MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

RELATÓRIO FINAL DE PROJETO DE INICIAÇÃO CIENTÍFICA

TRANSFORMADAS INTEGRAIS E SUAS APLICAÇÕES EM TECNOLOGIAS ESPACIAIS

(PIBIC/INPE - CNPq/MCTI, PROC. 167480/2014-6)

Evandro Bernardes - Bolsista
Universidade Estadual de Maringá - UEM
E-mail: evbernardes@gmail.com

Dra. Margarete Oliveira Domingues - Orientadora
Laboratório Associado de Computação e Matemática Aplicadar
LAC/CTE/INPE - MCTI
Instituto Nacional de Pesquisas Espaciais
E-mail: margarete.domingues@inpe.br

Maringá, Abril de 2015

Sumário

1 Série e Transformada de *Fourier* **4**
1.1 Série de *Fourier* . . . 4
1.1.1 Definição . . . 4
1.2 Transformada de *Fourier* . . . 5
1.2.1 Transformada Discreta de Fourier . . . 5
1.2.2 Exemplo . . . 6
1.3 Princípio da Incerteza de Heisenberg . . . 9
1.3.1 Exemplo . . . 9
1.4 Transformada Janelada de Fourier . . . 12
1.4.1 Janelas . . . 13
1.4.2 Exemplo . . . 16

2 Transformadas *Wavelet* contínuas **18**
2.1 Funções *wavelet* . . . 18
2.1.1 Exemplo: *Wavelet* complexa de Morlet . . . 18
2.2 Transformada *wavelet* . . . 19
2.2.1 Dilatações e translações das funções *Wavelet* . . . 21
2.2.2 Transformada *wavelet* discreta . . . 21
2.2.3 Escolha das *wavelets* . . . 22
2.3 Comparações de análises de tempo-frequência . . . 23
2.3.1 Sinal com mudança brusca de frequência . . . 23
2.3.2 Chirp Linear . . . 25
2.3.3 Chirps Hiperbólicos . . . 27
2.3.4 Função de Cantor (Devil's Staircase) . . . 29
2.4 Escalograma Global . . . 33
2.5 Transformada *wavelet* com normalização diferente . . . 36
2.5.1 Exemplo . . . 36

3 Palhetas de Cores **40**
3.1 Comparação de algumas palhetas . . . 40
3.2 Teoria das Cores . . . 45
3.2.1 Definições . . . 45
3.3 Teoria de Itten . . . 47
3.3.1 Tipos de contrastes . . . 48
3.3.2 Algebrização das cores . . . 48
3.4 Escolha das palhetas . . . 49
3.4.1 Exemplo . . . 49
3.5 Deficiência visual de cor (daltonismo) . . . 54
3.5.1 Tipos de daltonismo . . . 54

4 Sinais Analíticos **61**
4.1 Exemplo: Transformada da função real . . . 61
4.2 Exemplo: transformada de um sinal analítico . . . 62
4.3 Exemplo: extração de um sinal modulado . . . 64

5 Transformada de *Hilbert* **67**
5.1 Derivação . . . 67
5.1.1 Exemplo: Modulação em Amplitude (AM) . . . 70
5.2 Análise de tempo-frequência com a frequência instantânea . . . 72
5.2.1 Exemplos . . . 72
5.3 Análise com filtros e frequência instantânea . . . 73
5.3.1 Exemplo de filtro passa-baixas . . . 73
5.3.2 Exemplo de espectro de Hilbert com filtros passa-banda . . . 77
5.4 Comparação com a transformada *wavelet* . . . 84
5.4.1 Chirp Linear crescente . . . 87
5.4.2 Soma de chirps hiperbólicos . . . 88
5.4.3 Função de Cantor . . . 89

6 Transformada de *Hilbert-Huang* **91**
6.1 *Intrinsic Mode Functions* (IMF) . . . 91
6.2 *Empirical Mode Decomposition* (EMD) . . . 91
6.2.1 Método . . . 93
6.3 *Hilbert Spectrum Analysis* (HSA) . . . 95

7 Transformadas de *Fourier* e *Wavelet* contínuas em duas dimensões **98**
7.1 Transformada de *Fourier* para duas dimensões . . . 98
7.1.1 Filtragem no domínio de *Fourier* . . . 101
7.2 Transformadas *wavelet* contínuas bidimensionais . . . 105
7.2.1 Exemplo com *wavelet* invariante na rotação. . . . 105
7.2.2 *Wavelets* direcionais . . . 109
7.3 Testes com as imagens do experimento LASCO . . . 117

8 Análise de contornos com CWT **119**
8.1 Extração da imagem . . . 119
8.2 Segmentação . . . 119
8.3 Extração de um formato . . . 119
8.4 Extração do contorno . . . 123
8.5 CWT da função λ . . . 125
8.6 Exemplo . . . 129

9 Referências Bibliográficas **133**

A Derivação transformada contínua de *Fourier* **134**
A.1 Série de *Fourier* . . . 134
A.1.1 Ortogonalidade das funções trigonométricas . . . 134
A.2 Coeficientes da série de *Fourier* . . . 135
A.3 Transformada de Fourier . . . 137
A.3.1 Forma Exponencial . . . 137
A.3.2 Definição final da transformada de Fourier . . . 138

B Derivação da constante de admissibilidade e da CWT inversa 140
B.1 Dedução . . . 140
B.1.1 Aplicar transformada de *Fourier* . . . 140
B.1.2 Integrar na escala . . . 142
B.1.3 Aplicar a transformada inversa . . . 142

C Scripts Usados nos capítulos do Trabalho 144
C.1 Série e transformada de *Fourier* . . . 144
C.1.1 Transformada de *Fourier* de um sinal ruidoso . . . 144
C.1.2 Plotagem de algumas funções janela . . . 146
C.1.3 Demonstração das limitações da transformada de *Fourier* . . . 148
C.1.4 Plotagem do espectrograma de um sinal . . . 150
C.2 Transformada *Wavelet* . . . 152
C.2.1 Teste das funções de Morlet . . . 152
C.2.2 Implementação da CWT . . . 153
C.2.3 Função de Cantor . . . 156
C.3 Palhetas de Cores . . . 158
C.3.1 Criação da matriz de Farge . . . 158
C.3.2 Conversão da matriz de Farge em vetores RGB . . . 159
C.3.3 Criação de uma palheta de cores com os vetores RGB . . . 161
C.3.4 Plotagem de uma palheta de cores . . . 163
C.3.5 Exemplo de palheta criada com as funções deste trabalho . . . 164
C.4 Transformada de *Hilbert* . . . 165
C.4.1 Cálculo da fase instantânea . . . 165
C.4.2 Cálculo da frequência instantânea . . . 166
C.4.3 Cálculo do espectro de *Hilbert* . . . 167
C.5 Transformada de *Hilbert-Huang* . . . 169
C.6 Transformada de *Fourier* e *Wavelet* em duas dimensões . . . 169
C.6.1 Extração de frequências espaciais de uma imagem . . . 169
C.6.2 Filtragem de uma imagem no domínio de *Fourier* . . . 171
C.6.3 Análise de uma imagem com *wavelets* . . . 173
C.6.4 Representação de posição com YAWTB . . . 175
C.6.5 Representação de escala-ângulo com YAWTB . . . 176

D Instalação do YAWTB no GNU/Octave 178

Capítulo 1

Série e Transformada de *Fourier*

A análise de *Fourier* é muito mais ampla do que o trabalho feito por Joseph Fourier, porém é com o trabalho dele que tudo se deu início. Neste capítulo será explicado o processo que levou a criação da série de *Fourier* , da transformada de *Fourier* , quais suas motivações, e também suas implicações.

