e em geral, consistem em remapear os pontos que se situam fora da imagem em novas posições no interior da imagem. Apresentaremos as versões

unidimensionais[1] de três dos mais comumente utilizados, implementadas na sub-rotina **OV_Border**, invocada pelo programa OV_Atrous. O usuário pode informar ao programa OV_Atrous qual desses métodos deve ser usado nas filtragens passa-baixas realizadas. Nas explicações a seguir, assumimos que a função discreta analisada está definida no intervalo $[0, l]$:

1. **Método da continuidade**. O mais rápido computacionalmente. É feito simplesmente remapeando os pontos situados além de l em l, e aqueles aquém de 0 em 0.

2. **Método da periodicidade**. Em geral produz bons resultados apenas quando as propriedades da função são bastante homogêneas. Os pontos que se situam além de l por uma quantidade Δx qualquer, são remapeados para a posição $0 + \Delta x$. Os que se situam aquém de 0 por uma quantidade qualquer Δx são remapeados para $l - \Delta x$, por outro lado[2].

3. **Método do espelhamento**. Um dos mais robustos, entretanto o de pior performance em tempo de processamento. Os pontos que se situam além de l por uma quantidade Δx qualquer, são remapeados para a posição $l - \Delta x$. Os que se situam aquém de 0 por uma quantidade qualquer Δx são remapeados para $0 + \Delta x$, por sua vez[3].

---

[1] Na verdade, como a filtragem é realizada de forma separada em linhas e colunas pelo programa OV_Atrous, o tratamento de bordas é feito, efetivamente, em uma dimensão. De qualquer forma, as generalizações desses métodos para um número qualquer de dimensões não são difíceis de se obter, bastando aplicá-los separadamente a cada dimensão.

[2] Na verdade, como pode ser visto, caso o ponto se situe a uma distância muito grande da borda, esse remapeamento não será suficiente para que deixe de estar fora do domínio da função. Caso queiramos nos precaver contra esse tipo de problema (como na verdade a sub-rotina OV_Border o faz), devemos associar o novo valor da coordenada a uma função periódica do valor original. Nesse caso será uma função do tipo "dente de serra", com período igual a $l + 1$, com mínimo em 0 e máximo em l.

[3] Aqui vale a mesma observação feita na nota anterior. Nesse caso, a função periódica de remapeamento deverá ser uma onda triangular com período $2(l + 1)$, com mínimo em 0 e máximo em l.

2.3 Análise de ruído

A imagem astronômica é, em geral, fortemente contaminada por componentes aleatórias, de pequena escala, e não relacionadas aos objetos físicos observados, globalmente conhecidas como ruído. Qualquer aplicação computacional um pouco ambiciosa em processamento ou análise de imagens astronômicas deve envolver procedimentos para a determinação, com alguma profundidade, da natureza do ruído na imagem estudada, se se dispõe a obter resultados de boa qualidade. O estudo do ruído na imagem é uma etapa preliminar indispensável a todas as aplicações incluídas em nossos programas.

2.3.1 Natureza do ruído em imagens astronômicas

O ruído na imagem astronômica (como em dados de praticamente qualquer natureza), não é causado por um único fenômeno físico, mas pela adição das pequenas contribuições de inúmeros mecanismos.

Em geral, admite-se que o ruído em uma imagem astronômica típica é dominado por uma componente que obedece a uma densidade de probabilidades de Poisson. Em muitos casos podemos, sem introduzir grandes erros em nossas análises, considerar que esta é a única componente existente.

Essa modelagem do ruído torna-se extremamente ineficaz, no entanto, se a imagem tiver sido obtida em condições em que o ruído de leitura seja significativamente grande. O ruído de leitura introduz uma componente aleatória com distribuição aproximadamente gaussiana de probabilidades. Nesse caso, pode-se adotar um modelo mais refinado, em que a distribuição da componente de ruído é dada pela soma de uma variável aleatória com densidade de probabilidade de Poisson e uma outra de densidade gaussiana.

Pode-se adotar ainda um modelo de ruído puramente gaussiano, o que em geral é feito para que se tome proveito de propriedades especiais dessa distribuição, embora não seja uma aproximação muito boa do comportamento verdadeiro do ruído. Por outro lado, o modelo gaussiano é, em geral, a opção que produz melhores resultados na análise de dados em que o ruído possui comportamento desconhecido ou não usual, não se adequando aos modelos descritos acima.

Um desses três modelos deve ser adotado para o estudo de cada imagem. Exploraremos primeiramente o último, já que sua análise é a mais simples e servirá como base para a descrição dos demais.

2.3.2 Estimativa de parâmetros de ruído gaussiano

A distribuição gaussiana está definida no Apêndice B. Quando consideramos que uma imagem possui ruído gaussiano, estamos admitindo que o valor de cada píxel da imagem é uma variável aleatória de distribuição gaussiana, com média igual ao valor verdadeiro daquele píxel (valor que possuiria se o ruído não estivesse presente) e desvio padrão desconhecido a princípio. Na maioria dos casos, podemos considerar que o ruído é homogêneo, ou seja, que esse desvio padrão é o mesmo em cada ponto da imagem. Nesse caso, só nos resta descobrir o valor desse parâmetro, que denotaremos σ_I, para que a natureza do ruído seja completamente conhecida.

A transformada *à trous* nos fornece um método iterativo bastante preciso para a obtenção dessa grandeza:

1. Obtém-se a transformada de wavelet da imagem a ser analisada:

$$w = W[f(x,y)] \tag{2.8}$$

2. Como o ruído deve predominar no primeiro plano de wavelet (que contém estruturas de dimensões da ordem de 1 píxel), tomamos o desvio padrão dos pontos dessa escala[1]:

$$\sigma_1 = \sigma(w_1) \tag{2.9}$$

---

[1] É interessante notar que, como dito no Apêndice B, uma transformação linear de uma variável de distribuição gaussiana irá gerar uma outra variável também de distribuição gaussiana. Como a técnica *à trous* aplica apenas filtros lineares para a obtenção dos planos de wavelet, se uma imagem possui ruído gaussiano puro, então os seus planos de wavelet também o possuirão (embora certamente com parâmetros diferentes, como veremos).

3. Se houver pontos no primeiro plano de wavelet acima de um certo número de vezes (usualmente 3) o desvio padrão calculado[1], então volta para o passo 2, desconsiderando agora esses pontos (eles podem conter informação de baixa escala referente a algum objeto real), a não ser que esteja satisfeito algum outro critério de parada.

Assim obtemos o desvio padrão induzido pelo ruído no plano de wavelet de menor escala, mas esse não é igual ao desvio padrão do ruído na imagem original. Pela propriedade de linearidade da transformada de wavelet, a razão entre o desvio padrão do ruído no primeiro plano de wavelet e o da imagem original deve ser uma constante, dependendo somente do wavelet empregado na transformação. Assim, podemos realizar simulações com imagens artificiais contendo apenas ruído gaussiano de desvio padrão igual a 1 e obter o desvio padrão em cada escala, sendo que o valor obtido para a primeira será a constante que procuramos. Realizamos essas simulações para os 9 primeiros planos e obtivemos, para a técnica *à trous* com filtros de escala linear e *spline* B_3, os valores mostrados na tabela 2.1:

Razão	Filtro de escala linear	Filtro *spline* B_3
σ_1/σ_I	0.7984	0.9103
σ_2/σ_I	0.2731	0.2647
σ_3/σ_I	0.1204	0.0937
σ_4/σ_I	0.0588	0.0462
σ_5/σ_I	0.0300	0.0205
σ_6/σ_I	0.0160	0.0107
σ_7/σ_I	0.0091	0.0061
σ_8/σ_I	0.0054	0.0042
σ_9/σ_I	0.0031	0.0019

Tabela 2.1: Fatores de conversão de desvios-padrão

[1] Essa técnica de corte em um nível igual a um certo número de vezes o desvio padrão é conhecida como ***sigma clipping***.

Assim, se utilizamos o filtro de escala linear, temos

$$\sigma_I = \frac{\sigma_1}{0{,}7984} , \tag{2.10}$$

e se o filtro escolhido foi o *spline* B_3, então

$$\sigma_I = \frac{\sigma_1}{0{,}9103} . \tag{2.11}$$

Dessa forma temos, não somente o desvio padrão do ruído na imagem, como também em todos os planos de wavelet. Esses valores nos serão muito úteis para a análise de significância dos pontos no espaço de wavelet, detalhada na seção 2.4.

2.3.3 Transformada de Anscombe e ruído de Poisson

Caso o ruído obedeça a uma função densidade de probabilidade de Poisson (definida no Apêndice B) , não podemos realizar uma análise simples como a mostrada para o caso gaussiano. Uma transformação linear de uma variável poissoniana não gera outra variável poissoniana. Além disso, o desvio padrão σ_I e a média μ da variável aleatória que fornece o valor registrado em cada píxel da imagem estarão vinculados por

$$\mu = \sigma_I^2 \tag{2.12}$$

Como a média corresponde ao valor verdadeiro do píxel e esperamos que esse valor varie ao longo da imagem, o desvio padrão irá variar ponto a ponto.

Uma forma de estudar o ruído de uma imagem desse tipo é através da **transformada de Anscombe**:

$$A\left[f(x,y)\right] = 2\sqrt{f(x,y)+\frac{3}{8}} , \tag{2.13}$$

onde $f(x,y)$ é a imagem original com ruído de Poisson. A imagem transformada irá conter ruído Gaussiano puro e homogêneo, com desvio padrão igual a 1. Isso só será válido, no entanto, se o valor de cada píxel na imagem original for maior que cerca de 30 contagens, como mostrado por Starck, Murtagh e Bijaoui (1998). Essa limitação não representa nenhum problema se lidamos com imagens tomadas na região visível do espectro de radiação, onde geralmente se obtém contagens muito maiores. Para observações em raios-x, por exemplo, esse requisito dificilmente é satisfeito, sendo necessário adotar algum outro procedimento[1].

Podemos utilizar a transformada de Anscombe da imagem para qualquer procedimento que dependa da natureza do ruído, incluindo o estudo de significância feito na seção 2.4.

Todos os detalhes do estudo de imagens com ruído gaussiano, incluindo os resultados dados na tabela 2.1 podem ser aplicados a essa imagem transformada, mas não será necessário determinar o desvio padrão do ruído, que já é conhecido é, por definição, unitário.

2.3.4 Tratamento de ruído misto

Quando consideramos que o ruído é a soma de uma parcela de natureza poissoniana e de uma parcela gaussiana (devida, em geral, ao ruído de leitura de um CCD), precisamos recorrer a uma forma modificada da transformada de Anscombe (Starck, Murtagh e Bijaoui 1998):

$$A'\left[f(x,y)\right] = \frac{2}{g}\sqrt{g\cdot f(x,y)+\frac{3}{8}g^2+\sigma_G^2-g\mu_G}\ , \qquad (2.14)$$

onde g é o ganho de aquisição dos dados, σ_G é o desvio padrão do ruído de leitura e μ_G a média da componente gaussiana, em geral associada ao nível zero do conversor analógico-digital do sistema de leitura do CCD. Essa operação é conhecida como **transformada generalizada de Anscombe** ou como **transformada de estabilização**

[1] Embora não nos ocupemos em explorar esse problema aqui, algumas soluções podem ser vistas em Starck, Murtagh e Bijaoui (1998).

de variância. Um dos maiores problemas desse tratamento é que os parâmetros g, σ_G e μ_G teriam que ser conhecidos a priori para a imagem processada. Felizmente existe uma forma iterativa para se estimar g ou σ_G se apenas um deles não for conhecido (μ_G, no entanto, ainda precisa ser conhecida), o que ameniza o problema. Por exemplo, se σ_G for desconhecido:

1. Admite-se que σ_G esteja entre 0 e o desvio padrão total da imagem, $\sigma(f)$. Toma-se como estimativa inicial para σ_G o valor intermediário a esses dois:

$$\sigma_{\min_0} = 0 \,, \tag{2.15}$$

$$\sigma_{\max_0} = \sigma(f) \,, \tag{2.16}$$

$$\sigma_{G_0} = \frac{\sigma_{\max_0} + \sigma_{\min_0}}{2} \,. \tag{2.17}$$

2. Aplica-se a equação (2.14) à imagem, com o uso da estimativa atual de σ_G.

3. Calcula-se o desvio padrão do ruído na imagem transformada através do processo descrito na subseção 3.2.3. Se esse valor for menor que 1, então

$$\sigma_{\max_{n+1}} = \sigma_{G_n} \tag{2.18}$$

e

$$\sigma_{\min_{n+1}} = \sigma_{\min_n} \,, \tag{2.19}$$

e se for maior que 1, então

$$\sigma_{\min_{n+1}} = \sigma_{G_n} \tag{2.20}$$

e

$$\sigma_{\max_{n+1}} = \sigma_{\max_n} . \tag{2.21}$$

A nova estimativa para σ_G será

$$\sigma_{G_{n+1}} = \frac{\sigma_{\max_{n+1}} + \sigma_{\min_{n+1}}}{2} . \tag{2.22}$$

4. Se for satisfeito algum critério de parada (baseado, por exemplo, no valor de $\sigma_{\max_{n+1}} - \sigma_{\min_{n+1}}$), temos nossa estimativa para σ_G. Caso contrário, volta-se para o passo 2.

Um algoritmo muito semelhante poderia ser utilizado para que se obtivesse uma estimativa de g com σ_G e μ_G conhecidos.

As observações feitas sobre a transformada de Anscombe na subseção anterior se aplicam também à transformada generalizada.

2.3.5 Implementação da análise de ruído

A estimativa do desvio padrão do ruído gaussiano em uma imagem é realizada pelo programa **OV_Noise**, pelo método iterativo explicado na subseção 2.3.2. A chamada ao programa pode ser feita de duas formas, a primeira é

```
std = ov_noise(data)
```

onde `data` é a imagem original. A segunda é

```
std = ov_noise(wv)
```

onde `wv` é a estrutura de wavelet da imagem. Em ambos os casos, `std` irá conter o desvio padrão calculado para a imagem como um valor de ponto flutuante. Caso a primeira forma seja utilizada, o programa OV_Atrous será utilizado internamente para a extração do primeiro plano de wavelet, usado no processo iterativo de estimativa do ruído. Se essa informação já tiver sido gerada previamente, então a segunda forma deve ser usada, de forma a reduzir o tempo de execução.

O fator de conversão mostrado na tabela 2.1 é obtido através da sub-rotina **OV_Table**, que simplesmente retorna os fatores adequados para o filtro e o número de planos desejados. Dois critérios de parada estão disponíveis: 1) a não alteração, ou a alteração por um valor máximo especificado, da estimativa do desvio padrão entre duas iterações, e 2) a execução de um número máximo de iterações. O número de vezes o desvio padrão a ser utilizado a cada corte pode ser especificado também, sendo 3 o valor padrão, já que garante uma convergência correta do algoritmo (valores maiores podem fornecer resultados menos acurados, enquanto valores menores podem provocar uma divergência do algoritmo).

A aplicação tanto da transformada de Anscombe quanto de sua versão generalizada, para tratamento de ruídos de Poisson e misto, respectivamente, é feita pelo programa **OV_Anscombe**, através da chamada:

```
adata = ov_anscombe(data)
```

onde `data` é a imagem original e `adata` é a imagem transformada. A transformada generalizada será utilizada se forem, adicionalmente, especificados os valores de g, σ_G e μ_G. Se σ_G ou g não for conhecido (no caso generalizado), o programa pode ainda estimar seu valor segundo o método descrito na subseção 2.3.4.

2.4 Suporte de multirresolução

O **nível de significância**, ou **nível de validade**, de um ponto no espaço de wavelet é uma medida da confiabilidade de que a informação ali contida seja devida ao

sinal verdadeiro (e não ao ruído). Em geral, se atribui a cada ponto no espaço de wavelet um nível de significância que pode ir de 0 a 1, sendo valores maiores indicadores de uma contaminação menor pelo ruído. O conjunto desses valores forma uma estrutura que possui as mesmas dimensões da transformada de wavelet da imagem analisada, e que é denominada **fator de escala** (nome atribuído por Brown, 2000). Esses níveis podem ser obtidos por uma série de métodos diferentes, cada um apresentando vantagens e desvantagens. Alguns desses métodos (um dos quais veremos na próxima subseção) associam um valor binário a cada ponto, dando origem a uma classificação lógica em 1 (significante) ou 0 (não significante). Um fator de escala binário gerado dessa forma recebe o nome de **suporte de multirresolução**. Pelo fato de tradicionalmente esse ser o tipo mais usado de fator de escala, freqüentemente utilizamos o termo "suporte de multirresolução" ou simplesmente "suporte", mesmo quando se admite um fator de escala não binário.

Em geral, os métodos utilizados para a geração de um suporte ou fator de escala eficiente são adaptados de métodos tradicionais de filtragem para eliminação de ruído. Isso é muito compreensível se levarmos em conta que uma das aplicações mais imediatas do suporte de multirresolução é a remoção de ruído no espaço de wavelet, que veremos em detalhe na seção 3.1 (embora essa estrutura seja empregada para muitos fins além da remoção de ruído por nossos programas).

2.4.1 *Hard thresholding*

O método mais simples e, de longe, o mais popular para o cálculo do suporte, o ***hard thresholding*** é baseado na técnica homônima de filtragem de ruído, dada pela expressão:

$$v_F = \begin{cases} v; & |v| \geq T \\ 0; & |v| < T \end{cases} \tag{2.23}$$

onde v é o valor de um dado ponto contido na imagem ruidosa (ou em qualquer outra forma de dados que se queira tratar), v_F é o valor filtrado correspondente, e T é um valor, chamado *threshold*, correspondente ao valor mínimo que um ponto precisa ter

(em módulo) para ser considerado significante. Podemos ver graficamente a relação entre os valores originais e os filtrados na figura 2.2.