1.1 Série de *Fourier*

A análise de *Fourier* nasceu na verdade com o intuito de se encontrar a solução analítica para uma equação diferencial parcial (EDP) conhecida: a equação do calor. Anteriormente, a solução desta equação só era conseguida quando a fonte de calor se comportava como uma simples senoide. *Fourier* propôs o uso de uma superposição de senos e cossenos (agora conhecidas como "autofunções") como solução de qualquer equação (independente de suas condições iniciais ou fonte de calor). Não demorou muito para que seus resultados fossem aproveitados em vários outros ramos da matemática e da engenharia.

O que a série faz é dar uma representação alternativa de qualquer função periódica como uma soma infinita de senos e cossenos, que quando aplicada em uma EDP (utilizando-se a própria região do domínio da solução como o período da função), a equação se reduz a uma EDO para cada um de seus termos (ou autofunções).

1.1.1 Definição

A definição da serie de *Fourier* pode ser escrita como:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{+\infty} \left[a_n \cdot \cos\left(\frac{2\pi n}{T} \cdot x\right) + b_n \cdot \operatorname{sen}\left(\frac{2\pi n}{T} \cdot x\right) \right] \tag{1.1}$$

para toda função $f(x+T) = f(x)$. Logo, é necessário saber quais são os termos a_n e b_n.

Para se conseguir deduzir a fórmula para as constantes a_n e b_n, utiliza-se duas propriedades das funções trigonométricas: o fato de elas possuírem média nula, e o fato de elas serem ortogonais entre sí (para uma demonstração, ver apêndice A). A fórmula de seus coeficientes é dada por:

$$a_0 = \frac{2}{T}\int_0^T f(x)dx$$

$$a_n = \frac{2}{T}\int_0^T f(x)\cos\left(\frac{2\pi n}{T}\cdot x\right)dx$$

$$b_n = \frac{2}{T}\int_0^T f(x)\text{sen}\left(\frac{2\pi n}{T}\cdot x\right)dx$$

O que pode ser observado deste resultado é que com o cálculo dos coeficientes a_n e b_n da série, consegue-se uma representação alternativa da mesma função (ou sinal). Antes possuía-se a função em seu estado original com valores para cada instante no tempo, e agora tem-se a função desmembrada em valores discretos para cada frequência, já que cada coeficiente indica a energia que uma certa frequência $\frac{n}{T}$ possui no sinal. Isto é chamado de *representação no domínio da frequência* de um sinal.

Porém, por mais que a série de *Fourier* tenha muita utilidade em EDPs, para o processamento de sinais ela não é muito boa no sentido de que só existe para funções que sejam periódicas no tempo. Por isto, uma generalização deste resultado deve ser usado para que se consiga também uma representação no domínio da frequência de uma função qualquer que não seja periódica. Esta generalização será demonstrada na seção seguinte.

1.2 Transformada de *Fourier*

A transformada de *Fourier* nada mais é do que uma forma de representação de frequência de uma função, e sua diferença reside no fato de que ela pode ser aplicada em qualquer função, periódica ou não. Em geral, ela é usada para se analisar sinais aperiódicos, porém, caso seja aplicada em uma função periódica, o resultado será reduzido aos coeficientes discretos encontrados com a série de *Fourier* , o que mostra que a transformada nada mais é do que uma generalização do resultado obtido anteriormente.

No apêndice A é demonstrada a derivação da transformada de *Fourier* . A forma da transformada usada neste trabalho será a seguinte forma (utilizando-se de uma frequência dada em Hz):

$$\hat{f}(\xi) = \int f(t)\cdot e^{-2\pi\imath\xi t}dt \tag{1.2}$$

Que possui inversa definida como:

$$f(t) = \int \hat{f}(\xi)\cdot e^{2\pi\imath\xi t}d\xi \tag{1.3}$$

1.2.1 Transformada Discreta de Fourier

Para que a transformada seja útil computacionalmente, ela precisa ser adaptada para a natureza discreta do computador. Para isto, deve ser definida a *Transformada Discreta de Fourier*, ou simplesmente DFT (de *Discrete Fourier Transform*). Embora exista uma alternativa chamada DTFT (*Discrete Time Fourier Transform*, ou *Transformada de Fourier de Tempo Discreto*) onde apenas o tempo (e não a variável de frequência) é discretizada, definirei apenas a DFT, pois ela é a melhor para se trabalhar com um computador.

O sinal original neste caso será uma sequência $x_0, x_1, ..., x_{N-1}$ de N amostras. A DFT será também uma sequência $X_0, X_1, ..., X_{N-1}$ definida como:

$$X_k \stackrel{\text{def}}{=} \sum_{n=0}^{N-1} x(n) \cdot e^{-\imath \frac{2\pi n}{N} k} \tag{1.4}$$

e a fórmula da transformada inversa dada por:

$$x_n = \frac{1}{N} \sum_{n=0}^{N-1} X(k) \cdot e^{+\imath \frac{2\pi k}{N} n} \tag{1.5}$$

Deve-se notar que a variável n indica a amostra desejada, e a variável k indica frequência na unidade de "ciclos por amostra". Se for desejado utilizar unidades usuais de tempo e frequência (como segundos e Hertz), uma conversão deve ser feita quando os dados forem apresentados.

Essa transformada é excepcional para ser usada com computadores porque, além de ser totalmente discretizada, existem algoritmos extremamente eficientes que foram desenvolvidos para que o cálculo seja rápido. Estes algoritmos são as chamadas *Transformadas Rápidas de Fourier*, ou FFT (*Fast Fourier Transform*). Tais algoritmos são tão importantes que em programas matemáticos tais como o MATLAB, Scilab e o GNU/Octave, não existe nenhum comando chamado "DFT", apenas existe o comando "FFT".

Será mostrado agora um exemplo utilizando-se a transformada discreta de *Fourier* no programa GNU/Octave para ser feita uma análise das frequências existentes em um sinal.

1.2.2 Exemplo

Neste exemplo feito com o GNU/Octave, na primeira imagem observa-se um sinal ruidoso, onde dificilmente podem ser vistas frequências constituintes. Em seguida, a DFT do sinal. Note que, apesar de ruidosa, a imagem da transformada mostra picos nos valores de 10 e 30 Hertz, indicando que estas frequências estão presentes no sinal.