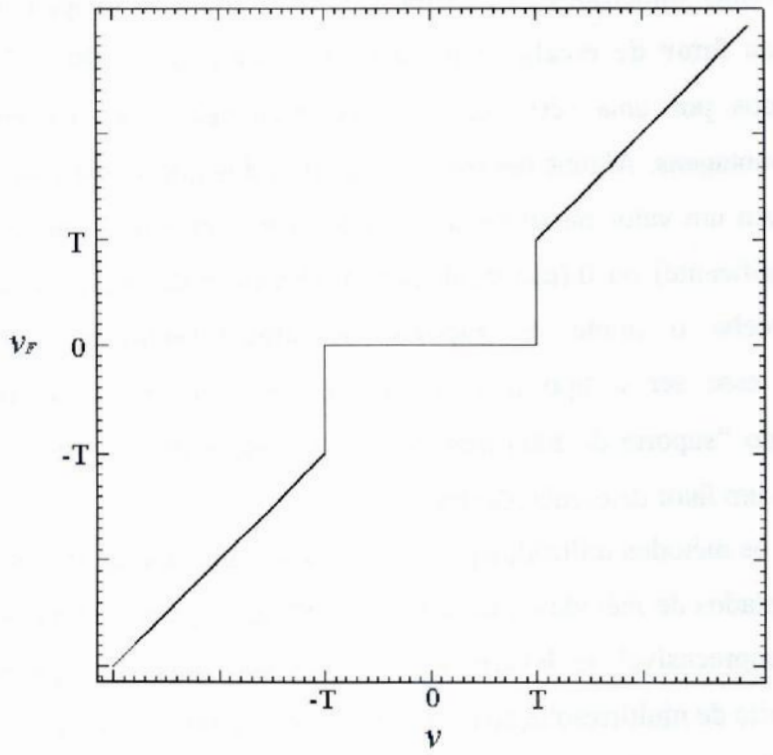


Figura 2.2: Filtragem por *hard thresholding*.

O método correspondente a essa técnica para a criação do suporte de multirresolução funciona a partir da comparação entre o valor de cada ponto no espaço de wavelet e o desvio padrão para o plano de wavelet ao qual pertence (obtido através do processo descrito na subseção 2.3.2 e utilizando a tabela 2.1). Se o módulo do valor do ponto for maior que o nível de corte (nosso *threshold*), igual a um certo número N de

vezes (usualmente entre 3 e 5)[1] esse desvio padrão, o nível de significância 1 é associado a ele; caso contrário sua significância é 0. É possível verificar que se multiplicarmos a estrutura de wavelet pelo suporte, os pontos com módulo acima do nível de corte serão mantidos como estão, enquanto os demais serão zerados, exatamente o comportamento imposto pela expressão (2.23).

Dependendo de qual será o uso futuro desse suporte, e devido à positividade característica das estruturas astronômicas comentada na seção 1.1, pode ser interessante utilizar nessa comparação o próprio valor do ponto em lugar do módulo, desprezando indiscriminadamente a significância dos pontos de valor negativo.

2.4.2 *Soft* e *semi-soft thresholding*

Outra técnica tradicional de filtragem de ruído semelhante ao *hard thresholding* é o ***soft thresholding***, dado por

$$v_F = \begin{cases} v - sig(v)\,T; & |v| \geq T \\ 0; & |v| < T \end{cases}, \tag{2.24}$$

onde $sig(v)$ é uma função que vale 1 se v for positivo, -1 se v for negativo e 0 se v valer 0. A diferença para a técnica *hard* é que, com o fim de manter a continuidade entre os valores dos pontos, estes, quando são, em módulo, acima do nível de corte, são subtraídos do mesmo (ou somados caso sejam negativos). Esse comportamento pode ser conferido na figura 2.3. Embora de aplicação em muitas áreas essa técnica não é de

---

[1] $N = 3$ fornece uma confiança de cerca de 99,7% na determinação de um ponto válido, sendo, em geral, um limite inferior aceitável (é importante lembrar que imagens astronômicas modernas facilmente chegam a 1 milhão de pontos, de forma que um erro em 0,3% dos pontos já é capaz de produzir efeitos não desprezíveis). A possibilidade de erros na medida dos desvios padrão e possíveis imperfeições na modelagem do ruído nos obrigam, na maioria dos casos, a adotar o valor $N = 5$. Problemas graves na modelagem do ruído, em geral decorrentes da análise de dados de natureza atípica, podem criar a necessidade de adotar valores ainda maiores em alguns casos extremos. Por outro lado, se existe a intenção de analisar estruturas extremamente tênues e se há grande confiança na modelagem de ruído adotada, é possível adotar valores realmente baixos, como $N = 2$, embora isso em geral implique em um risco de contaminação do suporte por ruído.

muito uso em imagens diretas astronômicas, já que modifica drasticamente o fluxo das fontes, comprometendo a fotometria. Nem tampouco pode ser usada com a finalidade de gerar um fator de escala para análise multiescalar, já que sua aplicação não pode ser expressa como uma multiplicação ponto a ponto entre os dados originais e uma outra estrutura.

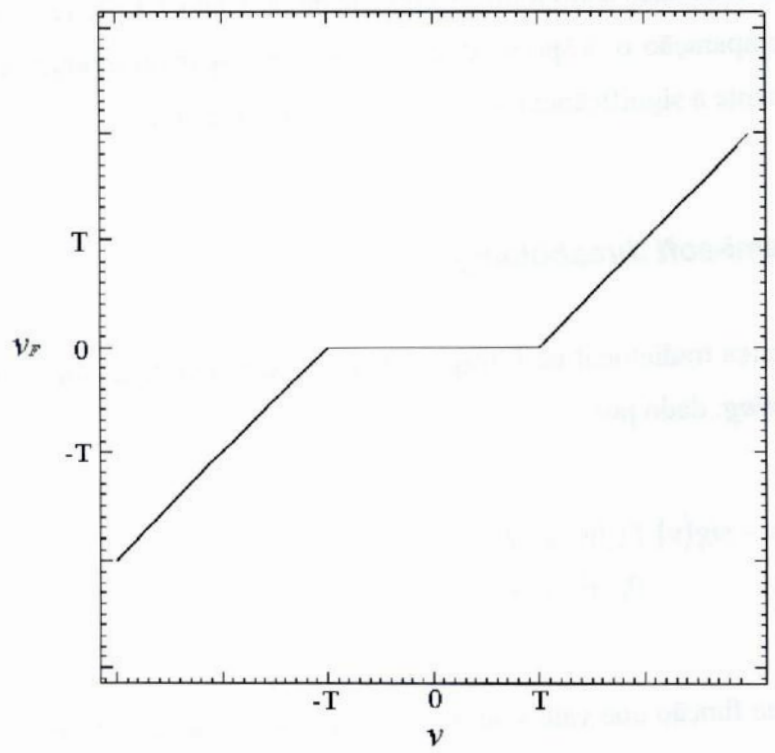


Figura 2.3: Filtragem por *soft thresholding*.

Uma técnica alternativa que diminui a descontinuidade do suporte e preserva melhor o valor dos pontos válidos é conhecida como ***semi-soft thresholding***, dado por

$$v_F = \begin{cases} v & ; |v| \geq T_2 \\ v \cdot \left(\dfrac{|v| - T_1}{T_2 - T_1} \right) & ; T_1 \leq |v| < T_2 \\ 0 & ; |v| < T_1 \end{cases} \qquad (2.25)$$

onde se utiliza não um, mas dois *thresholds* diferentes, sendo que a diferença básica para o método *hard* é que para valores abaixo do limite maior, mas acima do menor, o

valor filtrado é uma versão atenuada do original. Esse comportamento é mostrado na figura 2.4.

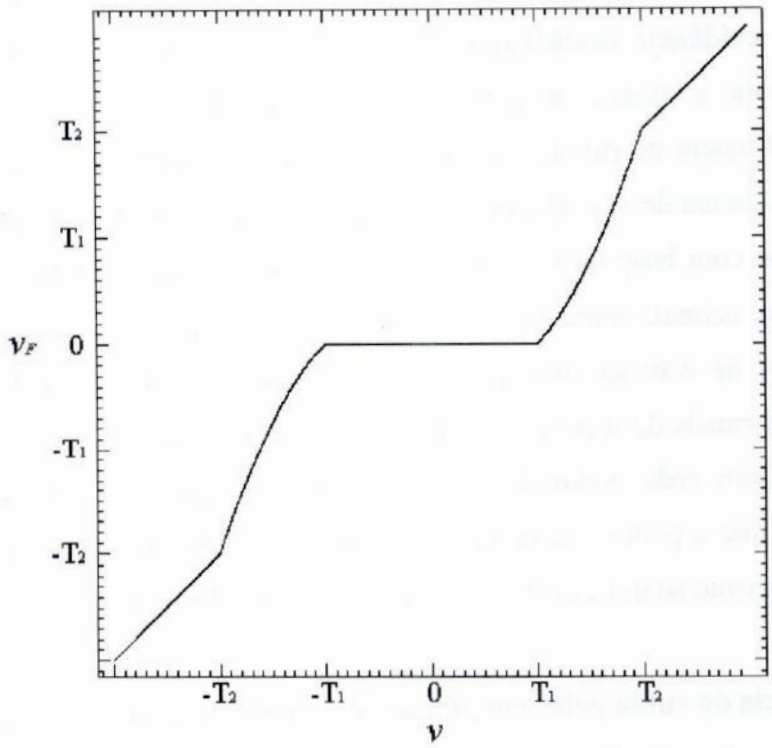


Figura 2.4: Filtragem por *semi-soft thresholding*.

Essa técnica pode ser adaptada com proveito para a obtenção de um suporte de multirresolução. Estabelecemos um limite superior, acima do qual (em módulo) os pontos são considerados completamente significantes (valor do suporte igual a 1) e um inferior, abaixo do qual são completamente não significantes (suporte igual a 0). A pontos com valores intermediários são atribuídos valores entre 0 e 1, indicando significância parcial. Estes valores são, em geral, obtidos através de uma interpolação linear, mas há outras formas possíveis para sua determinação. O nível de corte superior é escolhido de forma semelhante ao que se faria com o método de *hard thresholding*, enquanto o inferior possui um valor arbitrário menor que esse (quando o limite superior é de cinco vezes o desvio padrão, por exemplo, é comum a definição do inferior para três vezes esse desvio). A mesma observação sobre a utilização não do módulo, mas do próprio valor do ponto feita para o método de *hard thresholding* aplica-se aqui.

2.4.3 *Thresholding* de evidência combinada

Um método recente para determinação da validade de pontos num espaço multiescalar, aproveitando as qualidades de diversos métodos mais tradicionais é o ***thresholding* de evidência combinada** (Brown 2000). Esse método foi desenvolvido especialmente para a análise multiescalar, não sendo inspirado em uma técnica tradicional de filtragem de ruído. Ele utiliza dois critérios para considerar um ponto como ruidoso. Cada um desses critérios cria uma estrutura denominada **evidência**. Uma delas é produzida com base na intensidade de cada ponto (que é a base de todos os métodos descritos acima), enquanto a outra é calculada a partir da correlação entre pontos em planos de wavelet subseqüentes. As dimensões de uma evidência são as mesmas da transformada de wavelet da imagem. Uma evidência pode ter um valor que varia de 0 a 1 para cada ponto da imagem. Um valor igual a 1 indica uma alta probabilidade de que o ponto seja dominado pelo ruído e um valor igual a 0 indica o contrário; inversamente ao que ocorre com o suporte. As evidências são:

1. **Evidência do ruído pelas magnitudes dos coeficientes de wavelet**:

Aplicamos um método semelhante ao *semi-soft thresholding*. Calculamos o valor da primeira evidência segundo

$$E_1(x,y,z) = \begin{cases} 0 & ; |w(x,y,z)| \geq T(z) \\ 1 & ; |w(x,y,z)| = 0 \\ \dfrac{e^{\left(\frac{|w(x,y,z)|}{T(z)}\right)} - e}{1-e} & ; \textit{em outro caso} \end{cases}, \tag{2.26}$$

onde $T(z)$ é o nível de corte para a escala (ou plano de wavelet) z, $w(x,y,z)$ é apenas uma notação mais conveniente, em alguns casos, para $w_z(x,y)$ (valor de um ponto no espaço de wavelet), e $E_1(x,y,z)$ é a evidência de magnitude que calculamos para o ponto. O operador de módulo aplicado a $w(x,y,z)$ pode ser removido, se quisermos impor a positividade, como explicado para os dois métodos anteriores.

2. **Evidência de ruído pela correlação entre os planos de wavelet**:

É de se esperar que uma estrutura real (que não seja um artifício induzido pelo ruído) esteja representada em mais de um plano de wavelet. Esse comportamento não deve ser, de forma geral, observado no ruído, no entanto. Assim, uma forma de refinarmos o critério de significância é introduzindo um fator baseado na correlação entre planos de wavelet diferentes. Esse tipo de informação estará contido nesta segunda evidência. Para sua obtenção, primeiro calculamos, para cada plano de wavelet, a sua correlação ponto a ponto com o plano de escala imediatamente superior:

$$Corr_z = w_z \cdot w_{z+1} \tag{2.27}$$

onde a multiplicação é feita ponto a ponto.

Calculamos então um fator de normalização, dado por

$$Ncorr_z = \sqrt{\frac{\sum_x \sum_y w_z^2(x,y)}{\sum_x \sum_y Corr_z^2(x,y)}}\ , \tag{2.28}$$

que é utilizado para a obtenção de uma correlação normalizada entre os planos:

$$CN(x,y,z) = Corr_Z(x,y) \cdot Ncorr\ . \tag{2.29}$$

Finalmente, calcula-se $E_2(x,y,z)$, a evidência de correlação, utilizando

$$E_2(x,y,z) = \begin{cases} 0 & ; \ |CN(x,y,z)| \geq |w(x,y,z)| \\ 1 & ; \ |CN(x,y,z)| = \min(|CN|)\ , \\ \tilde{E}(x,y,z) & ; \ \textit{em outro caso} \end{cases} \tag{2.30}$$

onde

$$\tilde{E}(x,y,z) = \frac{\left|w(x,y,z)\right| - \left|CN(x,y,z)\right|}{\left|w(x,y,z)\right| - \min\left(\left|CN\right|\right)}, \tag{2.31}$$

sendo que $\min\left(\left|CN\right|\right)$ retorna o menor valor encontrado em *CN*.

Não é possível calcular essa segunda evidência para o último plano de wavelet obtido com a transformação *à trous*, pois não há um plano superior para a estimativa dessa correlação. Nesse caso, estipulamos que a evidência do último plano deve ser igual a 1 em todos os pontos.

De posse das duas evidências, podemos combiná-las em uma evidência única através de uma média geométrica (ponto a ponto):

$$E = \sqrt{E_1.E_2}\,. \tag{2.32}$$

E finalmente podemos calcular nosso suporte de multirresolução $S(x,y,z)$ segundo:

$$S(x,y,z) = \begin{cases} 0 & ; \quad E(x,y,z) \geq E_S \\ 1 & ; \quad E(x,y,z) \leq E_I \\ \dfrac{E_S - E(x,y,z)}{E_S - E_I} & ; \quad \textit{em outro caso} \end{cases} \tag{2.33}$$

onde os valores E_S e E_I devem ser fornecidos como níveis de tolerância inferior e superior. Brown (2000) sugere, para a maior parte das aplicações, $E_S = 0,6$ e $E_I = 0,1$ como valores ótimos.

2.4.4 Implementação do cálculo do suporte

O cálculo do suporte de multirresolução ou do fator de escala é realizado pelo programa, **OV_Support**, que incorpora os métodos *hard* e *semi-soft thresholding* e o de evidências combinadas de Brown. A chamada ao programa é feita através de:

```
sup = ov_support(wv, std)
```

onde `wv` é a estrutura de wavelet da imagem original (ou, nos casos de ruído de Poisson ou misto, da transformada de Anscombe da imagem) e `std` o desvio padrão do ruído calculado com OV_Noise (ou igual a exatamente 1, nos casos de Poisson e misto). A variável `sup` irá conter a informação retornada pelo programa, sob a forma de uma matriz tridimensional com as mesmas dimensões de `wv`, de tipo "byte" (se for utilizado o método *hard thresholding*) ou de ponto flutuante (se for utilizado um dos outros métodos).

Os dados da tabela 2.1 são obtidos por esse programa através da sub-rotina OV_Table.

Não houve grandes complicações na tradução em código dos métodos de *thresholding*, exceto pelo método das evidências combinadas. Neste foi preciso algum cuidado para evitar problemas de divisão por zero com a implementação da equação (2.31). Foi necessário também, nesse caso, adotar um certo nível de reaproveitamento de memória para evitar problemas de falta de recursos, tendo em vista a quantidade de estruturas de dados intermediárias criadas antes do suporte propriamente dito.

3 Módulos Avançados

Construindo o caminho até o topo

Uma vez de posse das informações fundamentais sobre uma imagem e sua representação no espaço multiescalar, podemos realizar as tarefas de remoção de ruído, detecção hierárquica de fontes e reconstrução parcial ou total da imagem. As duas últimas, em particular, são muito mais complexas que as realizadas pelos módulos básicos e são de compreensão um pouco mais difícil, além de consumirem recursos computacionais em um nível muito maior.

3.1 Remoção do ruído

A diminuição da quantidade de ruído presente em uma imagem é alvo de muitas técnicas de processamento já desenvolvidas.

Em astronomia, a remoção de ruído é freqüentemente vista com reservas, pois há um risco de se deformar a informação originada nos objetos verdadeiros, o que pode causar erros consideráveis na fotometria desses objetos, por exemplo.