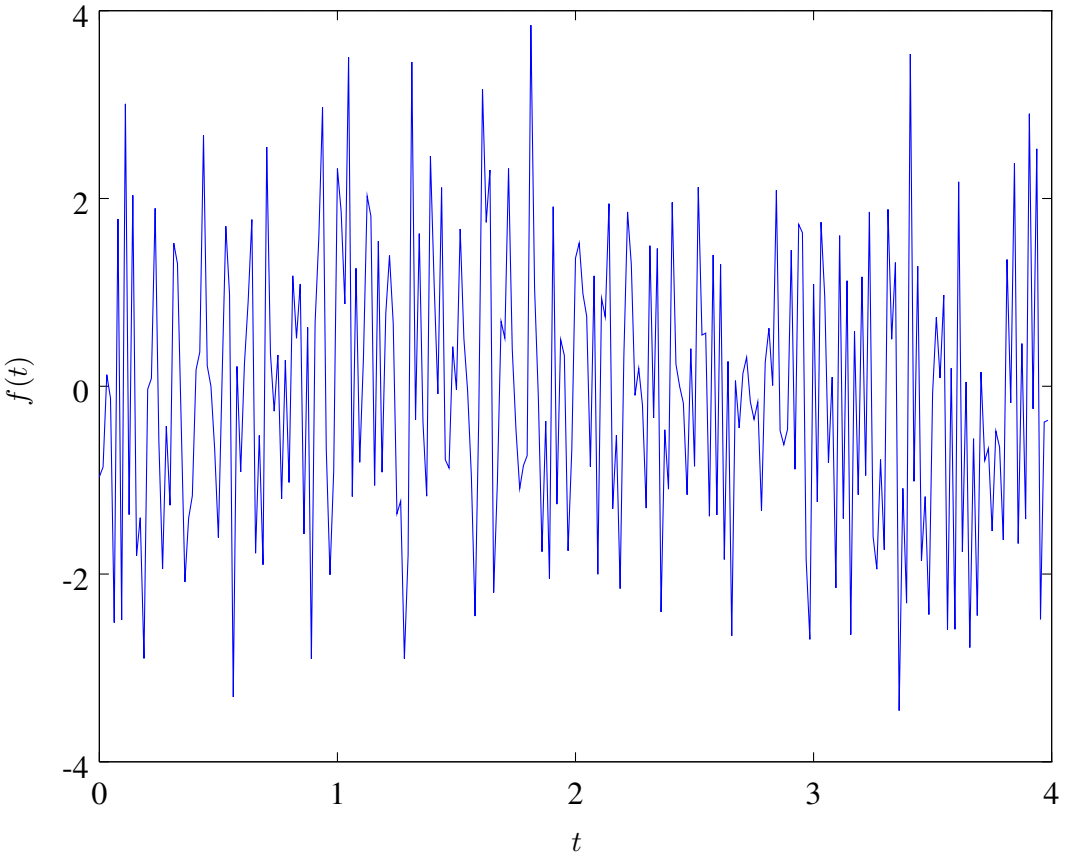


Figura 1.1: Sinal original

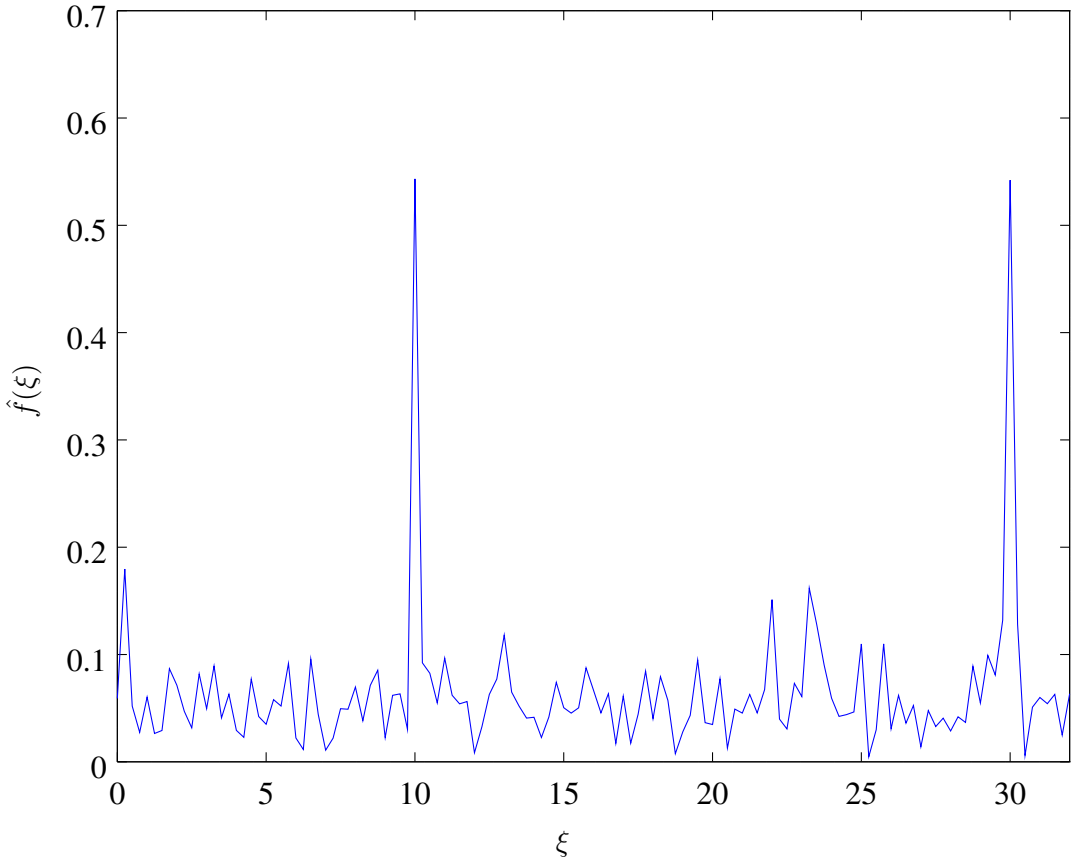


Figura 1.2: DFT do sinal

O código do GNU/Octave para gerar este exemplo está na página 144.

1.3 Princípio da Incerteza de Heisenberg

Do modo como a série de Fourier é definida, sopõe-se que o sinal possui as mesmas frequências durante todo o seu domínio, fato que acompanha também a transformada. Desta forma, não se pode tratar o sinal como tendo diferentes frequências em diferentes momentos, o que dificultaria muitas aplicações. Esta dificuldade é formalizada com o seguinte teorema:

Teorema 1.1 (Princípio da Incerteza de Heisenberg)**.** *Existe um balanço entre a resolução em tempo e a resolução em frequência. Se f é uma função do $\mathbb{L}^2(\mathbb{R})$, então as variações em tempo Δt e em frequência $\Delta\xi$ são inversamente proporcionais, tal que:*

$$(\Delta t)^2(\Delta\xi)^2 \geq constante \tag{1.6}$$

Em que tal constante depende da normalização utilizada.

Desta maneira, supondo que a ferramenta de análise escolhida é alguma transformada de Fourier, temos dois casos:

- Domínio do tempo, onde existe total resolução no tempo ($\Delta t = 0$, indicando que é possível saber a exata força do sinal em qualquer instante), porém nenhuma resolução na frequência ($\Delta\xi = +\infty$, indicando que não se sabe em nenhum instante qual a frequência do sinal).
- Domínio da frequência, onde existe total resolução na frequência ($\Delta\xi = 0$, indicando que se sabe a exata força de cada frequência no sinal), porém nenhuma resolução mínima no tempo ($\Delta t = +\infty$).

Portanto, o único meio de se analisar um sinal como uma sucessão de diferentes frequências em diferentes momentos, é sacrificar a resolução total em uma das variáveis, obtendo uma resolução finita nas suas simultaneamente, chamada de representação tempo-frequência.