No entanto, esse problema não é tão grave, em aplicações de reconhecimento de padrões e detecção de fontes, já que nesses casos o que se deseja conhecer é basicamente a existência ou não dos objetos e, caso existam, sua posição; medidas fotométricas exatas não são tão importantes. Nessas situações, o emprego de procedimentos de remoção de ruído como uma etapa de pré-processamento pode ser muito recompensador.

Algumas técnicas multiescalares para a remoção de ruído que foram incluídas em nosso pacote de programas são apresentadas aqui.

3.1.1 Filtragem simples

A aplicação do suporte de multirresolução é a simples operação de multiplicar o valor de cada ponto no espaço de wavelet pelo valor correspondente no suporte, dando origem a uma nova estrutura, a estrutura filtrada de wavelet. Essa operação será denotada daqui em diante como

$$w_F = S(w_a) \cdot w_b, \tag{3.1}$$

onde w_a é a estrutura de wavelet a ser filtrada; $S(w_b)$ é o suporte de multirresolução obtido a partir da análise de significância de uma outra estrutura de wavelet w_b; e w_F é a estrutura filtrada resultante. Freqüentemente aplica-se à estrutura de wavelet o seu próprio suporte de multirresolução, de forma que w_a e w_b são o mesmo objeto. Nesse caso escrevemos simplesmente

$$w_F = S \cdot w, \tag{3.2}$$

onde w é a estrutura de wavelet a ser filtrada[1].

Realizando nessa nova estrutura filtrada a transformada inversa de wavelet, que no caso da técnica *à trous* corresponde à soma de todos os diferentes planos de wavelet e da imagem residual, obtemos o que se chama imagem filtrada, uma versão sem ruído da imagem original[2]:

$$f_F = W^{-1}[w_F] = W^{-1}[S \cdot W[f]], \tag{3.3}$$

onde f representa a imagem original e f_F a imagem filtrada.

---

[1] Pode parecer estranha, a princípio, a possibilidade de se aplicar a uma estrutura de wavelet o suporte obtido a partir de outra; mas, como veremos na seção 3.3, esse tipo de operação torna-se útil quando tentamos reconstruir parte das componentes de uma imagem. Por enquanto, no entanto, não precisamos nos preocupar com essa possibilidade e podemos adotar a notação simplificada.

[2] Convém notar que o termo geral "filtragem" é empregado aqui com o significado específico de "filtragem para remoção de ruído", como é comum em outros textos sobre o assunto.

Infelizmente, não só o ruído é eliminado no processo, mas também uma parte, em geral não desprezível, da informação verdadeira da imagem. A estratégia empregada na determinação do suporte terá grande influência sobre a qualidade dos resultados.

Esse processo de supressão de ruído é conhecido como filtragem simples.

3.1.2 Técnicas aperfeiçoadas de filtragem

Um outro tipo de filtragem mais elaborado consiste em um processo iterativo de recuperação das estruturas reais. Conhecido como filtragem adaptativa, o método segue a seguinte seqüência de operações:

1. Realiza-se uma filtragem simples:

$$f_{F_0} = W^{-1}[S \cdot W[f]]. \tag{3.4}$$

2. Subtrai-se a imagem original da imagem filtrada, gerando uma imagem residual:

$$r_n = f - f_{F_n}. \tag{3.5}$$

3. Faz-se uma filtragem simples nessa imagem residual. A imagem encontrada contém as estruturas que foram removidas junto com o ruído. Essa imagem residual filtrada é somada à imagem filtrada f_{F_n}:

$$r_{F_n} = W^{-1}[S \cdot W[r_n]]; \tag{3.6}$$

$$f_{F_{n+1}} = f_{F_n} + r_{F_n}. \tag{3.7}$$

4. Se o resultado for satisfatório (segundo algum critério de parada), o processo termina. O resultado é a imagem filtrada adicionada dos resíduos filtrados a cada iteração:

$$f_F = f_{F_N} = f_{F_n} + \sum_{n=0}^{N-1} r_{F_n}, \tag{3.8}$$

onde N é o número total de iterações utilizados no processo. Caso contrário, volta-se ao passo 2.

A remoção do ruído de uma imagem pode também ser feita através da inversão da aplicação do suporte por um método iterativo, no que chamamos de reconstrução total. Esse tópico será explicado mais adiante, quando tratarmos da reconstrução de objetos isolados.

3.1.3 Implementação da remoção de ruído

A remoção de ruído foi implementada em uma rotina, **OV_DeNoise**, que realiza as operações tanto de filtragem simples quanto de filtragem adaptativa:

```
clean = ov_denoise(wv, sup, res)
```

onde `wv` contém a transformada *à trous* dos dados originais, `sup` é o suporte de multirresolução para esses dados e `res` é a imagem residual da transformada *à trous*. A variável `clean` irá conter a imagem filtrada, com as mesmas dimensões da original.

Para a filtragem simples, a chamada à rotina equivale ao comando IDL:

```
clean = total(wv * sup, 3) + res
```

onde `total` é uma função interna da IDL que retorna a soma dos elementos de uma matriz. O número 3 é passado como argumento a `total` para indicar que a soma deve ser feita apenas ao longo da terceira dimensão dos dados (escala). O asterisco (*) denota, em IDL, a multiplicação.

Para a filtragem adaptativa, o critério de parada pode ser baseado simplesmente no número de iterações ou na convergência do ruído estimado.

A reconstrução total é realizada pelo módulo de reconstrução, OV_Reconstruct, e tem seus detalhes explorados na seção 3.3 (e especialmente em 3.3.4 e 3.3.5).

Um exemplo de aplicação do programa é mostrado na figura 3.1 abaixo.

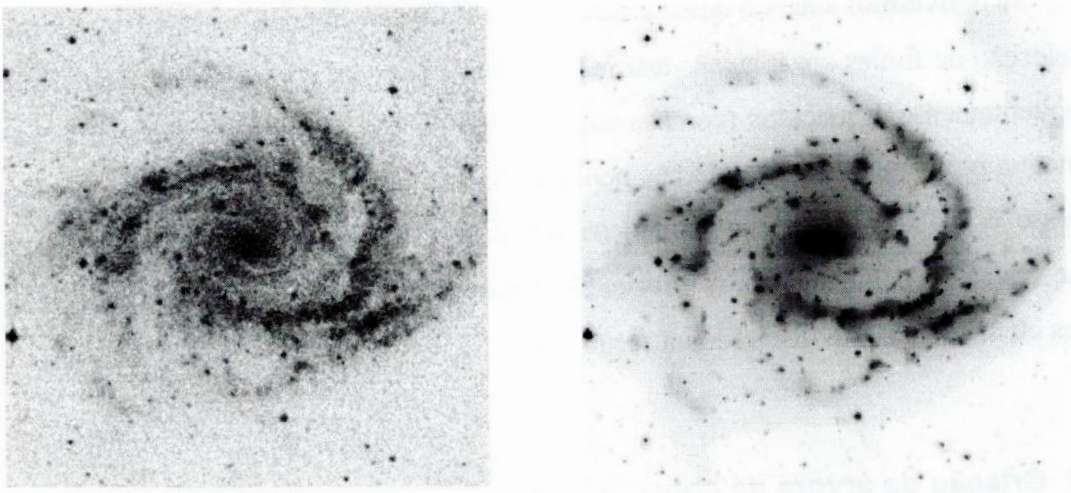


Figura 3.1: Filtragem simples feita a partir de um suporte obtido com um *hard thresholding*. À esquerda, imagem de NGC 2997 com ruído artificial adicionado. À direita sua versão filtrada.

3.2 Detecção de objetos

Observando o suporte de multirresolução de uma imagem astronômica, verificamos que existem conjuntos de pontos contíguos significantes nas áreas identificadas com fontes reais na imagem, sendo que o formato dessas regiões segue aproximadamente o aspecto da fonte. O mais importante é que, em geral, essas **regiões de validade** estão presentes em vários níveis do suporte, já que um objeto tende a se apresentar em várias escalas consecutivas do espaço de wavelet.

As regiões de validade estabelecem uma forma de segmentação da imagem no espaço tridimensional de wavelet. Sua natureza, no entanto, é bidimensional (são conjuntos conexos de pontos válidos em um dado plano de wavelet). Como a informação sobre um objeto pode estar presente em vários planos de wavelet, é preciso estabelecer um critério de conectividade entre regiões de validade em diferentes planos e um modo de avaliar se um conjunto de regiões conectadas pertence a um objeto real.

Para que realizemos esse tipo de estudo, devemos utilizar um suporte de multirresolução estabelecido de forma que os valores negativos da estrutura de wavelet tenham sido indiscriminadamente considerados não significantes, ou corremos o risco de detectar como objetos os anéis negativos presentes ao redor dos objetos nos planos de wavelet e introduzidos pela condição de admissibilidade da transformada.

Se obtivermos sucesso nesse critério de definição de objetos, teremos uma forma de detecção de fontes em imagens astronômicas, que leva em conta efeitos importantes e freqüentemente esquecidos (como a superposição) e que não requer, em geral, uma estimativa prévia nem do nível do céu nem da PSF característica da imagem. Se o nosso interesse é analisar apenas um objeto ou um grupo de objetos na imagem, teremos ainda, com essa segmentação, a informação necessária para a reconstrução individual desses elementos, como discutido na seção 3.3.

3.2.1 Criação da árvore de conectividade

Para estabelecer a conectividade entre regiões em vários planos de wavelet devemos utilizar um conjunto de regras adaptadas ao tipo de imagem analisada. Para imagens astronômicas, Starck, Murtagh e Bijaoui (1998) sugerem um procedimento bem simples. Para cada região de validade:

1. Identifica-se o espaço correspondente na representação multiescalar da imagem original.

2. Encontra-se o ponto de máxima intensidade nesse espaço.

3. Se, no suporte de multirresolução do plano seguinte (de escala imediatamente superior), o ponto encontrado para o máximo estiver localizado no interior de uma outra região, então as duas regiões estão conectadas (ver figura 3.2 abaixo).

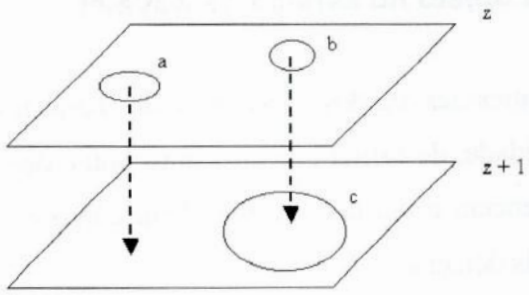


Figura 3.2: Conectividade entre regiões: a região **a** não está conectada a nenhuma outra, enquanto **b** está conectada a **c**. O ponto de onde as setas se projetam na figura representa o ponto de máximo da região.

Note que, dessa forma, é possível que mais de uma região em uma certa escala estejam conectadas a uma mesma região na escala imediatamente superior. É comum chamarmos cada região conectada a uma região de escala maior de **região filha** daquela, que seria a **região mãe**. Assim, construímos uma espécie de visão hierárquica das regiões válidas no espaço de wavelet. A visão que construímos pode ser encarada também como uma estrutura de árvores, em que os troncos seriam as regiões de escalas maiores, das quais se projetam, como ramos, as regiões em escalas menores (como mostrado na figura 3.3).

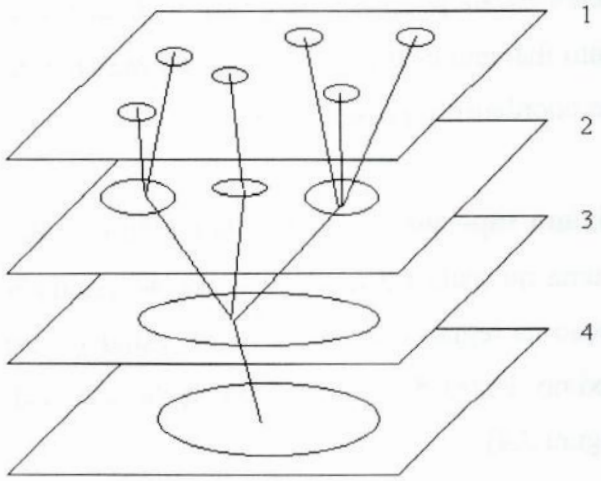


Figura 3.3: Árvore de conectividade entre as regiões no espaço multiescalar.

3.2.2 Definição de um objeto no espaço de wavelet

Uma visão hierárquica das regiões de validade nos permite realizar um estudo de suas relações de conectividade, de forma que possamos então determinar, no espaço de wavelet, quais delas pertencem a um dado objeto, bem como as relações hierárquicas deste objeto com os demais detectados.

O critério empregado para a identificação de objetos no espaço multiescalar não é uma questão fechada. Os resultados de uma aplicação de detecção de fontes ou de qualquer outra que dependa dessa separação, como, por exemplo, a reconstrução parcial, são fortemente dependentes desse critério, de sua capacidade de definir objetos que correspondem aos objetos físicos presentes na imagem original.

Para essa tarefa, Starck, Murtagh e Bijaoui propõem o seguinte procedimento, que impõe um critério de definição de objeto que chamaremos **critério padrão**. Para cada região de validade:

1. Obtém-se o valor máximo da área equivalente no espaço de wavelet.

2. Obtém-se o **máximo inferior**, máximo da região filha da região estudada, se estiver definida (a região pode não estar conectada a nenhuma filha ou pode simplesmente estar na menor escala possível). Se existir mais que uma filha, toma-se o máximo da filha cujo ponto máximo está mais próximo do máximo da região estudada (considerando apenas suas coordenadas posicionais).

3. Obtém-se o **máximo superior**, o máximo da região mãe da região estudada. Se a região mãe possui uma ou mais filhas além da região estudada, considera-se o máximo dentro da interseção da região mãe com a região estudada transportada para a escala superior (ou o máximo dentro da "sombra" da região analisada sobre a região mãe, como mostrado na figura 3.4).

4. Compara-se o máximo da região com os máximos superior e inferior. Se for maior que ambos, ou maior que um deles estando o outro não definido, considera-se que a região estudada é a **região principal** de um objeto no espaço de wavelet, e que esse objeto contém todos os pontos desta região, assim como os pontos de suas regiões filhas (e os das filhas destas filhas, os das filhas destas etc.). Note que se tanto o máximo

superior quanto o máximo inferior não estiverem definidos (ou seja, se a região estudada for isolada no espaço multiescalar), então a região não define um objeto.

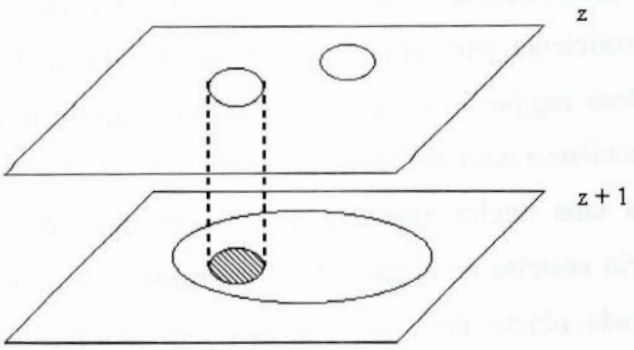


Figura 3.4: O máximo superior, no caso em que a região mãe possui outras filhas, é obtido dentro da "sombra" da região estudada sobre a região mãe.

O método descrito acima foi desenvolvido para ser empregado apenas em suportes obtidos com o método *hard thresholding*, contendo apenas os valores 0 e 1. Para suportes não binários, adaptamos esse procedimento para que as regiões fossem definidas como conjuntos conexos de pontos com valor significância maior que zero (na verdade, nossa implementação permite escolher um outro nível de corte diferente de 0 para a significância).

Assim é possível construir uma estrutura de dados que contém a informação sobre quais pontos no espaço de wavelet estão associados a um dado objeto. Uma estrutura com as dimensões do espaço de wavelet pode ser definida para um objeto, apresentando para cada um desses pontos um valor igual ao corresponde no suporte de multirresolução e 0 nos demais. Essa estrutura é chamada **suporte individual**, do objeto. Uma hierarquia entre os objetos pode ser facilmente estabelecida, sendo que cada objeto possui como sub-objetos aqueles cujos suportes individuais localizam-se inteiramente dentro de seu próprio suporte. O nível (plano de wavelet) em que se localiza a região principal nos dá a **escala** de um objeto.

Esse critério padrão gera bons resultados no que se refere à identificação de objetos físicos reais, mas permite desvios indesejáveis na definição da escala do objeto. É possível que objetos com a mesma forma e o mesmo tamanho, mas em condições ligeiramente diferentes no que se refere ao fluxo de fundo e à quantidade de ruído,

sejam classificados como de escalas diferentes. Esse desvio pode ser um grande empecilho à reconstrução correta de um objeto.

Além disso, o critério padrão associa a objetos grupos de apenas duas regiões conectadas. Essas regiões, quando estão associadas a fontes reais, não representam, em geral, informação suficiente para uma reconstrução do objeto. Muitas vezes ainda, grupos de apenas duas regiões conectadas não são indicativos de fontes verdadeiras, mas de simples coincidência entre componentes multiescalares do ruído.

Introduzimos uma ligeira alteração no procedimento, dando origem ao que chamamos de **critério restrito** de identificação de objetos, que se destaca por impor a condição de que cada objeto deva ser definido por não menos que três regiões conectadas, amenizando os problemas mencionados. Os máximos são calculados exatamente da mesma forma, apenas a forma de declarar a existência ou não de um objeto é alterada.

Segundo o critério restrito, existem três casos em que uma região pode ser considerada a região principal de um objeto:

1. O máximo da região estudada é maior que os máximos superior e inferior. Essa é basicamente a mesma exigência imposta pelo critério padrão, mas aqui não se enquadram os casos em que um dos máximos não esteja definido.

2. O máximo da região estudada é maior que o máximo inferior, que é fornecido por uma região que possui regiões filhas, e o máximo superior não é definido. Exceto pela exigência de que a região que fornece o máximo inferior possua filhas, esse caso também corresponde à exigência básica do critério padrão, no caso em que o máximo superior não está definido.