Para exemplificar, serão comparados os espectros de dois sinais diferentes:

1.3.1 Exemplo

Desta vez, o sinal ruidoso usado anteriormente será comparado com outro sinal que, sem ruído, que ao invés de portar as duas frequências a todo instante, ela apresenta cada uma em um certo momento. É fácil verificar que, em uma aplicação prática, dificilmente usando a transformada de *Fourier* convencional os dois sinais poderiam ser distinguidos. O código para esta comparação está na página C.1.3

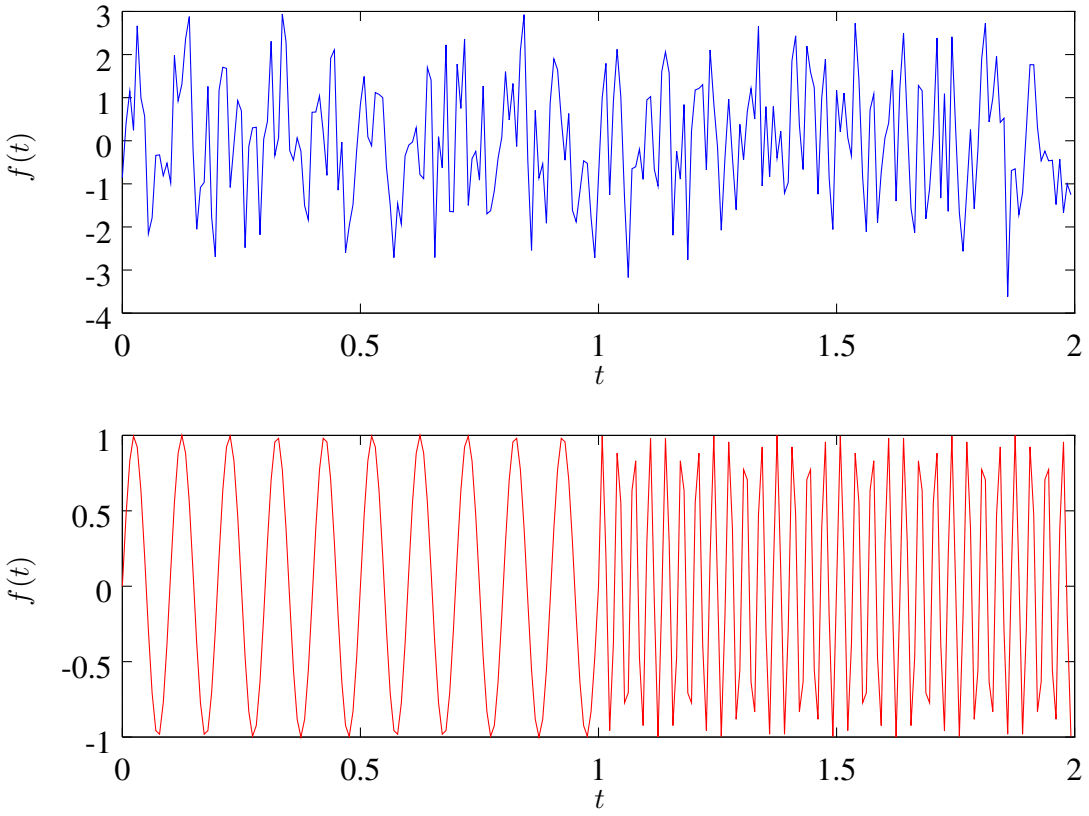


Figura 1.3: Sinais a serem comparados

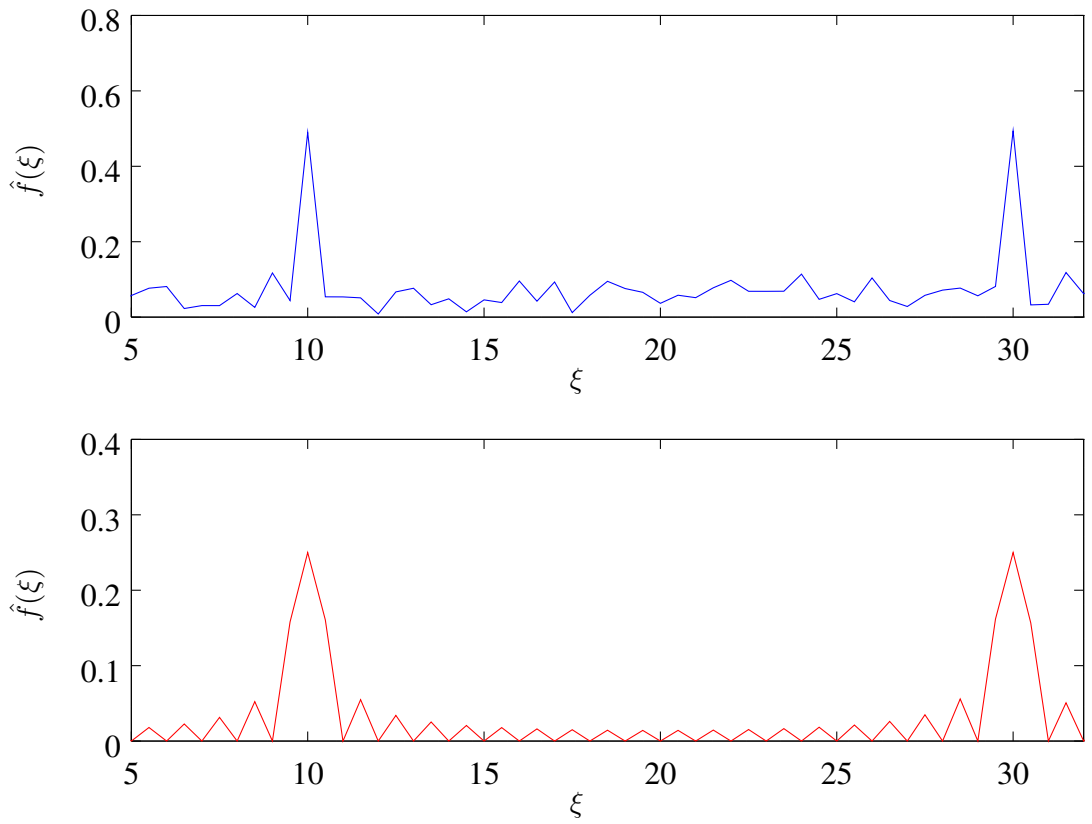


Figura 1.4: FFT dos sinais

Na teoria, é possível de se calcular a informação temporal da transformada de *Fourier* calculando as fases dos coeficientes de *Fourier* , que é o que se faz quando a fórmula da transformada inversa é aplicada. Na prática, computá-los com precisão é impossível, e o fato de não se ter informação temporal é um sério problema, que faz com que uma característica *local* de um sinal acabe se tornando uma característica *global* [9]. Como dito por Meyer, "se, ao codificar um sinal de uma hora, um erro ocorrer nos últimos 5 minutos, este erro irá corromper toda a transformada de *Fourier* ".

1.4 Transformada Janelada de Fourier

Um modo de se conseguir este meio termo é utilizando a chamada *Transformada Janelada de Fourier*, ou WFT (*Windowed Fourier Transform*). Ela é calculada adequando a transformada de Fourier, primeiro multiplicando a função por uma "janela", e em seguida calculando a transformada. Esta janela é, idealmente, uma função de suporte compacto (nula fora de um certo intervalo).