3. O máximo da região estudada é maior que o máximo superior e menor que o máximo inferior, que é fornecido por uma região que não possui regiões filhas.

Ambos os critérios, padrão e restrito, são baseados em princípios bastante simples. Um deles é o de que se um objeto possui um tamanho característico de uma escala, então sua intensidade relativa nessa escala deve ser máxima (apenas o 3º caso do critério restrito admite o contrário).

Embora formalmente só sejam consideradas como integrantes do suporte individual de um objeto a sua região principal e as filhas desta, a região mãe da região principal pode ser utilizada posteriormente pelo algoritmo de reconstrução e é denominada **região complementar** do objeto. Um objeto que não a possua é denominado **objeto especial**, e sua reconstrução é, em geral, menos eficiente.

3.2.3 Implementação da detecção de objetos

O estabelecimento da árvore de conectividade, assim como a detecção dos objetos pela análise desta foi implementado no programa **OV_Regions**, executado através do comando:

```
reg = ov_regions(wv, sup)
```

onde `wv` e `sup` são a estrutura de wavelet e o suporte de multirresolução da imagem a ser analisada, respectivamente, e `reg` é a estrutura de informações retornada pelo programa, cuja natureza é descrita adiante. A análise feita pelo programa é realizada em algumas etapas bem definidas:

1. Com o auxílio de rotinas internas IDL, uma nova matriz valores inteiros longos é criada, com as mesmas dimensões do suporte de multirresolução. Os pontos em cada região contígua no suporte de multirresolução são assinalados nessa nova estrutura com um valor inteiro identificador da região. Esse valor é utilizado no futuro, para se resgatar as coordenadas dos pontos de uma região a partir de seu número de identificação.

2. É criado um vetor de estruturas de dados, com um número de elementos igual ao número de regiões identificadas no passo anterior. Essas estruturas irão conter todas as informações relevantes de cada região. A posição de uma estrutura no vetor será coerente com o número de identificação estabelecido para a região correspondente no passo 1.

3. É feita uma varredura pelas regiões identificadas no passo 1 para a obtenção das coordenadas do ponto máximo dentro da área equivalente a cada reagião na estrutura de wavelet da imagem. Essas coordenadas são armazenadas na estrutura definida no passo 2, juntamente com o valor do máximo.

4. Com as informações geradas nos passos 1 e 3, identificam-se as condições de conectividade para cada região. O número de identificação da região mãe, quando existir, e o número de regiões filhas são armazenados na estrutura de dados criada no passo 2.

5. Uma nova varredura é feita agora, utilizando as informações geradas nos passos anteriores, procurando estabelecer, para cada região, os máximos superior e inferior. Aplica-se um dos critérios de definição de objetos e atribui-se a cada região principal de um objeto um número de identificação único (associado a esse objeto), não relacionado aos números atribuídos às regiões no passo 1.

A matriz de identificação das regiões criada no passo 1 e o vetor de estruturas criado no passo 2, e preenchido ao longo dos passos seguintes, são retornados pelo programa sob a forma de uma nova superestrutura IDL. Para cada região sabemos:

- Exatamente quais os pontos que se situam em seu interior.
- Qual a posição do máximo correspondente à sua área no espaço de wavelet.
- Quais as suas relações de conectividade com as regiões em escalas vizinhas.
- Se é ou não a região principal de um objeto.

Embora essas informações sobre as regiões de validade sejam importantes para a nossa análise, não seria, em princípio, desejável que um usuário do pacote tivesse que lidar com elas de forma direta. Em geral o que interessa ao usuário são as informações sobre os objetos. Desenvolvemos, portanto, um outro programa, **OV_Objects**, que toma como entrada a saída de OV_Regions e reorganiza parte de sua informação de forma orientada aos objetos e não às regiões. A chamada ao programa possui a forma:

```
obj = ov_objects(reg)
```

sendo `reg` a estrutura retornada por OV_Regions, e `obj` os dados sobre os objetos, a serem retornados pelo programa (com estrutura a ser definida adiante). O processamento feito por OV_Objects possui as seguintes etapas:

1. São buscadas, na estrutura de dados fornecida por OV_Regions, as regiões principais dos objetos detectados.

2. Cria-se um novo vetor de estruturas que deverá conter informações sobre cada objeto. O vetor terá um número de elementos igual ao número de objetos detectados. A ordem de disposição dos objetos nesse vetor será regida pelos números dados às suas regiões principais no passo 5 de OV_Regions. Esses serão também seus números de identificação.

3. Atribui-se a cada objeto um conjunto de três de coordenadas no espaço multiescalar: as coordenadas x e y são aquelas em que se encontrou o máximo da região principal, e a coordenada z, escala, é igual à escala dessa região.

4. Encontra-se, por um processo recursivo, o número de identificação de cada região pertencente a um dado objeto (filhas da região principal, filhas de cada uma dessas filhas e assim por diante). Esses números são acomodados em um vetor de inteiros longos, sendo o primeiro elemento a identificação da região principal. Um ponteiro para esse vetor é armazenado na estrutura referente ao objeto criada no passo 2.

5. Para cada objeto, uma pesquisa pelo vetor criado no passo 4 revela quais outros objetos têm como região principal alguma de suas regiões filhas. Esses são considerados sub-objetos daquele e seus números de identificação são armazenados de forma semelhante ao que foi feito para as regiões filhas no passo 4.

6. A partir dos dados de conectividade entre as regiões, é armazenado na estrutura de objetos o número de identificação da região complementar (mãe da região principal) de cada objeto. Para cada um, também é determinado se a região complementar possui outras filhas além da sua região principal. A utilidade dessa informação só será discutida na seção 3.3, quando tratarmos da reconstrução individual dos objetos.

O programa OV_Objects retorna o vetor de estruturas de objetos criado. Cada elemento desse vetor possui:

- Uma estimativa de posição (x e y), uma escala de tamanho característica (z).
- Referências às regiões nas quais é definido e à sua região complementar.
- Referências aos sub-objetos e referência a seu super-objeto (do qual é sub-objeto).

Essas informações podem ser acessadas com relativa facilidade. Digamos que o programa OV_Objects foi executado e sua estrutura de retorno armazenada na variável `obj`. A posição aproximada do décimo objeto detectado, por exemplo, pode ser exibida através do comando IDL:

```
print, obj[9].x, obj[9].y
```

Como a indexação em IDL se inicia com 0, o elemento de índice 9 será o décimo. Convém lembrar que essa posição obtida é apenas a posição do máximo multiescalar (máximo da região principal), possuindo precisão máxima de um píxel e não correspondendo, em geral, à posição do centróide[1]. A posição do centróide pode ser obtida a partir da aplicação do suporte individual do objeto à estrutura de wavelet da imagem, assim como algumas outras propriedades. Na verdade, os parâmetros posicionais e de fluxo dos objetos só podem ser obtidos com boa precisão a partir da reconstrução de sua imagem, como veremos na próxima seção.

Um exemplo de aplicação dos programas de detecção pode ser visto na figura 3.5 a seguir:

---

[1] Em uma distribuição discreta de intensidades, como uma imagem digital, o centróide pode ser definido como a posição obtida através da média dos valores das coordenadas de cada ponto interno ao objeto ponderada pelas intensidades desses pontos.

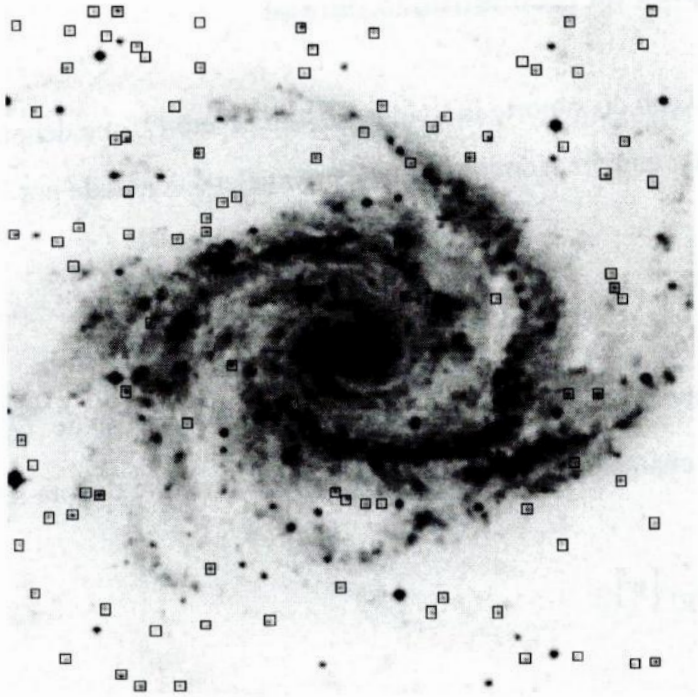


Figura 3.5: Detecção de objetos na imagem da galáxia NGC 2997. São marcados apenas os objetos detectados que não são sub-objetos de nenhum outro e que possuem escala menor ou igual a 1.

3.3 Reconstrução parcial da imagem

O procedimento para a detecção de objetos a partir do espaço de wavelet nos fornece uma visão segmentada e bem definida de cada objeto. Os pontos contidos nas regiões associadas a um objeto são, em princípio, livres da contaminação pelo ruído ou da presença de objetos próximos. A simples aplicação do suporte individual do objeto seguida de uma transformação inversa de wavelet, no entanto, nos mostra uma imagem do mesmo com algumas de suas características distorcidas (principalmente pelo fato de que esse suporte não inclui a informação contida nas escalas superiores à do objeto). A obtenção de uma imagem de boa qualidade de um objeto depende do emprego de técnicas iterativas de inversão, que veremos a seguir.

3.3.1 Princípios da reconstrução parcial

A aplicação do suporte individual de um objeto O, que denotaremos como S_O, à representação de uma imagem no espaço de wavelet é denotada por

$$w_P = S_O \cdot w \ , \tag{3.9}$$

onde w_P é denominada **estrutura de wavelet projetada** de O. Essa operação é freqüentemente chamada de **projeção** de w no suporte S_O e denota-se

$$w_P = \mathrm{Pr}_{(S_O)}\left[w\right] . \tag{3.10}$$

Em geral omite-se o suporte de projeção na notação, admitindo-se que seja o mesmo ao longo de todo o processo, obtido a partir do objeto a ser reconstruído:

$$w_P = \mathrm{Pr}\left[w\right] . \tag{3.11}$$

Segundo Starck, Murtagh e Bijaoui (1998), melhores resultados são obtidos com o processo de reconstrução se considerarmos como parte do suporte individual de um objeto a sua região complementar. Caso esta possua outras filhas além da região principal do objeto, apenas a interseção entre ela e a principal (calculada como se estivessem em um mesmo plano) deve ser considerada parte do suporte.

Se a projeção descarta realmente toda a informação pertinente a outros objetos, então a estrutura resultante deveria poder ser obtida também pela projeção da estrutura de wavelet w_R de uma imagem que contivesse apenas o objeto O:

$$w_P = \mathrm{Pr}\left[w_R\right] . \tag{3.12}$$

Essa estrutura, w_R, é a **estrutura reconstruída** de O. O problema da reconstrução do objeto é a inversão da equação acima para a obtenção de w_R (e,

conseqüentemente, da imagem reconstruída, f_R , através da transformada inversa de wavelet).

Por (3.11) e (3.12), temos que, para uma reconstrução ideal

$$\Pr[w_R] = \Pr[w] . \tag{3.13}$$

Evidentemente, existe uma solução trivial:

$$w_R = w , \tag{3.14}$$

mas essa solução não nos interessa, pois contém todos os outros objetos presentes na imagem. Existe uma infinidade de outras soluções. Um critério importante para eliminar soluções indesejáveis é o de que a solução deve ser verdadeiramente a transformada de wavelet de uma imagem (e não apenas uma estrutura com as mesmas dimensões de uma transformada de wavelet), ou seja, deve obedecer a

$$w_R = WW^{-1}[w_R] , \tag{3.15}$$

segundo o que foi discutido na subseção 2.2.2.

No caso de imagens diretas astronômicas, podemos ainda impor uma condição de positividade. Todos os pontos da imagem reconstruída (transformada inversa da estrutura reconstruída) devem possuir valores positivos:

$$f_R(x,y) \geq 0 \qquad \forall (x,y) . \tag{3.16}$$

Os dois métodos de reconstrução apresentados a seguir utilizam os critérios apresentados em (3.13), (3.15) e (3.16).

3.3.2 Método direto

O **método direto** de reconstrução (Bijaoui e Rué, 1995) estabelece uma solução para a equação (3.13) através do seguinte procedimento:

1. A estimativa inicial para a estrutura de wavelet reconstruída, w_R, do objeto O é dada pela projeção da transformada de wavelet da imagem original:

$$w_{R_0} = w_P \ . \tag{3.17}$$

2. Calcula-se a estrutura complementar ao suporte individual do objeto, denominada **suporte externo**, segundo

$$Z_O = 1 - S_O \ , \tag{3.18}$$

onde a subtração é feita ponto a ponto.

3. Calcula-se uma nova estrutura w'_{R_n} que satisfaça a condição (3.15):

$$w'_{R_n} = WW^{-1}\left[w_{R_n} \right]. \tag{3.19}$$

De fato, substituindo w'_{R_n} em (3.15) temos:

$$WW^{-1}\left[w_{R_n}\right] = WW^{-1}\left[\ WW^{-1}\left[w_{R_n}\right]\ \right], \tag{3.20}$$

ou

$$WW^{-1}\left[w_{R_n}\right] = WW^{-1}WW^{-1}\left[w_{R_n}\right] \ ; \tag{3.21}$$

mas como

$$W^{-1}W = 1 \, , \tag{3.22}$$

temos que os dois lados da equação são idênticos, o que prova que w'_{R_n} satisfaz (3.15).

4. Para que as condições (3.13) e (3.15) sejam satisfeitas simultaneamente (de forma aproximada), fazemos

$$w''_{R_n} = S_O \cdot w + Z_O \cdot w'_{R_n} \tag{3.23}$$

ou seja, determinamos que dentro do suporte individual do objeto teremos os valores dados por w, satisfazendo a (3.13), enquanto fora, os valores serão determinados pela condição (3.15).

5. Finalmente, aplicamos a condição dada por (3.16):

$$f_{R_{n+1}}(x,y) = \begin{cases} f''_{R_n}(x,y) & ; \quad f''_{R_n}(x,y) > 0 \\ 0 & ; \quad \textit{em outro caso} \end{cases}, \tag{3.24}$$

onde

$$f''_{R_n} = W^{-1}\left[w''_{R_n}\right]. \tag{3.25}$$

Se algum critério de parada for satisfeito, então $f_{R_{n+1}}$ é a estimativa final para a imagem reconstruída e sua transformada *à trous* é uma solução aproximada de (3.13). Caso contrário, calculamos

$$w'_{R_{n+1}} = w_{R_{n+1}} = W\left[f_{R_{n+1}}\right] \tag{3.26}$$

e retornamos ao passo 4 (o passo 3 não será necessário, já que $w'_{R_{n+1}}$ é a transformada de wavelet de uma imagem).

Esse método muito simples tem a grande vantagem de não exigir cálculos muito complexos, o que acarreta em uma maior velocidade de execução. Por outro lado,

experimentos simples mostram que o procedimento diverge em muitos casos. Além disso, os resultados dificilmente são melhores do que aqueles obtidos com o método do gradiente, descrito a seguir.

3.3.3 Método do gradiente e *steepest descend*

O **método do gradiente** (Bijaoui e Rué 1995) difere do direto principalmente por não impor a condição dada em (3.13) de forma bruta, pela substituição dos valores na estimativa da solução. O método induz uma convergência lenta e natural da estimativa, reduzindo, a cada passo, o valor do módulo

$$\| \Pr[w_R] - w_P \| , \tag{3.27}$$

onde o módulo é definido por

$$\| A \| = \sqrt{\sum A^2} , \tag{3.28}$$

para uma estrutura A qualquer.

O método consiste nos seguintes passos:

1. A estimativa inicial para a estrutura de wavelet reconstruída, w_R, do objeto O é dada pela projeção da transformada de wavelet da imagem original:

$$w_{R_0} = w_P . \tag{3.29}$$

2. Calcula-se uma nova estrutura que satisfaz a condição (3.15), como no método direto:

$$w'_{R_n} = WW^{-1}\left[w_{R_n} \right]. \tag{3.30}$$

3. Uma **estrutura de wavelet residual** r_n é calculada como sendo

$$r_n = w_P - \mathrm{Pr}[w'_{R_n}] . \tag{3.31}$$

Essa é a diferença entre a projeção da estrutura de wavelet original e a projeção da estrutura estimada.

4. A partir da estrutura residual, calculamos uma **imagem residual**:

$$\tilde{f}_n = \tilde{\mathrm{Pr}}\,[r_n] . \tag{3.32}$$

O operador $\tilde{\mathrm{Pr}}$, conhecido como **operador adjunto** e definido mais adiante, fornece um efeito aproximadamente igual ao inverso do operador de projeção. Ele suaviza uma estrutura de wavelet projetada e a reagrupa em uma imagem bidimensional. Assim, $\tilde{f}_n$ será aproximadamente a imagem-diferença entre a imagem que se quer obter e a transformada de wavelet inversa da estimativa atual de w_R.

5. Modifica-se a estimativa atual da imagem reconstruída (transformada inversa de wavelet da estrutura reconstruída) pela adição de $\tilde{f}_n$, gerada no passo 4, regulada em fluxo através da multiplicação por um valor escalar α, para que uma convergência suave seja garantida (esse fator terá seu valor discutido adiante):

$$f'_{R_{n+1}} = f_{R_n} + \alpha \tilde{f}_n , \tag{3.33}$$

onde, como de costume,

$$f_{R_n} = W^{-1}[w_{R_n}] . \tag{3.34}$$

6. Aplicamos o critério de positividade (3.16) à imagem obtida, assim como é feito pelo método direto:

$$f_{R_{n+1}}(x,y) = \begin{cases} f'_{R_{n+1}}(x,y) & ; \quad f'_{R_{n+1}}(x,y) > 0 \\ 0 & ; \quad \textit{em outro caso} \end{cases} . \tag{3.35}$$

Caso algum critério de parada seja satisfeito, temos nossa imagem reconstruída, $f_{R_{n+1}}$. Sua transformada de wavelet é, por construção, a estrutura de wavelet reconstruída. Caso contrário, fazemos

$$w'_{R_{n+1}} = w_{R_{n+1}} = W\left[f_{R_{n+1}}\right] \tag{3.36}$$

e retornamos ao passo 3.

O operador adjunto $\tilde{\mathrm{Pr}}$ utilizado no passo 4 deve ser definido como

$$\tilde{\mathrm{Pr}}\,(w) = W^{-1}[\tilde{w}], \tag{3.37}$$

onde

$$\tilde{w}_z = H_1 \cdot H_2 \cdot H_3 \cdots H_{z-1}[w_z] \; ; \tag{3.38}$$

para a obtenção de bons resultados, segundo Starck, Murtagh e Bijaoui (1998).

A constante α pode possuir um valor fixo durante todo o processo (geralmente, nesse caso, $\alpha = 1$ é escolhido), ou pode ser adaptada a cada iteração. Uma forma freqüentemente escolhida é

$$\alpha_n = \frac{\left\| \tilde{f}_n \right\|}{\left\| r_n \right\|} . \tag{3.39}$$

Essa forma de determinação de α recebe o nome de ***steepest descend***, e permite uma convergência acelerada. Em compensação o cálculo de α a cada iteração representa um acréscimo considerável de tempo de execução.

Esse método, embora também possa divergir em casos extremos, o faz com menos freqüência que o método direto. Os resultados obtidos, em princípio, também deveriam ser muito mais próximos dos corretos.

3.3.4 Reconstrução total

Os métodos descritos podem ser aplicados também como uma técnica especial de remoção de ruído. A **reconstrução total** é realizada através dos mesmos procedimentos que são utilizados para a reconstrução parcial. A diferença é que, como operador de projeção utilizamos o suporte de multirresolução de forma integral, e não apenas o suporte individual. Assim, ao fim da execução do algoritmo, teremos reconstruído não um, mas todos os objetos presentes na imagem. Mais ainda, toda a informação associada a regiões significantes será reconstruída, mesmo que estas não estejam associadas a nenhum objeto em especial.

Para a execução de uma reconstrução total não é necessário nem mesmo que tenha sido feita a detecção de objetos. Precisamos apenas da transformada de wavelet da imagem original e de seu suporte de multirresolução associado, obtido por qualquer um dos métodos expostos para a tarefa.

Após a utilização do algoritmo de reconstrução para a remoção do ruído, é preciso ainda somar à imagem resultante a imagem residual da transformada *à trous*.

3.3.5 Implementação dos métodos de reconstrução

A reconstrução parcial e a total são realizadas pelo programa **OV_Reconstruct**, que toma como dados de entrada as estruturas fornecidas por OV_Atrous, OV_Support, OV_Regions e OV_Objects no caso parcial:

```
rec = ov_reconstruct(wv, sup, reg, obj, objID)
```

e apenas OV_Atrous e OV_Support no caso total:

```
rec = ov_reconstruct(wv, sup)
```

sendo `wv`, `sup`, `reg` e `obj`, respectivamente, a estrutura de wavelet, o suporte de multirresolução, a estrutura de informações sobre regiões e a estrutura de informações sobre os objetos. `objID` é um número inteiro que corresponde ao número de identificação (determinado durante a detecção de objetos) do objeto a ser reconstruído, no caso parcial, e `rec` conterá a imagem reconstruída retornada pelo programa.

Antes da aplicação dos métodos de reconstrução propriamente ditos, no caso da reconstrução parcial, é preciso construir uma estrutura de dados correspondente ao suporte individual do objeto a ser reconstruído. Para isso, o vetor de referências às regiões pertencentes ao objeto e a referência à região complementar contidos nos dados retornados pelo programa OV_Objects são utilizados. O conjunto de pontos no espaço multiescalar correspondente a essas referências é obtido a partir dos dados fornecidos por OV_Regions. A região complementar é considerada integralmente se a região principal do objeto for sua única filha. Apenas a intersecção entre as duas (tendo sido a principal projetada no nível da complementar) é tomada em caso contrário (OV_Objects fornece informações a esse respeito também, como visto na seção 3.2). O programa permite que se reconstrua simultaneamente vários objetos, sendo que nesse caso o suporte utilizado será a união dos suportes dos objetos desejados[1].

Uma vez obtido o suporte individual, recortamos ainda uma região retangular da imagem que contenha o objeto a ser reconstruído. Sem esse recorte, a reconstrução de qualquer objeto, mesmo que suas dimensões fossem pequenas, seria operada na imagem como um todo, com gastos de tempo e memória excessivos. Cada dimensão desse retângulo será um número inteiro de vezes a dimensão correspondente do suporte: três vezes quando se usa o filtro de escala linear e cinco vezes com o *spline* B_3. Essa folga no tamanho da região é necessária porque durante a reconstrução os limites do objeto se estendem devido à aplicação do operador $\tilde{P}r$ (e se estendem mais se o filtro *spline* B_3 for utilizado).

---

[1] Convém notar que a reconstrução simultânea de dois ou mais objetos não fornece, para cada um, exatamente os mesmos resultados de uma reconstrução individual. Isso se dá pelo fato de que os parâmetros que controlam a convergência e os critérios de parada (que veremos adiante), nesse caso, são aplicados globalmente. Em geral, quando se requer uma grande precisão na reconstrução, deve-se reconstruir cada objeto separadamente.

O algoritmo de reconstrução escolhido é então aplicado nessa região. O critério de parada pode ser escolhido dentre várias opções. O programa pode ser instruído a parar quando:

1. Um número fixo de iterações já foi processado.

2. A diferença entre os fluxos (soma das intensidades dos pontos da imagem) obtidos em duas iterações sucessivas é menor do que uma dada fração do fluxo obtido na última iteração.

3. No caso do *steepest descend*, α calculado para a última iteração é menor que um valor dado.

4. No caso do método do gradiente (*steepest descend*, ou com α fixo), o módulo da estrutura de wavelet residual é menor que um valor dado.

A quarta opção, a menos significativa do ponto de vista de grandezas com as quais o usuário interage diretamente, foi incluída entre as possibilidades por ser o critério mais utilizado na literatura. Vários critérios podem ser simultaneamente selecionados, de forma que o processo pára quando qualquer um deles é satisfeito.

A estrutura retornada pelo programa é uma imagem simples, com as mesmas dimensões da original (o corte mencionado acima é desfeito após o processamento), mas contendo apenas o objeto selecionado (ou vários, no caso de seleção múltipla). Nessa imagem é possível fazer medições de fluxo, posição e outros parâmetros da forma usual, já que nela há apenas o objeto que se quer medir. Os resultados assim obtidos são muito mais acurados que os que se teria através de sua medição na estrutura de wavelet projetada. Alguns testes a esse respeito são discutidos no próximo capítulo.

Exemplos de reconstrução parcial com o programa OV_Reconstruct são mostrados na figura 3.6.

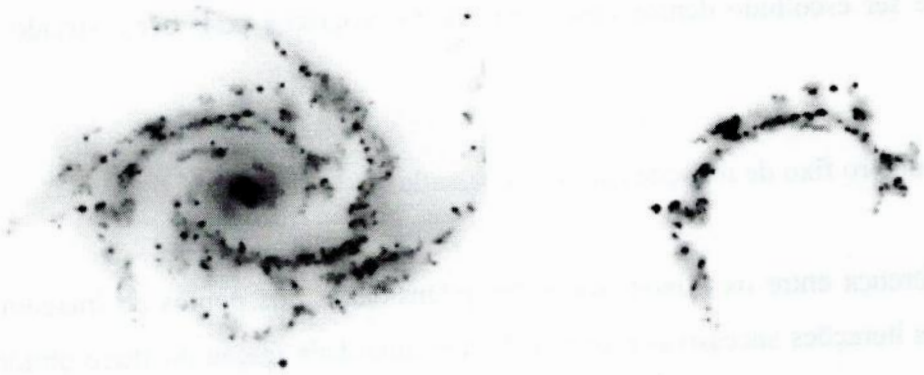


Figura 3.6: Duas reconstruções parciais da imagem da NGC 2997 utilizando o método do gradiente com o *steepest descend*. À esquerda, uma imagem com a estrutura espiral completa reconstruída. À direita temos a reconstrução de apenas uma das componentes espirais.

4 Testes e Aplicações

O que se enxerga lá do alto

Certamente uma das mais importantes motivações para o desenvolvimento dos programas descritos nos dois capítulos anteriores foi a possibilidade de vê-los aplicados a uma grande quantidade de problemas reais em astronomia. Para que um novo programa possa ser utilizado em qualquer aplicação séria, no entanto se faz necessário que seu grau de eficiência seja bem conhecido (assim como seus principais problemas). Esse tipo de avaliação é feito, em geral, através de testes em condições controladas.

Neste capítulo descrevemos os testes aos quais submetemos os programas do pacote até o momento e, em seguida, mostramos as primeiras experiências de aplicação desses programas a problemas reais em astronomia.

4.1 Testes controlados

Uma das formas mais seguras de se conhecer o nível de eficiência de uma técnica ou um algoritmo é através da realização de testes em condições perfeitamente controladas, onde se conhece cada propriedade dos dados de entrada e o resultado que se espera obter com os dados de saída. Embora ainda haja muito a ser feito nesse sentido em relação aos nossos programas, apresentamos o que já foi realizado até o presente momento.

4.1.1 Funcionalidade dos módulos básicos

Os módulos básicos de transformação *à trous*, análise de ruído e suporte de multirresolução, representados pelos programas OV_Atrous, OV_Noise, OV_Anscombe e OV_Support foram testados através de aplicações seguidas de conferência de resultados, de forma não automática em experimentos simples.

O programa OV_Atrous foi testado em relação a algumas propriedades inerentes à transformada *à trous* expostas na seção 1.2.4. Foi verificado que obedece, por exemplo, à equação de inversão

$$f = c_0 = \left(\sum_{n=1}^{N} w_n \right) + c_N \ , \tag{4.1}$$

exceto por erros mínimos, decorrentes da aritmética de ponto flutuante utilizada internamente. Uma inspeção visual também revela que os planos de wavelet obtidos possuem formas gerais coerentes com as propriedades da transformada e com exemplos gráficos encontrados na literatura. Verifica-se que a isotropia da transformada é quase perfeita com o uso do filtro *spline* B_3, enquanto que para o filtro de escala linear deteriora-se nas escalas maiores (o que já era esperado).

O programa OV_Noise foi testado a partir da aplicação em imagens artificiais com ruído de desvio padrão conhecido e da comparação desse valor com o resultado do programa, mostrando uma convergência rápida e resultados sempre muito próximos do correto.

OV_Anscombe foi testado também com imagens artificiais com parâmetros de ruído conhecidos *a priori*. Foi verificado um valor correto de desvio padrão unitário na imagem transformada, a menos de pequenas flutuações. Quando é preciso estimar um dos parâmetros da transformada generalizada o programa chega ao valor correto e com convergência rápida.

Os resultados de OV_Support foram avaliados a partir de comparações com exemplos encontrados na literatura e por inspeção e conferência dos resultados em alguns casos.

Foi realizado ainda um teste adicional do programa OV_Atrous, não quanto a sua funcionalidade, mas quanto à rapidez de seu processamento. Esse teste é importante uma vez que esse programa pode precisar ser executado até mesmo algumas centenas de vezes durante o processamento de uma única imagem, já que é utilizado internamente por vários outros programas do pacote. Experimentos muito simples, baseados em execuções sucessivas e cronometradas do programa, mostram que seu tempo de processamento é aproximadamente proporcional ao número de pontos da estrutura a ser gerada, isto é, o número de píxeis da imagem original vezes o número de planos de wavelet a serem calculados. Assim, o tempo de processamento pode ser escrito como

$$t = S_x \,.\, S_y \,.\, S_z \,.\, k \tag{4.2}$$

onde S_x e S_y são a largura e a altura em píxeis da imagem original, respectivamente, S_z é o número de planos de wavelet a ser calculado e k depende do sistema em que o programa está sendo executado e das configurações adotadas para o mesmo. Por exemplo, para um Pentium III com 600 Mhz e com a versão 5.3 da IDL em Windows 98, com o uso do filtro de escala linear e com tratamento de bordas por continuidade temos $k = 1{,}33 \times 10^{-6}\, s^{-1}$. Ou seja, o processamento de uma imagem de 500 por 500 píxeis (250.000 no total) para a obtenção de nove planos de wavelet tomaria aproximadamente três segundos.

4.1.2 Testes dos módulos de detecção e reconstrução de objetos

Testes com imagens artificiais simples foram realizados para a avaliação da eficiência dos módulos avançados. Imagens contendo um único objeto de perfil gaussiano foram criadas e ruído gaussiano foi adicionado. A largura do objeto e o nível de ruído foram selecionados de forma aleatória. As dimensões da imagem foram ajustadas segundo a largura do objeto, de forma a garantir que este sempre estivesse integralmente contido na imagem. Como indicador no nível relativo de ruído, foi adotado a razão sinal/ruído de pico (**PSNR**)[1] , calculada como a razão entre o valor do ponto de maior intensidade na imagem, antes da adição do ruído, e o desvio padrão do ruído introduzido. A esse parâmetro foi permitido assumir valores entre 10^{-3} e 10^{3}. Foram processadas 2.500 imagens com o filtro de escala linear e outras 2.500 com o filtro *spline* B_3.

Os programas OV_Regions e OV_Objects foram utilizados para a detecção das fontes simuladas, com a utilização do critério padrão para a definição dos objetos. Como níveis de ruído muito altos estavam envolvidos, em muitos casos o objeto presente na imagem não pôde ser detectado de forma satisfatória. Os resultados do

---

[1] PSNR é sigla da forma inglesa "*Peak Signal to Noise Ratio*".

processo de detecção aplicado a cada imagem artificial foram classificados em um de quatro grupos abaixo:

- **Não-detecção**: nenhum objeto foi detectado.

- **Detecção deslocada**: foi detectado apenas um objeto, mas a posição em que foi encontrado diferia da posição da fonte simulada por mais que três vezes sua largura à meia altura.

- **Detecção múltipla**: mais que um objeto foi detectado, quando apenas um estava presente na imagem simulada.

- **Detecção correta**: um único objeto foi encontrado e aproximadamente na mesma posição em que o objeto simulado foi criado (deslocado em até três vezes sua largura à meia altura).

A amostra utilizada no teste foi subdivida, para análise, em faixas de valores logarítmicos (base 10) da PSNR. Para cada uma dessas faixas calculamos a razão entre o número de detecções em cada uma das classes acima definidas e o número total de imagens simuladas na faixa. Os resultados são mostrados nas tabelas 4.1 e 4.2 (para as imagens processadas com o filtro de escala linear e com o *spline* B_3, respectivamente) e na figura 4.1.

Faixa de $\log_{10}(PSNR)$	Não-detecção	Deslocada	Múltipla	Correta
-3,0 a -2,5	93,0%	6,9%	0,0%	0,0%
-2,5 a -2,0	90,5%	9,0%	0,5%	0,0%
-2,0 a -1,5	90,2%	9,8%	0,0%	0,0%
-1,5 a -1,0	92,9%	7,1%	0,0%	0,0%
-1,0 a -0,5	78,1%	4,9%	0,4%	16,5%
-0,5 a 0	21,7%	0,1%	3,2%	74,1%
0 a 0,5	2,4%	0,0%	3,4%	93,7%
0,5 a 1,0	0,0%	0,0%	1,9%	98,0%
1,0 a 1,5	0,0%	0,0%	1,9%	98,1%
1,5 a 2,0	0,0%	0,0%	2,7%	97,3%
2,0 a 2,5	0,0%	0,0%	1,1%	98,9%
2,5 a 3,0	0,0%	0,0%	2,5%	97,5%

Tabela 4.1: Resultado da detecção dos objetos simulados com o uso do filtro de escala linear.

Faixa de $\log_{10}(PSNR)$	Não-detecção	Deslocada	Múltipla	Correta
-3,0 a -2,5	87,5%	12,5%	0,0%	0,0%
-2,5 a -2,0	89,9%	8,9%	1,2%	0,0%
-2,0 a -1,5	87,8%	11,8%	0,5%	0,0%
-1,5 a -1,0	90,9%	9,1%	0,0%	0,0%
-1,0 a -0,5	67,4%	8,1%	3,1%	21,3%
-0,5 a 0	20,9%	1,9%	8,0%	69,2%
0 a 0,5	2,5%	0,8%	6,2%	90,5%
0,5 a 1,0	0,0%	0,0%	2,8%	97,2%
1,0 a 1,5	0,0%	0,0%	3,1%	96,9%
1,5 a 2,0	0,0%	0,0%	2,1%	97,9%
2,0 a 2,5	0,0%	0,0%	0,9%	99,1%
2,5 a 3,0	0,0%	0,0%	0,9%	99,1%

Tabela 4.2: Resultado da detecção dos objetos simulados com o uso do filtro *spline* B_3.