A transformada neste caso é definida como:

$$\hat{f}(\tau,\xi) = \int f(t)w(t-\tau)e^{-2\pi\imath\xi t}dt \tag{1.7}$$

Em que a função $g(t)$ é a janela e o parâmetro τ é chamado de "deslocamento". Se esta janela for nula (ou suficientemente próxima de zero) fora de um certo intervalo $L = \Delta t$, a transformada irá, para cada valor de τ, analisar quais as frequências existentes em uma região de tamanho L próxima de $t = \tau$ (com uma resolução $\Delta\xi$ inversamente proporcional a τ).

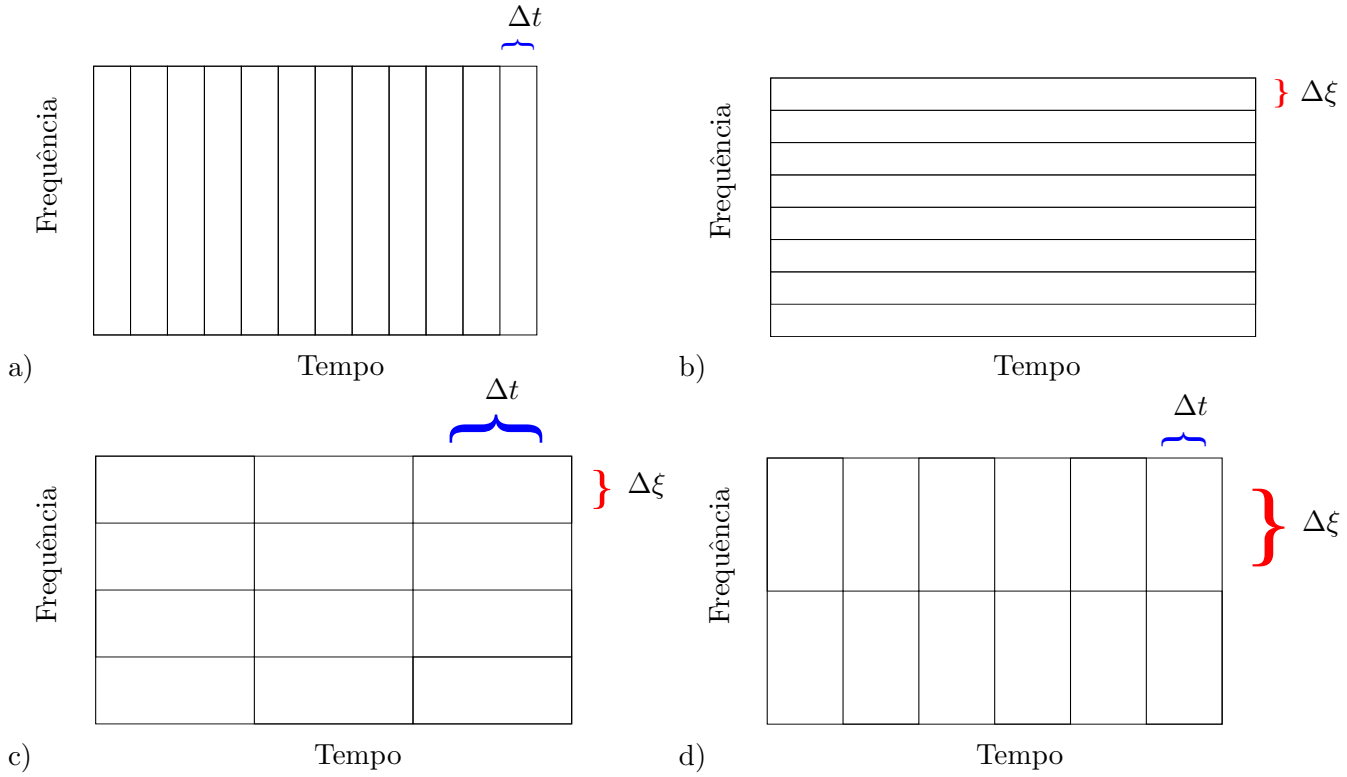


Figura 1.5: Comparação das resoluções no domínio do tempo (figura a), no domínio de *Fourier* (figura b), e de duas transformadas janeladas com diferentes tamanhos de janela (figuras c e d).

Deste que a janela seja normalizada de modo que $\int w(t)dt = 1$, a inversão pode ser feita pela fórmula:

$$f(t) = \int\int \hat{f}(\tau,\xi)e^{+2\pi\imath\xi t}d\tau d\xi \tag{1.8}$$

A WFT também possui uma forma discreta, e assim como a transformada de *Fourier* original, pode ser implementada usando FFTs para melhorar sua performance computacional. Ela será usada a seguir para demonstrar o espectro de um sinal.

1.4.1 Janelas

Aqui serão mostrados alguns exemplos de funções janela que podem ser usados, além de seus espectros e o logaritmo de seus espectros. O script escrito para definir e plotar estes gráficos está na página 146

Janela retangular

A janela mais simples de se imaginar é a janela retangular, presenta na Figura 1.6. Ela possui a propriedade de seu suporte ser totalmente compacto, porém outras janelas são formadas para tentar se amenizar o efeito de descontinuidade que existe nela, o que gera efeitos indesejáveis em suas transformadas.

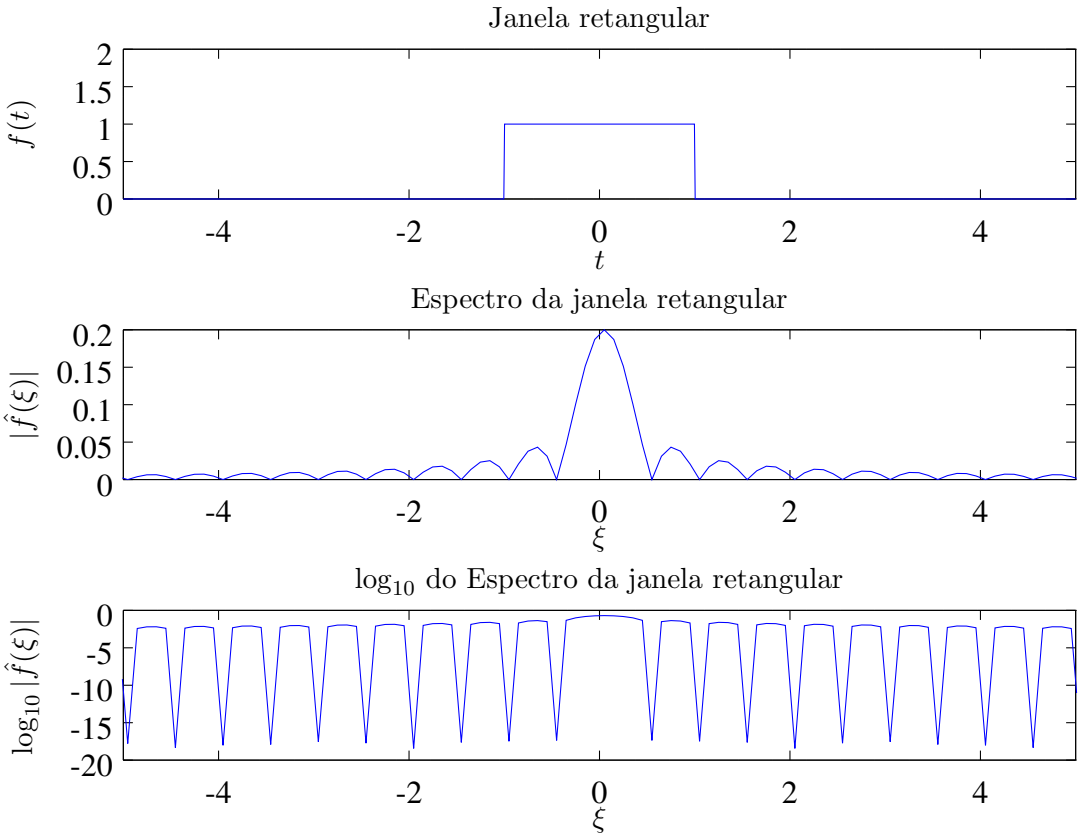


Figura 1.6: Janela retangular e sua transformada de *Fourier* .

Janela de Hanning

Na Figura 1.7, vemos a janela de Hanning. Ela foi implementada com a fórmula:

$$f(t) = -0.5\left(1 - \cos\left(\frac{2\pi\ t}{10}\right)\right) \tag{1.9}$$

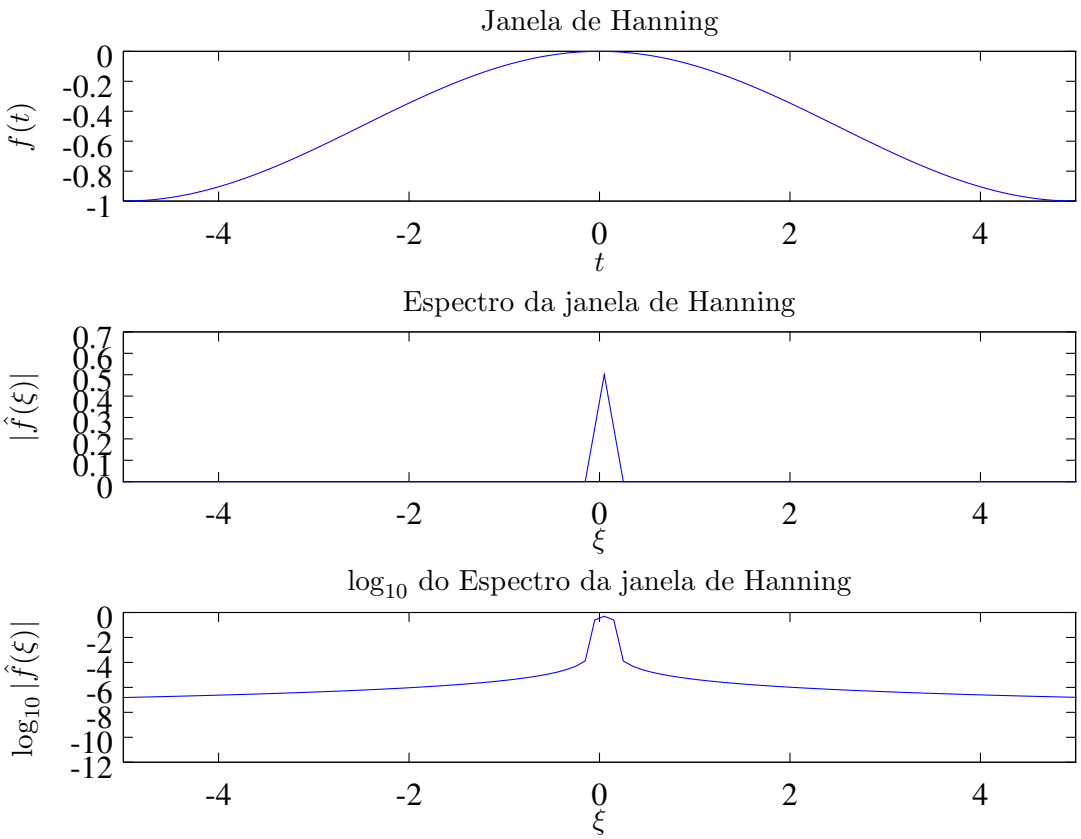


Figura 1.7: Janela de Hanning e sua transformada de *Fourier* .

Janela gaussiana

Aqui, na Figura 1.8, vemos o uso de uma função Gaussiana como janela. Ela foi implementada com a fórmula:

$$f(t) = e^{-\frac{1}{2}\left(\frac{t}{5\sigma}\right)^2} \tag{1.10}$$

Possui a vantagem de seu espectro de *Fourier* também ser uma função Gaussiana, porém seu suporte não é realmente compacto.

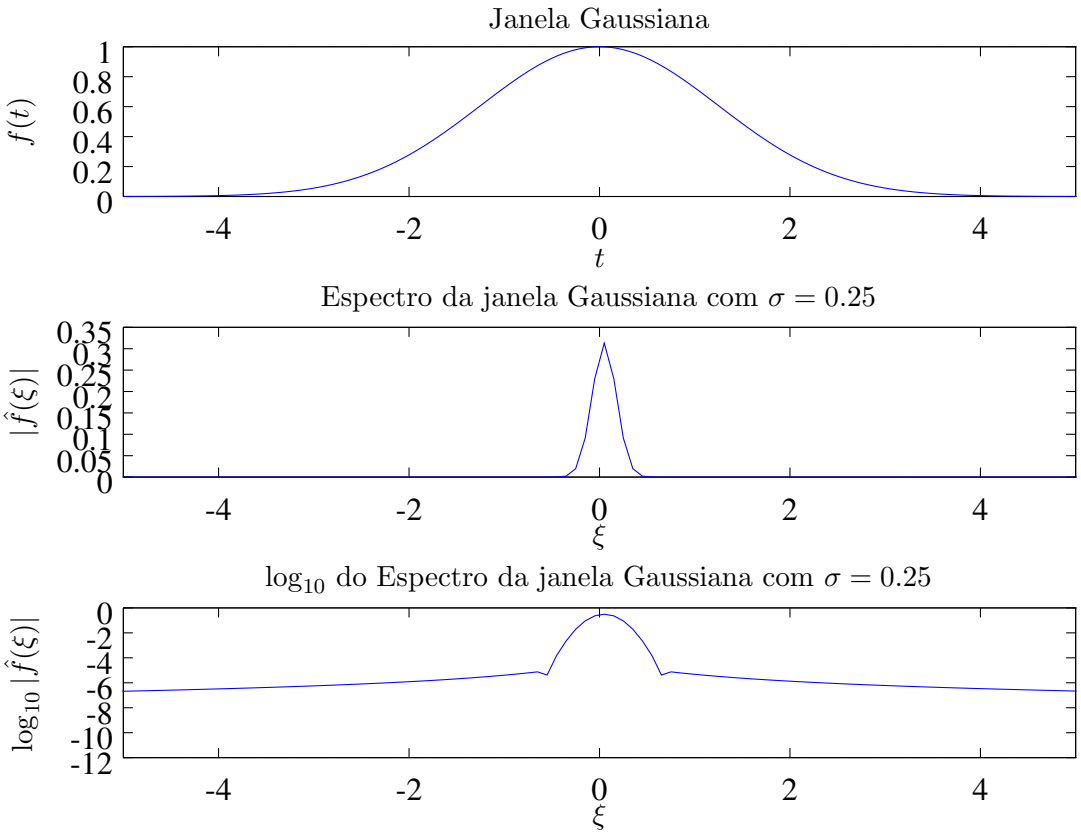


Figura 1.8: Janela Gaussiana com $\sigma = 0.25$ e sua transformada de *Fourier* .