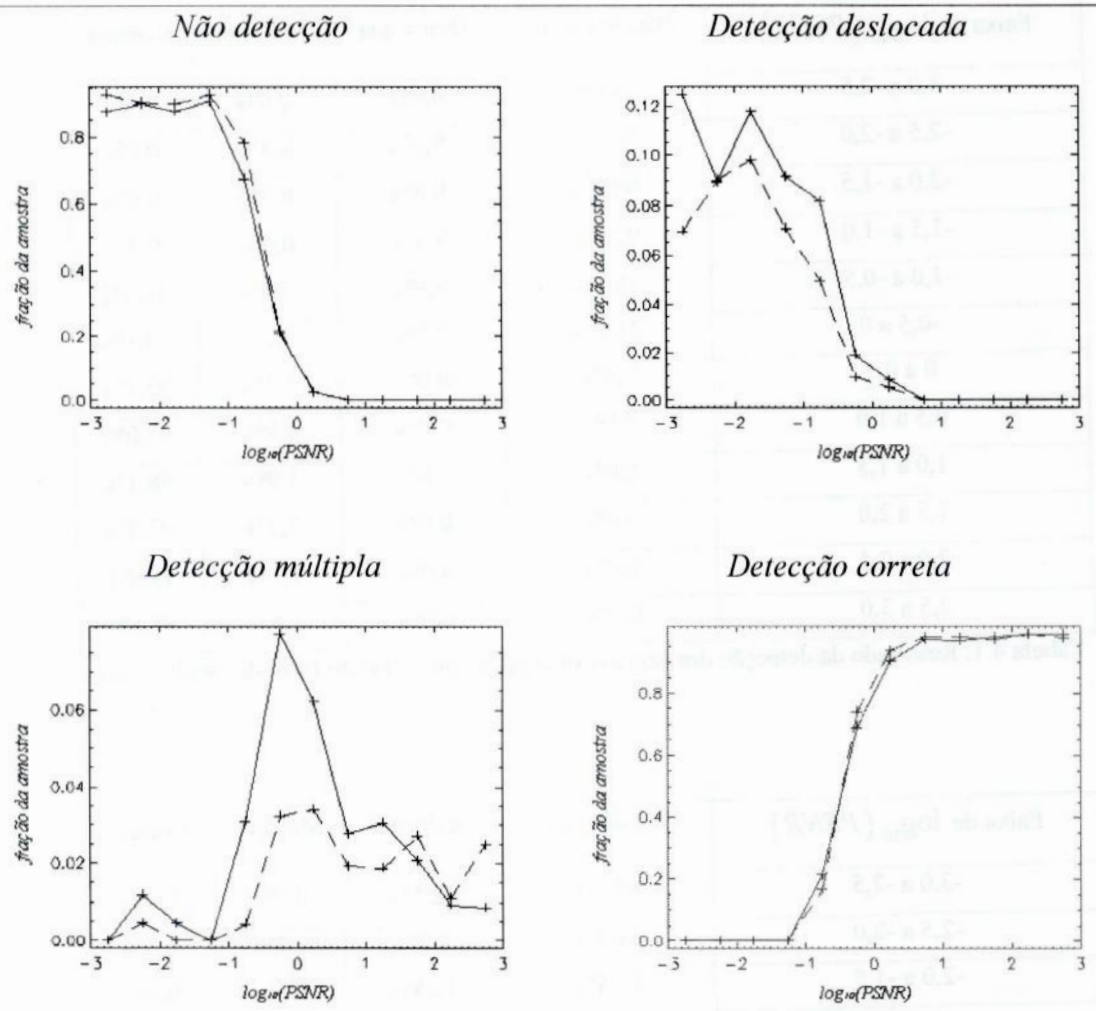


Figura 4.1: Fração de cada condição de detecção em função do logaritmo em base 10 da PSNR. Não detecção (em cima, à esquerda), detecção múltipla (em cima, à direita), detecção deslocada (em baixo, à esquerda) e detecção correta (em baixo à direita), que mostra a eficiência geral dos programas de detecção. A linha tracejada refere-se aos dados obtidos com o filtro de escala linear, enquanto a sólida ao que se obtém com o *spline* B_3.

Os resultados demonstram que os dois filtros podem ser utilizados na tarefa de detecção com praticamente a mesma eficiência. É natural que nas imagens com PSNR maior os objetos sejam detectados com mais freqüência, já que o ruído é um dos principais (e, nesse experimento controlado, o único) obstáculo ao processo de detecção. Observamos que para PSNRs menores que 1 (logaritmo negativo), que indicam um regime em que o ruído domina sobre o sinal na imagem, freqüentemente não é encontrado objeto algum e, quando o é, está quase sempre muito deslocado de sua posição verdadeira. Os casos de detecção múltipla ocorrem com alguma freqüência para

PSNRs intermediárias. As detecções múltiplas não indicam a não detectabilidade do objeto simulado, mas a presença de objetos espúrios (inexistentes, detectados por engano). O mecanismo que leva a essa falsa detecção ainda não está muito bem determinado, mas está sendo explorado no momento (na verdade, o critério restrito para a detecção, explicado na seção 3.2, foi introduzido por nós para, entre outras coisas, reduzir o problema das detecções espúrias; mas por ter sido um dos elementos mais recentemente implementados no pacote, ainda não foi possível verificar sua eficiência nesse aspecto através de testes como o que descrevemos aqui).

Nos casos em que se obteve uma detecção correta, o programa OV_Reconstruct foi empregado para a obtenção da imagem reconstruída de cada fonte simulada. O fluxo total dessas fontes reconstruídas foi medido e convertido para uma escala de magnitudes[1] e, então comparado com a magnitude das fontes simuladas. Esse tipo de análise nos permite avaliar a confiabilidade fotométrica da técnica. Na figura 4.2 podemos ver a dispersão dos desvios em magnitude em função da PSNR obtidos com o uso do filtro de escala linear e com o *spline* B_3.

---

[1] A magnitude adotada aqui, a magnitude instrumental, é definida como

$$m = -2{,}5 \log_{10}(F),$$

onde F é o fluxo total do objeto.

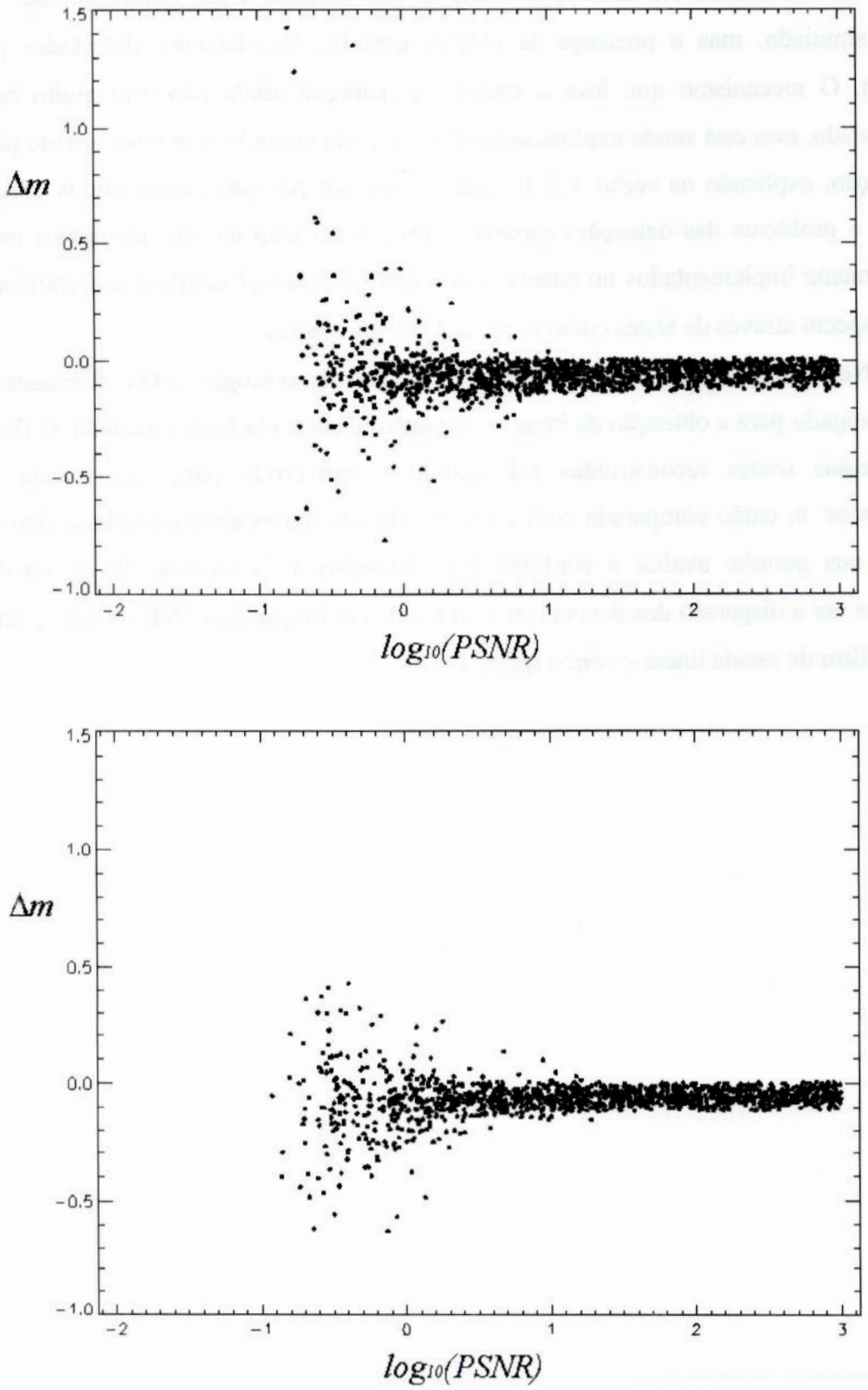


Figura 4.2: Diferença entre a magnitude original e a reconstruída em função da PSNR para cada imagem processada com o filtro de escala linear (em cima) e com o *spline* B_3 (em baixo).

Verifica-se uma certa tendência à obtenção de um fluxo superestimado (ou uma magnitude subestimada) com a reconstrução. Os desvios de magnitude são maiores para PSNRs menores do que 1, o que já era esperado. Para um estudo mais completo desses dados, obtemos o desvio médio da magnitude e o desvio padrão da magnitude reconstruída para cada uma das faixas de PSNR definidas para o estudo da detecção das fontes. Os resultados são mostrados na tabela 4.2 e na figura 4.3.

Faixa de $\log_{10}(PSNR)$	Escala linear		*Spline* B_3	
	μ	σ	μ	σ
-3,0 a -2,5	-	-	-	-
-2,5 a -2,0	-	-	-	-
-2,0 a -1,5	-	-	-	-
-1,5 a -1,0	-	-	-	-
-1,0 a -0,5	0,038	0,417	-0,057	0,240
-0,5 a 0	-0,046	0,205	-0,109	0,155
0 a 0,5	-0,066	0,094	-0,083	0,092
0,5 a 1,0	-0,070	0,056	-0,066	0,043
1,0 a 1,5	-0,062	0,040	-0,061	0,034
1,5 a 2,0	-0,053	0,038	-0,052	0,027
2,0 a 2,5	-0,049	0,035	-0,056	0,030
2,5 a 3,0	-0,048	0,034	-0,056	0,028

Tabela 4.3: Desvio médio da magnitude (μ) e o desvio padrão da magnitude reconstruída (σ) para as reconstruções com filtro linear de escala e *spline* B_3. As células com um traço indicam que o valor não pôde ser obtido por não haver objetos corretamente detectados na faixa correspondente (ver tabelas 4.1 e 4.2).

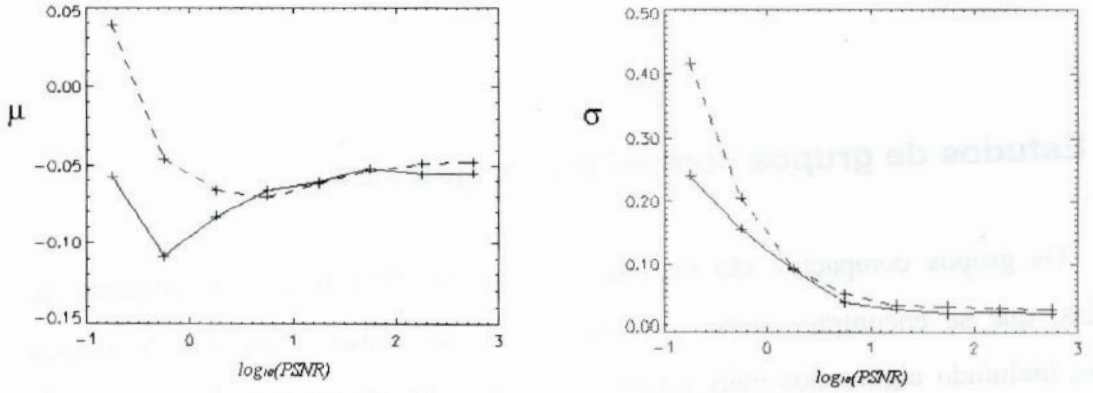


Figura 4.3: Desvio médio da magnitude (μ) e o desvio padrão da magnitude reconstruída (σ) para as reconstruções com filtro linear de escala (linha tracejada) e *spline* B_3 (linha sólida).

Esses resultados indicam um pequeno desvio sistemático dos valores de magnitudes dos objetos reconstruídos. O desvio padrão, que nos dá uma medida dos erros não sistemáticos envolvidos na reconstrução, decresce com um aumento da PSNR, como esperado, e o uso do filtro *spline* B_3 aparentemente reduz seu valor para todos os níveis de PSNR, permitindo, portanto, reconstruções fotometricamente mais precisas.

Se considerarmos que os valores obtidos para o desvio da magnitude são universais, então podemos corrigir a magnitude obtida com a reconstrução de qualquer objeto, desde que conheçamos a PSNR correspondente à situação do mesmo na imagem. No caso dos testes que acabamos de apresentar, esse valor já era conhecido *a priori* para cada objeto, mas em aplicações reais, somos obrigados a adotar um valor estimado. Uma boa opção é considerar a PSNR como sendo a razão entre o valor máximo da imagem reconstruída do objeto e o valor do desvio padrão do ruído obtido da imagem através do programa OV_Noise, descrito na seção 2.3. Os experimentos que realizamos até o momento não são, no entanto, suficientes para que determinemos fatores de correção a serem aplicados a qualquer objeto e em qualquer imagem astronômica.

Os desvios padrão encontrados nos dão uma medida da precisão dos valores obtidos pela reconstrução. No estágio atual do desenvolvimento do programa, somos capazes de obter flutuações menores do que 0,1 magnitude (um erro fotométrico típico para aplicações desse tipo) para PSNRs maiores que 1.

Convém notar que o mesmo não se pode dizer do desvio máximo ocorrido, que situa-se na faixa de 0,2 magnitudes mesmo a PSNRs altas. Estratégias para a redução desse efeito indesejável ainda estão em estudo.

4.2 Estudos de grupos compactos de galáxias

Os grupos compactos são estruturas compostas de um número pequeno de galáxias, que se encontram muito próximas umas das outras. Uma centena desses grupos, incluindo alguns dos mais estudados, foram catalogados por Hickson (1982) com base em critérios de seleção bem definidos, e são, por isso, denominados grupos

compactos de Hickson (HCGs) . Posteriormente, medidas das velocidades radiais[1] de todas as galáxias pertencentes a cada um desses grupos foram feitas (Hickson *et al.*, 1992). As galáxias foram então classificadas em **concordantes** ou **discordantes**. As concordantes possuíam uma velocidade radial que não diferia da velocidade mediana do grupo por mais que $1.000\ km/s$, com uma grande probabilidade de pertencerem realmente ao grupo, enquanto as discordantes, que não atendiam a esse critério, poderiam simplesmente estar na mesma direção das outras do grupo, mas sem qualquer relação física com elas. Verificou-se que 92 dos 100 grupos pertencentes ao catálogo possuíam pelo menos três galáxias concordantes, e desses, 69 possuíam quatro ou mais galáxias concordantes.

4.2.2 Luz difusa em grupos compactos

A grande proximidade entre as galáxias em um grupo compacto faz com estejam constantemente interagindo por forças de maré. Com o passar do tempo, as interações ocorridas causam a perda de quantidade considerável de matéria galática; matéria que passa a vagar pelo grupo como um todo, associada ao potencial gravitacional integrado do mesmo. Como parte dessa matéria é de natureza estelar, essa massa liberada pode ser observada sob a forma de uma tênue luz difusa à qual sobrepõem-se os sinais das próprias galáxias do grupo. Acredita-se que a estrutura de luz difusa presente em um grupo compacto esteja intimamente ligada a sua história evolutiva.

Tipicamente, a componente de luz difusa presente em uma imagem de grupo compacto é de detecção muito difícil, dada a sua configuração tênue e pouco concentrada em contraste com a forma relativamente compacta e luminosa como as galáxias do grupo se apresentam. Exatamente por essa diferença de escala entre as galáxias e a luz difusa é que um procedimento de reconstrução multiescalar nos deve permitir obter isoladamente a contribuição luminosa de cada um desses elementos.

---

[1] Velocidades na direção radial (aproximação/afastamento), do ponto de vista de um observador na Terra.

4.2.3 Aplicação da técnica de reconstrução

Utilizamos os programas do nosso pacote para a reconstrução da componente difusa de dois grupos compactos de Hickson: o grupo VV179 (HCG55) e o Sexteto de Seyfert (HCG79), o segundo grupo compacto descoberto. Também realizamos uma análise superficial de alguns outros grupos em busca de indícios de luz difusa.

Para trabalhar as imagens dos grupos compactos e tentar separar a luz das galáxias da luz difusa aplicamos o programa OV_Reconstruct (após todas as outras etapas necessárias). Foi utilizado o método do gradiente com passo fixo ($\alpha = 1$) com 20 iterações e as transformações *à trous* foram feitas com filtro de escala linear. Isolamos a luz das galáxias e de outras componentes compactas progressivamente, até que fosse possível detectar uma componente difusa. Reconstruindo a componente difusa, pudemos calcular a razão entre a luz difusa e a luz total do grupo (difusa + galática).