1.4.2 Exemplo

Usarei novamente o software GNU/Octave para demonstrar o espectrograma de um sinal, utilizando uma janela retangular. Este sinal é composto por várias frequências em diferentes momentos, além de algum ruído, e usarei o espectrograma para tentar verificar quais são as frequências que o compõem.

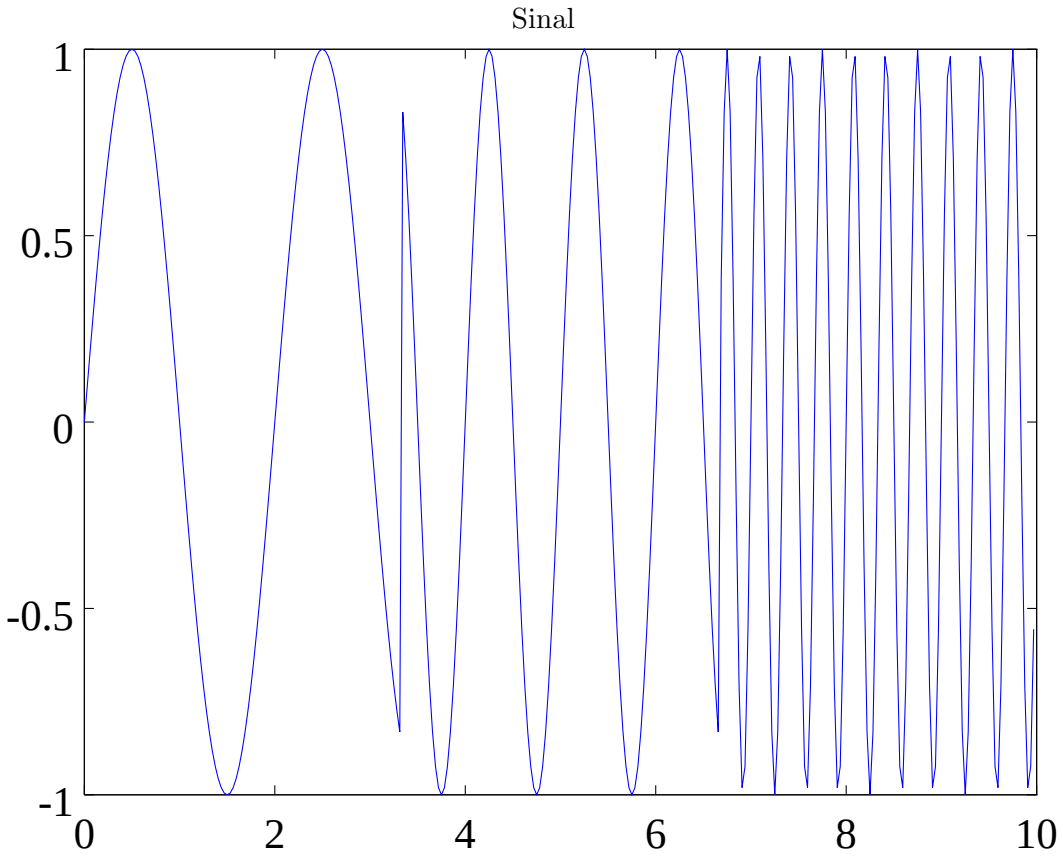


Figura 1.9: Sinal a ser analisado.

Pode ser observado destas imagens na Figura 1.10 que do espectrograma com a menor janela, mesmo sendo de péssima resolução na frequência (eixo vertical), é fácil observar o momento em que a sinal tem uma mudança drástica de comportamento (em torno de 3 e 7 segundos), o que indica sua boa resolução no tempo. Por outro lado, enquanto o sinal com janela maior indicou melhor quais as frequências presentes, ele não conseguiu demonstrar em que momento do sinal elas estão presentes. Estas observações estão de acordos com o teorema da incerteza.

O código que gerou os espectrogramas no GNU/Octave é dado na página 150.