Tivemos o cuidado de não considerar como parte da luz do grupo aquela pertinente a galáxias discordantes.

O percentual de luz difusa para a luz total dos grupos é mostrado aqui para cada banda espectral em que uma imagem do grupo estivesse disponível (as bandas seguem o sistema UBV estendido):

Banda	HCG55	HCG79
R	27,6%	42,4%*
B	34,4%	11,7%
U	0%	-

Tabela 4.4: Frações de luz difusa obtidas através do algoritmo de reconstrução parcial.

* Ver texto a seguir.

Para HCG79, não havia imagem disponível na banda U. Já em HCG55, não foi possível detectar qualquer traço de luz difusa nessa banda, de forma que se houver alguma, é tão fraca que não é possível detectá-la com a presente técnica. Uma análise posterior detalhada das estruturas reconstruídas revelou que a imagem utilizada para o estudo de HCG79 na banda R possuía problemas de redução que se apresentavam como estruturas positivas com a mesma escala de tamanho que a componente de luz difusa, de forma que a reconstrução desta estava contaminada com fluxo espúrio. Assim, a fração

medida nessa imagem deve consistir, na verdade, num limite superior para a contribuição da luz difusa.

As formas das componentes difusas reconstruídas desses dois grupos podem ser vistas, ao lado das imagens originais, nas figuras 4.4 e 4.5.

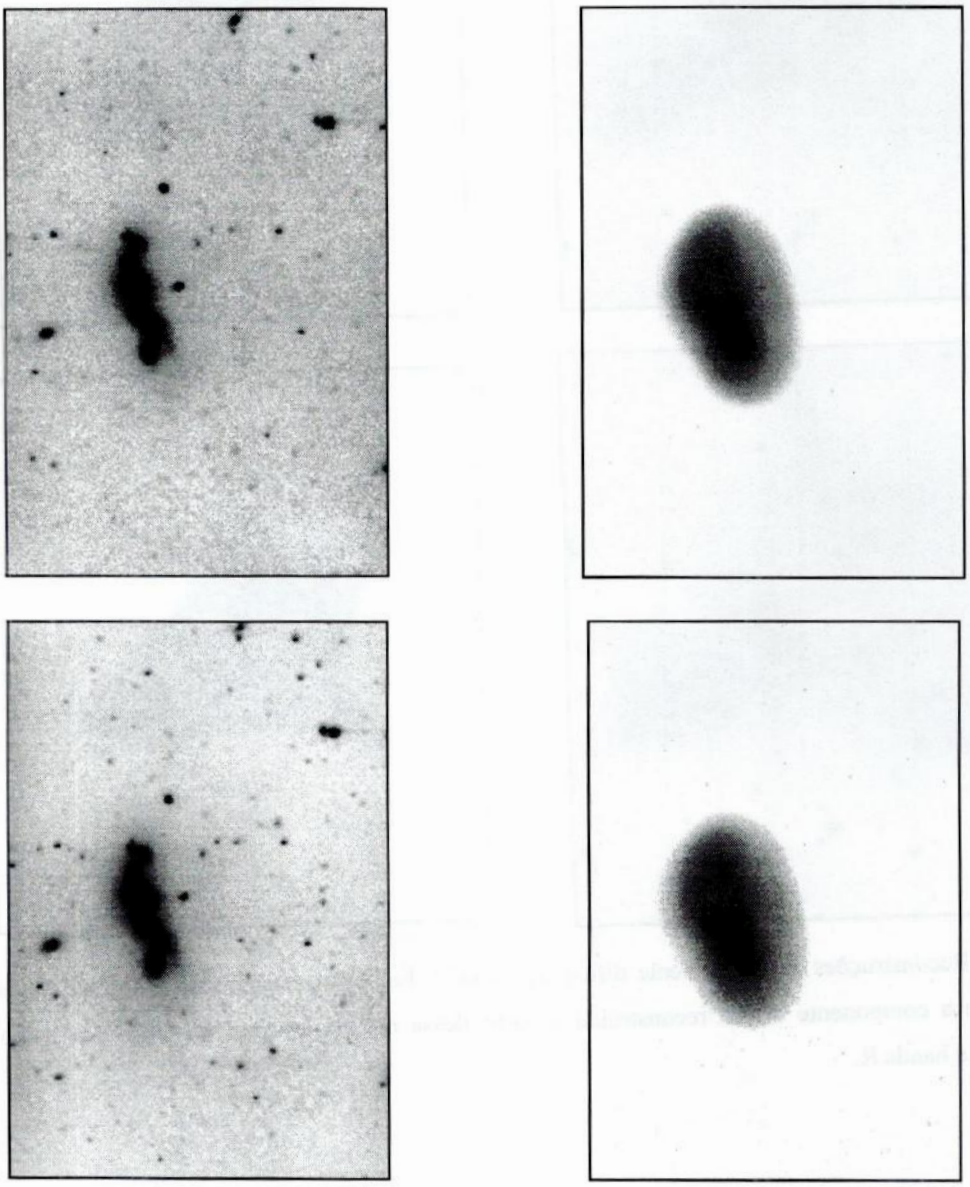


Figura 4.4: Reconstruções da componente difusa de HCG55. Em cima, imagem do grupo na banda B (esquerda) e a componente difusa reconstruída a partir dessa mesma imagem (direita). Em baixo, o mesmo para a banda R.

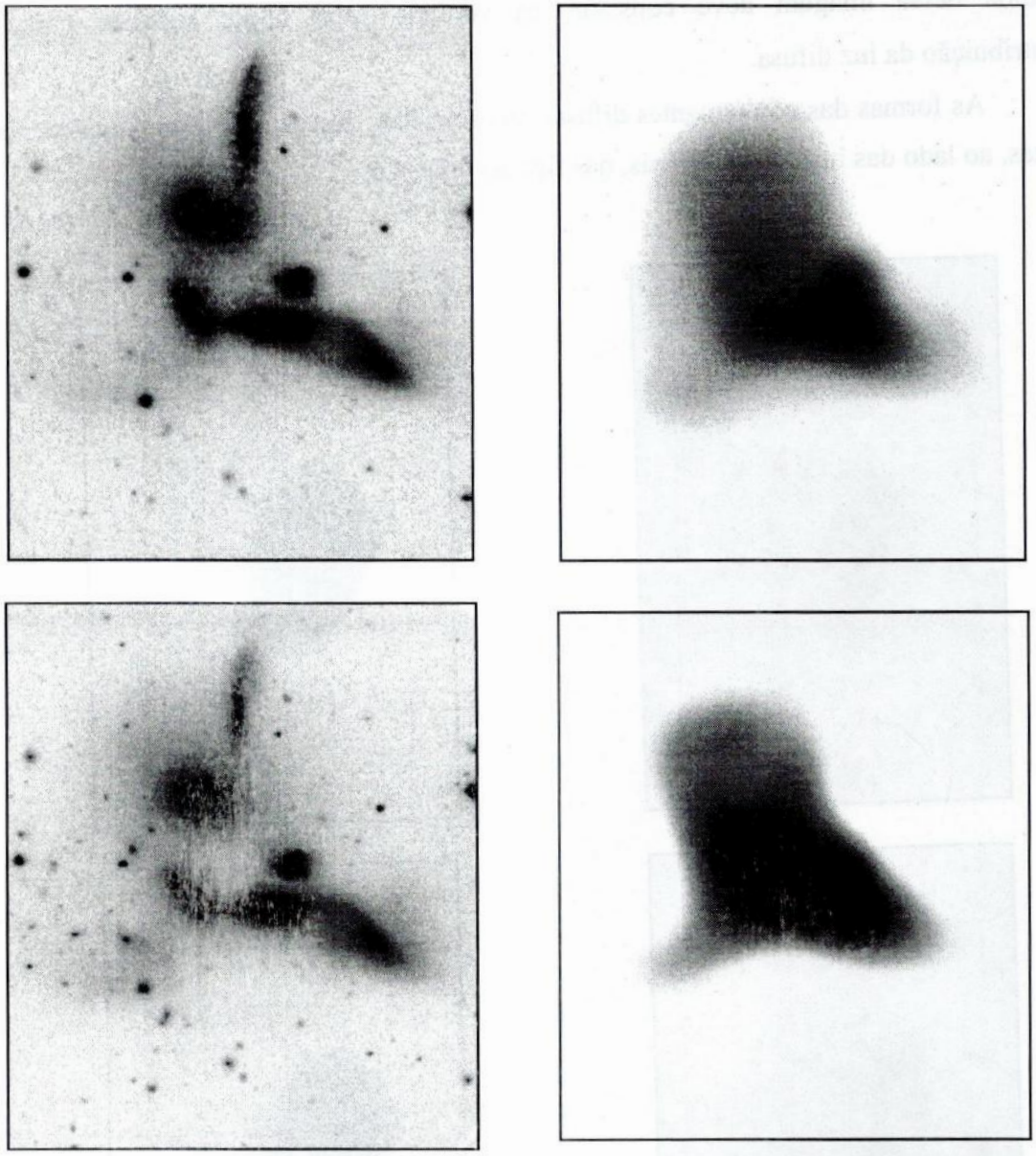


Figura 4.5: Reconstruções da componente difusa de HCG79. Em cima, imagem do grupo na banda B (esquerda) e a componente difusa reconstruída a partir dessa mesma imagem (direita). Em baixo, o mesmo para a banda R.

Analisamos também, outros 15 grupos de Hickson em busca de traços de luz difusa. Infelizmente as imagens desses grupos às quais tivemos acesso eram muito antigas e de má qualidade para os padrões modernos, de tal maneira que apenas um estudo muito superficial pôde ser realizado. Por esse mesmo motivo, também não podemos descartar a possibilidade da presença de luz difusa em qualquer dos grupos pesquisados, mesmo que não tenha sido possível encontrá-la (de fato, Da Rocha (2002) pôde encontrar uma componente de luz difusa em HCG95 – um dos grupos que

descartamos a princípio por não dispormos uma imagem de qualidade adequada – em um trabalho comentado a seguir). Talvez, com uma futura melhora no processo, mesmo essas imagens possam nos revelar mais informação.

Esses 15 grupos foram selecionados por apresentarem indícios de forte interação gravitacional, como deformações possivelmente causadas pelas forças de maré ou ligações entre os envoltórios de dois ou mais membros. Esses grupos pré-selecionados foram HCG5, HCG8, HCG17, HCG24, HCG33, HCG50, HCG54, HCG56, HCG64, HCG65, HCG74, HCG75, HCG81, HCG95 e HCG96. Alguns desses foram descartados num estado inicial do processo de análise porque a baixa qualidade da imagem não permitiu a continuação do mesmo. Em outros, nenhuma estrutura difusa se apresentava. Os três grupos que mostraram características interessantes foram HCG17, HCG54 e HCG74.

Em HCG17 obtivemos um resultado bastante impressionante. Uma estrutura difusa com características geométricas bem independentes dos elementos do grupo foi encontrada, como vemos na figura 4.6.

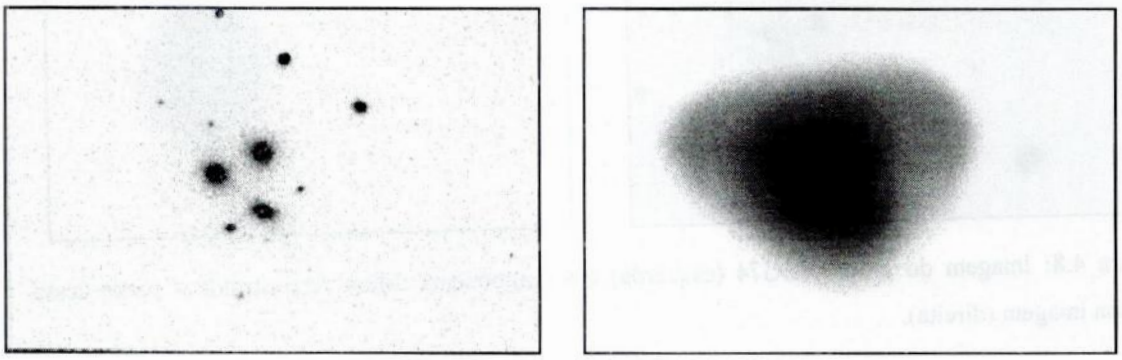


Figura 4.6: Imagem do grupo HCG17 (esquerda) e a componente difusa reconstruída a partir dessa mesma imagem (direita)

Em HCG54 não foi possível separar totalmente a estrutura de fundo dos envoltórios das galáxias ali presentes, mas caudas de maré difusas são bem visíveis, mesmo na imagem original, o que nos leva a acreditar que haja um tênue halo difuso no qual o grupo está imerso, que, no entanto, só poderia ser visto claramente com o uso de uma imagem de maior resolução e menos ruidosa. A melhor separação obtida é mostrada na figura 4.7.

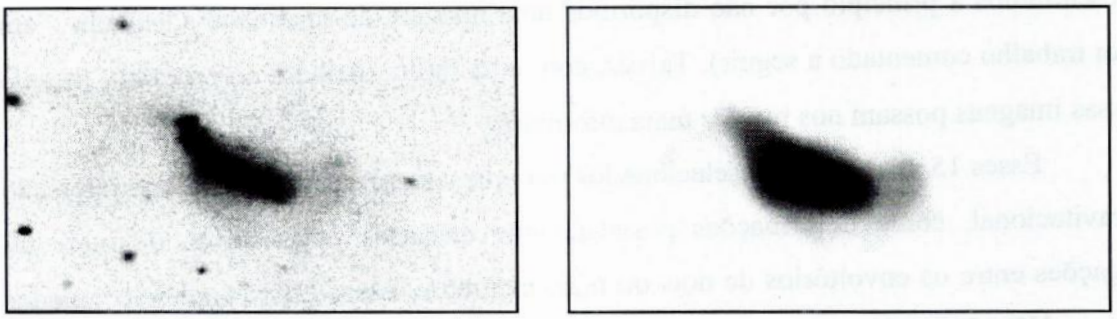


Figura 4.7: Imagem do grupo HCG54 (esquerda) e a tentativa de isolamento da componente difusa a partir dessa mesma imagem (direita)

Em HCG74 encontramos uma componente difusa na região mais densa do grupo, como mostrado na figura 4.8. É importante notar que o centróide da estrutura difusa está ligeiramente deslocado daquele das galáxias próximas.

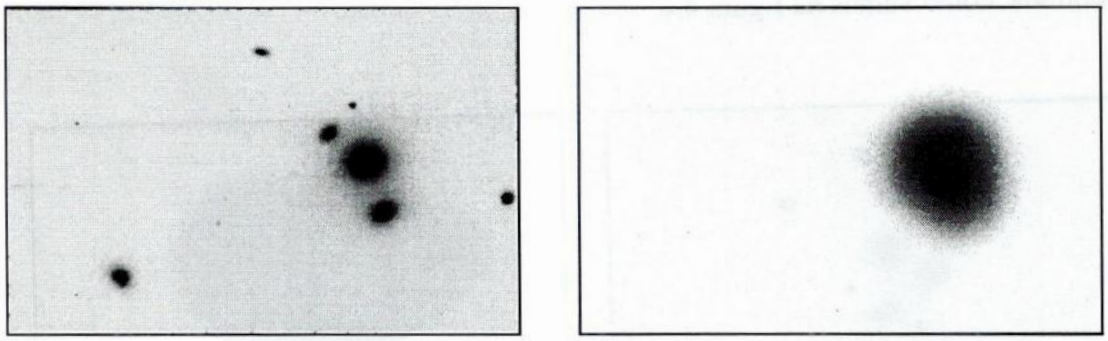


Figura 4.8: Imagem do grupo HCG74 (esquerda) e a componente difusa reconstruída a partir dessa mesma imagem (direita).

Uma análise semelhante, embora mais detalhada, foi feita por Da Rocha (2002) em três grupos de Hickson, incluindo HCG 79 e HGC95, também com o uso do nosso pacote de programas.

Existe atualmente um regime de colaboração entre nós e os pesquisadores Cláudia Mendes de Oliveira e Cristiano Da Rocha, do Instituto de Astronomia e Geofísica da USP, para o estudo de temas relacionados à estrutura de HCGs. Os últimos resultados desse trabalho conjunto, no que toca a análise da luz difusa nesses grupos foi publicado em Mendes de Oliveira *et al.* (2002).

4.3 Subestruturas em nebulosas planetárias e AGBs extremas

Um outro projeto envolvendo a utilização de nossos programas, em colaboração com os pesquisadores Silvia Lorenz-Martins e François Cuisinier, do Observatório do Valongo/UFRJ, consiste na aplicação dos programas desenvolvidos a imagens de estruturas de nebulosas planetárias e estrelas AGBs extremas (essencialmente as OH/IR, as carbonadas extremas e as pós-AGBs).

Através da detecção de subestruturas tênues nesses objetos pretendemos determinar as conexões evolutivas entre as nebulosas planetárias e as AGBs extremas, já que existem indícios de que algumas estruturas das nebulosas planetárias iniciam sua formação ainda na fase de AGB extrema.

Pretendemos ainda, através de uma visão multiescalar, estabelecer uma classificação morfológica quantitativa e sistemática das nebulosas planetárias, tendo como bases as propriedades de suas subestruturas.

Esse projeto encontra-se ainda em um estágio inicial, mas as primeiras investigações e perspectivas a seu respeito podem ser vistas em Rabaça *et al.* (2001).

5 Conclusões e Perspectivas

O que terminou e o que está por vir

Temos em nossas mãos um pacote funcional para a análise de imagens astronômicas sob uma perspectiva multiescalar. Em essência, nosso trabalho está concluído e com resultados satisfatórios. Há, no entanto, uma série de pequenos detalhes que precisam ainda ser acertados antes que nossos programas possam ser distribuídos à comunidade científica. Existem ainda algumas idéias mais ambiciosas, que consumiriam mais tempo e trabalho para serem desenvolvidas, e que poderiam ser implementadas em uma versão posterior do pacote. Neste último capítulo apresentamos nossa visão sobre os progressos atingidos nos vários tópicos trabalhados e descrevemos o que deverá ser feito ou investigado no futuro.

5.1 Aspectos teóricos e programas principais

As técnicas de análise multiescalar através da transformada de wavelet demonstraram, ao longo do desenvolvimento deste trabalho, serem de grande valor para o tratamento de diversos tipos de problemas no processamento de dados astronômicos. Os programas desenvolvidos podem ser empregados para muitos outros problemas além daqueles abordados diretamente neste trabalho.

Todo o trabalho de elaboração dos programas principais do código já foi concluído, de forma que não serão necessárias alterações em seu conteúdo até a época de distribuição do pacote. Existem ainda, no entanto, várias idéias de melhoria e expansão dos programas que podem ser incorporadas no futuro. Entre elas estão a implementação de versões da transformada discreta de wavelet aplicadas a dados em múltiplos canais (o que já é feito por um módulo do MR, através de PCA[1]) e a

[1] PCA é sigla de "*Principal Components Analysis*" – "análise de componentes principais". É uma técnica tradicional de análise de dados em múltiplos canais.

adaptação das rotinas desenvolvidas para aplicação a dados espectrais unidimensionais e a imagens em raios-x.

5.2 Testes e aplicações

Os testes realizados até agora demonstram um funcionamento adequado de todos os programas desenvolvidos e a aplicação a grupos compactos de galáxias forneceu resultados interessantes. Porém, como comentado no Capítulo 4, muitos dos elementos mais recentemente incluídos no pacote não foram ainda submetidos a testes ou aplicados a problemas reais.

A aplicação de testes completos a todos os programas do pacote, com suas várias possibilidades de configuração seria por si só um projeto de proporções consideráveis. Um novo projeto que iniciamos em colaboração com Cláudia Mendes de Oliveira e Cristiano Da Rocha do IAG/USP consiste em uma comparação dos parâmetros obtidos com nossos programas de detecção e reconstrução com resultados obtidos com o programa SExtractor[1] em imagens artificiais.

Os projetos iniciados com a colaboração de outros pesquisadores, mencionados no Capítulo 4, envolvendo o estudo de grupos compactos de galáxias e nebulosas planetárias (assim como suas progenitoras), continuam em andamento e esperamos que produzam resultados importantes em breve.

5.3 Apresentação e distribuição do pacote

Para que o pacote OV_WAV possa ser utilizado por um grande número de pesquisadores, é fundamental que esteja bem documentado. A documentação do programa deverá ser feita o quanto antes e deverá conter uma parte considerável das informações presentes neste documento.

---

[1] O SExtractor (Bertin e Arnouts, 1996) é um programa para a detecção e obtenção das propriedades de fontes em uma imagem. Seu uso tem sido cada vez mais difundido na comunidade astronômica.

Outra preocupação é quanto à facilidade de utilização. Presentemente, todos os programas do pacote devem ser executados a partir de uma linha de comando IDL e os recursos de visualização de dados não são suficientes para uma fácil interpretação dos resultados obtidos a cada etapa do processamento. Há algum tempo planejamos a incorporação ao pacote de rotinas que integrem todos os programas e os apresentem ao usuário de forma visual e interativa. O modo de execução por linha de comando continuaria existindo, para utilização em ambientes em que elementos gráficos não estão disponíveis e para chamada direta por outros programas. Planejamos incluir também alguns *scripts* em linha de comando, que integrem dois ou mais programas do pacote, agilizando a execução de tarefas comuns.

Apêndices

A – Convolução e Filtros Digitais

A convolução e sua aplicação à teoria de filtros digitais são explorados em certo grau neste documento e, portanto, para facilitar o entendimento do mesmo por aqueles que não estão familiarizados com esses tópicos, faremos aqui uma breve revisão. Para uma explicação mais detalhada sobre esses tópicos e muitos outros fundamentais ao estudo do processamento digital de imagens, veja Niblac (1986).

A.1 Convolução

A **convolução** é uma operação matemática que atua entre duas funções. Para funções unidimensionais (de uma variável) é dada por

$$g(x) = \int_{-\infty}^{+\infty} f(t)\, h(x-t)\, dt, \qquad \text{(A.1)}$$

onde g é dita a convolução entre f e h ou a convolução de f por h (ou de h por f, já que é uma operação comutativa).

Geralmente escreve-se

$$g = f * h\ . \qquad \text{(A.2)}$$

O resultado da operação é uma função que pode ser entendida como uma versão de f suavizada, de forma que cada ponto é "espalhado" segundo a forma de h, ou vice-versa.

A.2 Convolução Digital

A **convolução digital**, ou convolução discreta, em uma dimensão é dada naturalmente por

$$g(x) = \sum_{t=-\infty}^{+\infty} f(t)\ h(x-t)\ , \tag{A.3}$$

onde f, g e h são funções de uma variável discreta. O somatório deve feito de $-\infty$ a $+\infty$, mas em geral a função h pode ser compacta, isto é, igual a zero em todo domínio, exceto para valores entre $-l$ e $+l$, sendo l um inteiro qualquer. Assim, o número de termos do somatório fica reduzido:

$$g(x) = \sum_{t=x-l}^{x+l} f(t)\ h(x-t)\ . \tag{A.4}$$

No caso de sinais bidimensionais, como imagens, por exemplo, temos

$$g(x,y) = \sum_{a=-\infty}^{+\infty} \sum_{b=-\infty}^{+\infty} f(a,b)\ h(x-a, y-b) \tag{A.5}$$

ou, no caso de h compacto,

$$g(x,y) = \sum_{a=x-l_x}^{x+l_x} \sum_{b=y-l_y}^{y+l_y} f(a,b)\ h(x-a, y-b)\ . \tag{A.6}$$

A.3 Filtros Digitais

O termo "filtro" é aplicado em processamento de imagens a uma gama enorme de operações, envolvendo técnicas e ferramentas de muitas espécies. De uma forma geral, podemos dizer que um filtro é uma operação que transforma uma imagem em outra (ou um sinal em outro, de uma forma geral). Tudo o mais que se possa dizer sobre o processo é dependente do filtro aplicado.

Muitas espécies de filtros obtêm o valor de cada píxel da nova imagem a partir dos valores na vizinhança do píxel de mesma posição na imagem original (geralmente incluindo o próprio). O filtro mediano e os filtros de passa-baixas descritos nas seções 1.3.4 e 1.4.1 são exemplos de filtros de suavização que atuam dessa forma.

Os filtros em que o valor do novo píxel é dado por uma combinação linear dos valores na vizinhança do píxel correspondente, sendo as constantes multiplicativas as mesmas ao longo da imagem e atribuídas com base na posição relativa de cada ponto, são chamados **filtros lineares**. A operação de aplicação de um filtro linear digital pode ser escrita em termos de uma convolução digital em que a função h caracteriza o filtro. Como h tende a ser compacta, é comum a notação

$$h(x,y) \longleftrightarrow \begin{bmatrix} h(-l_x,-l_y) & \cdots & h(0,-l_y) & \cdots & h(l_x,-l_y) \\ \vdots & & \vdots & & \vdots \\ h(-l_x,0) & \cdots & h(0,0) & \cdots & h(l_x,0) \\ \vdots & & \vdots & & \vdots \\ h(-l_x,l_y) & \cdots & h(0,l_y) & \cdots & h(l_x,l_y) \end{bmatrix} \tag{A.7}$$

para filtros em duas dimensões. Geralmente a matriz acima é associada diretamente ao filtro na notação, e é utilizada para identificá-lo.

Filtros lineares também podem ser aplicados a sinais unidimensionais, sendo sua aplicação dada por (A.4) e a representação de seus coeficientes dada por

$$h(x) \longleftrightarrow \begin{pmatrix} h(-l) & \cdots & h(0) & \cdots & h(l) \end{pmatrix}. \tag{A.8}$$

Se um filtro bidimensional é tal que sua matriz de coeficientes pode ser escrita como o produto matricial de um vetor coluna por seu transposto:

$$h(x,y) \longleftrightarrow v \times v^T, \tag{A.9}$$

então dizemos que esse filtro é **separável**. A aplicação de filtros separáveis pode ser feita de forma unidimensional linha por linha, e então coluna por coluna, produzindo o mesmo resultado que uma aplicação direta produziria. Essa propriedade dos filtros separáveis favorece uma implementação computacional eficiente.

B – Distribuições Gaussiana e de Poisson

Faremos aqui uma revisão muito breve de duas distribuições de probabilidades que caracterizam os modelos de ruído utilizados em nossas aplicações. Para um estudo completo sobre estes e outros tópicos em probabilidade e estatística, procure, por exemplo, Meyer (1976).

B.1 Distribuição Gaussiana

A distribuição de probabilidades **gaussiana**, também conhecida como **distribuição normal**, é uma distribuição contínua, dada por

$$f(x) = \frac{1}{\sqrt{2\pi}\ \sigma}\ e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}, \tag{B.1}$$

onde μ pode ser qualquer real e σ qualquer real positivo. Se uma variável aleatória X possui densidade de probabilidades gaussiana, então a probabilidade de que tenha um valor entre x_1 e x_2 vale

$$P(x_1 \le X \le x_2) = \int_{x_1}^{x_2} \frac{1}{\sqrt{2\pi}\ \sigma}\ e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx \tag{B.2}$$

e a média e o desvio padrão associados a X são dados diretamente por μ e σ, respectivamente.

Uma das propriedades mais importantes dessa distribuição é a sua invariância a transformações lineares. Se somarmos a uma constante ou multiplicarmos por uma constante uma variável aleatória de distribuição gaussiana, obteremos uma outra variável gaussiana. Combinações lineares de variáveis gaussianas também geram outras variáveis gaussianas.

B.2 Distribuição de Poisson

A **distribuição de Poisson** é uma distribuição de probabilidades para variáveis discretas. Sua função densidade de probabilidades é

$$f(x) = \frac{e^{-\alpha}\alpha^{x}}{x!}, \tag{B.3}$$

onde α deve ser um inteiro maior que 0. A probabilidade, então, de que uma certa variável aleatória X com distribuição de Poisson assuma um valor x_0, é dada por

$$P(X = x_0) = \frac{e^{-\alpha}\alpha^{x_0}}{x_0!}, \tag{B.4}$$

e sua média será dada por

$$\mu = \alpha, \tag{B.5}$$

enquanto seu desvio padrão será

$$\sigma = \sqrt{\alpha}. \tag{B.6}$$

C – Resumo da Notação

Apresentamos aqui uma listagem de referência com os símbolos e expressões mais importantes utilizados ao longo deste documento, para consulta rápida em caso de dificuldade de interpretação de alguma passagem. Elementos que são definidos ou explicados em algum ponto deste documento possuem referências de número de página junto à sua identificação.

C.1 Variáveis e estruturas de dados

x, y	Coordenadas espaciais.
ω	Freqüência angular.
z, a	Escala (p.14).
μ	Média.
σ	Desvio padrão.
$f(x)$, f	Função arbitrária, função original.
$f(x,y)$, f	Imagem original.
w	Conjunto de planos de wavelet, também chamado estrutura de wavelet (p.21)
w_n	Plano de wavelet (p.21, 26).

$w_z(x,y)$, $w(x,y,z)$	Ponto no espaço de wavelet (p.21).
S	Suporte de multirresolução (p. 51).
$S_z(x,y)$, $S(x,y,z)$	Ponto no suporte de multirresolução (p. 51).
σ_I	Desvio padrão do ruído na imagem (p. 45).
σ_n	Desvio padrão do ruído no plano de wavelet de escala *n* (p. 44).
σ_G	Desvio padrão da componente gaussiana do ruído (p. 47).
v	Valor original do sinal (p. 51).
v_F	Valor filtrado do sinal (p. 51).
E_1	Evidência de magnitude (p. 56).
E_2	Evidência de correlação (p. 57).
E	Evidência combinada (p.58).
$Corr_z$	Correlação bruta entre planos de wavelet (p. 57).

$Ncorr_z$	Constante de normalização da correlação bruta (p. 57).
CN_z	Correlação normalizada entre planos de wavelet (p. 57).
w_F	Estrutura de wavelet filtrada do ruído (p. 62).
f_F	Imagem filtrada do ruído (p. 62).
S_O	Suporte individual do objeto O (p.76)
Z_O	Suporte externo do objeto O (p. 78)
w_P	Estrutura de wavelet projetada em um suporte individual (p. 76).
w_R	Estrutura de wavelet reconstruída (p. 76).
R	Estrutura de wavelet residual (p. 81)
$\widetilde{f}$	Imagem residual (p. 81).
f_R	Imagem reconstruída (p. 77).

C.2 Operações e funções especiais

$A \cdot B$	Multiplicação comum (escalar) entre A e B, multiplicação ponto a ponto de A por B, aplicação de suporte de multirresolução A à estrutura de wavelet B (p. 62).
$A \times B$	Multiplicação matricial de A por B.
$A * B$	Convolução entre A e B (p. 109).
A^*	Conjugado complexo de A.
$\min(A)$	Valor mínimo da função ou estrutura A.
$sig(A)$	Função sinal. Retorna +1 se A for positivo, -1 se for negativo e 0 se for igual a 0.
$\sigma(A)$	Desvio padrão encontrado entre os elementos de A.
$A[B]$	Aplicação do operador A à função ou estrutura B.

$A^n[B]$	Aplicação repetida n vezes do operador A a B.
$A\ B\ [C]$, $A \cdot B\ [C]$	Aplicação sucessiva dos operadores B e A a C.

C.3 Filtros, operadores e transformadas

H	Filtro de média de Haar (p. 21).
$H_1, H_2, H_3, ...$	Filtros passa-baixa da técnica *à trous* (p. 24).
G	Filtro de meia diferença de Haar (p. 21).
$G_1, G_2, G_3, ...$	Filtros passa-alta da técnica *à trous* (p. 25).
FM_l	Filtro mediano com janela de largura ou diâmetro l (p. 46).
D	Operador de dilatação (p. 32).
E	Operador de erosão (p. 32).
F	Transformada de Fourier (p. 13).

F', F'', F''', ...	Formas modificadas da transformada de Fourier (p. 14).
W	Transformada *à trous* (p. 26, 39). Transformada de wavelet em geral, no Capítulo 1 (p. 7).
W^{-1}	Transformada *à trous* inversa (p. 26, 40).
A	Transformada de Anscombe (p. 46).
A'	Transformada generalizada de Anscombe (p. 47).
Pr	Operador de projeção (p. 76).
$\widetilde{\text{Pr}}$	Operador adjunto (p. 81).

Bibliografia

Bertin, E., Arnouts, S. (1996), SExtractor: Software for source extraction, *Astron. Astrophys. Suppl. Ser.*, 117, p. 393-404.

Bijaoui, A., Rué, F. (1995). A multiscale vision model adapted to the astronomical images, *Signal Processing*, 46, p. 345-362.

Bijaoui, A., Rué, F. (1996). Image processing from significant wavelet coefficients, *Sydney International Congress on Image Processing*, p. 8-12.

Brown, T. J. (2000). Combined Evidence Thresholding: A new wavelet regression technique for detail preserving image de-noising, *Proceedings IMVIP*, p. 83 - 92.

Da Rocha, C. (2002). *Estudo das propriedades das galáxias em grupos compactos*, tese de doutorado, IAG/USP.

Daubechies, I. (1994). Wavelets and other phase space localization methods, *Proceedings of the International Congress of Mathematicians*, Zürich, Switzerland.

Daubechies, I. (1996). Where do wavelets come from? A personal point of view, *Proceedings of the IEEE Special Issue on Wavelets* 84 (nº 4), p. 510-513.

Gabor, D. (1946). Theory of communication, *Journal of the IEE*, 93, p. 429-457.

Hickson, P. (1982). Systematic properties of compact groups of galaxies, *ApJ*, 255, p. 382-391.

Hickson, P., Mendes de Oliveira, C., Huchra, J. P., Palumbo, G. G. C. (1992). Dynamical properties of compact groups of galaxies, *ApJ*, 399, p. 353-367.

Hsu, H. P. (1970). *Análise de Fourier*, Editora Livros Técnicos e Científicos, Rio de Janeiro.

Lepley, M. A., Forkert, R. D. (1997). AWIC: Adaptive Wavelet Image Compression, *MITRE Technical Report* nº R 97B0000040.

Mendes de Oliveira, C., Da Rocha, C., Rabaça, C. R., Epitácio Pereira, D. N., Bolte, M. (2002). Optical Diffuse Light in Nearby Compact Groups, *Proceedings of the ESO Workshops: Extragalatic Globular ClustersSystems*.

Meyer, P. L. (1976). *Probabilidade e Aplicações à Estatística* (6ª edição), Editora Livros Técnicos e Científicos, Rio de Janeiro.

Niblac, W. (1986). *An Introduction do Digital Image Processing*, Prentice/Hall International.

Rabaça, C. R., Cuisinier, F., Lorenz-Martins, S., Epitácio Pereira, D. N., Gonçalves, D., Lastennet, E. G. F. (2001). New Insight on Hubble 4, *IAU Symposium*, v. 209, p. 1.

Spiegel, M. R. (1977). *Análise de Fourier – Coleção Schaum*, Editora McGraw-Hill do Brasil.

Starck, J. - L., Murtagh, F., Bijaoui, A. (1998). *Image processing and data analysis: the multiscale approach*, Cambridge University Press.

Starck, J. - L., Bijaoui, A., Valtchanov, I., Murtagh, F. (2000). *Astron. Astrophys. Suppl. Ser.*, 147, p. 139.

Strang, G. (1989). Wavelets and dilation equations: a brief introduction, *Siam Review*, p. 613-627.

Índice Remissivo