Capítulo 2: Transformadas Wavelet contínuas

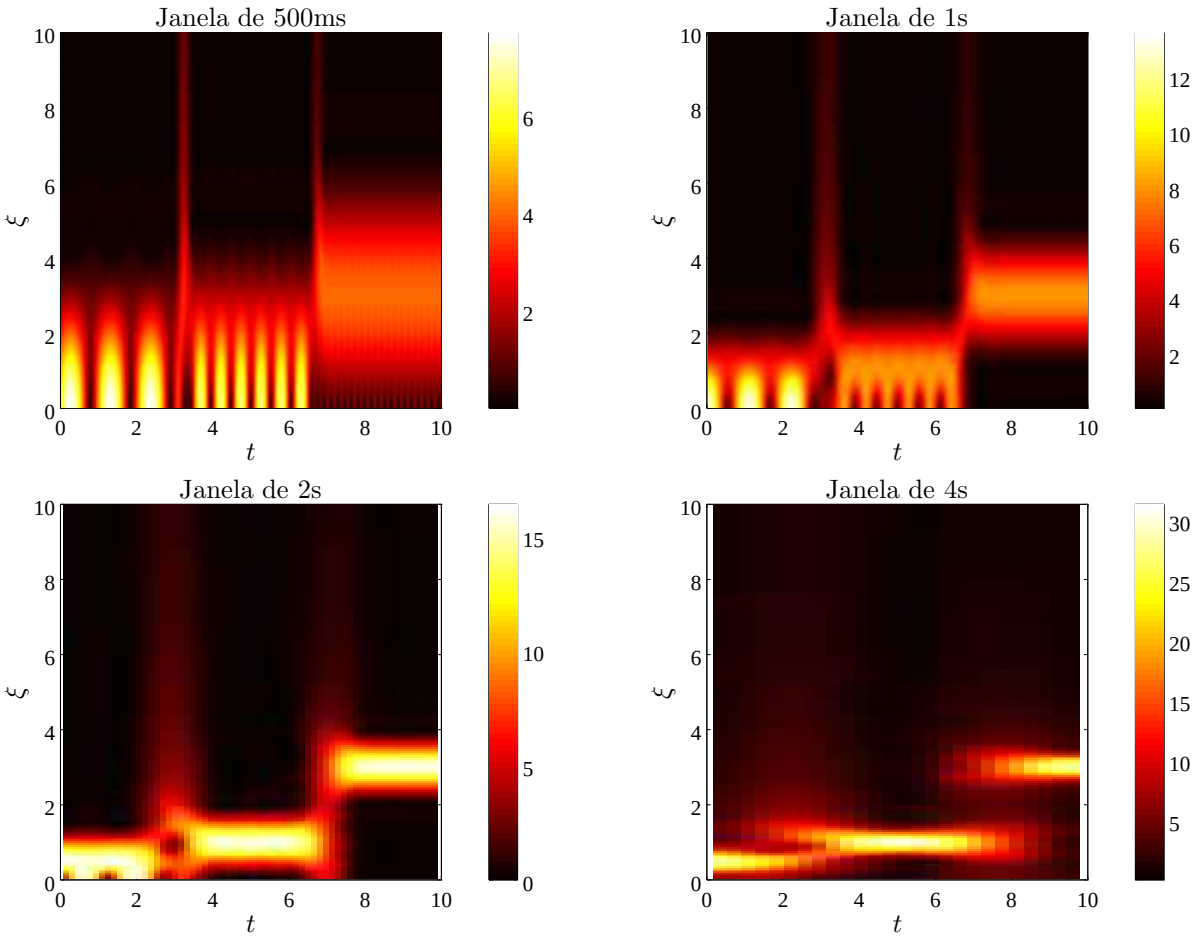


Figura 1.10: Transformadas Janeladas para janelas de diferentes tamanhos

Capítulo 2

Transformadas *Wavelet* contínuas

A transformada Janelada de *Fourier* , embora a primeira vista pareça ser a solução definitiva para o problema da representação tempo-frequência, possui um grande problema: sua janela possui tamanho fixo. Aos fazer espectrogramas com certos sinais, percebe-se que em cada caso, há alguns tamanhos de janela que funcionam melhor que outros, dependendo da gama de frequências presentes no sinal.

Enquanto, por exemplo, uma janela w_1 possa funcionar muito bem com um sinal que possua frequências de 100 a 500Hz, e uma outra janela w_2 possa funcionar perfeitamente bem para um sinal com frequências de 1000 a 5000Hz, nenhuma delas funcionará para um sinal que possua todas estas frequências.

Sinais que mudam bruscamente de frequência precisam de uma janela que se adapte, e é para estes casos que a transformada *Wavelet* é necessária.

2.1 Funções *wavelet*

As "*wavelets* ", que vem do francês "*ondelettas*" são literalmente "pequenas" ondas, oscilações que acontecem por um breve intervalo e então cessam. Para que uma função seja uma função *wavelet* (neste trabalho denotava pela letra ψ), ela deve seguir algumas propriedades:

1) A condição de admissibilidade

$$c_\psi = \int \frac{|\hat{\psi}(\xi)|^2}{|\xi|} d\xi < \infty, \tag{2.1}$$

em que $\hat{\psi}$ é a transformada de *Fourier* da função ψ, e normalmente é o suficiente apenas garantir que

$$\int \psi(t) dt = 0, \tag{2.2}$$

2) A condição de energia unitária

$$\int |\psi(t)|^2 dt = 1, \tag{2.3}$$

Que garante que a função *wavelet* possua de fato suporte compacto (ou, ao menos um suporte efetivo).

Serão testadas agora as características das funções de Morlet.

2.1.1 Exemplo: *Wavelet* complexa de Morlet

Definindo a função complexa de Morlet como $\psi_\beta(t) = \pi^{-\frac{1}{4}} e^{-\frac{1}{2}t^2} (e^{\imath\beta t} - e^{-\frac{1}{2}\beta^2})$, as condições de energia unitária e admissibilidade devem ser testadas para saber se pode-se chamá-las de funções *wavelet* ou não.

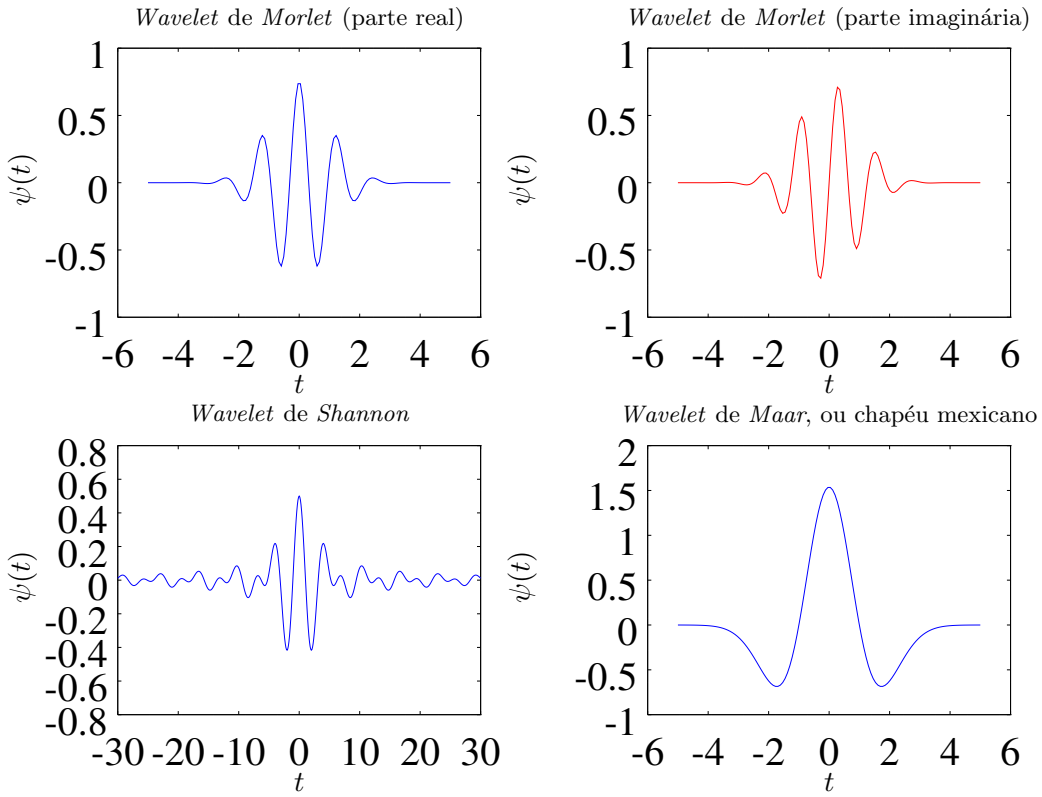


Figura 2.1: Algumas *wavelets*

Foi usado o GNU/Octave para testar esta função para alguns valores de β (na página 152). Estes são os gráficos de algumas das funções testadas, sendo a parte real em azul e a parte imaginária em vermelho:

Conclui-se que para valores de β maiores ou iguais a 5 a função pode ser classificada como uma *wavelet* . Observa-se também pelas figuras que o valor desta constante está relacionada com a quantidade de oscilações da *wavelet* .

2.2 Transformada *wavelet*

A transformada *wavelet* contínua (CWT, ou *Continuous Wavelet Transform*) de uma função f é definida como:

$$\mathfrak{W}_f^{\psi}(a,b) = \int f(t)\bar{\psi}_{a,b}(t)dt \quad a > 0, \tag{2.4}$$

em que

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right)$$

indica várias dilatações, contrações e translações da *wavelet* original, chamada de *wavelet* -mãe (*mother wavelet*), e $\bar{\psi}$ indica o conjugado complexo de ψ.

A CWT, assim como a transformada janelada, depende de dois parâmetros. O parâmetro b aqui é análogo ao parâmetro τ de translação da WFT, enquanto o parâmetro a é chamado de "escala" e substitui o conceito de frequência. A escala é, de certo modo, análogo a um inverso da frequência, pois uma menor

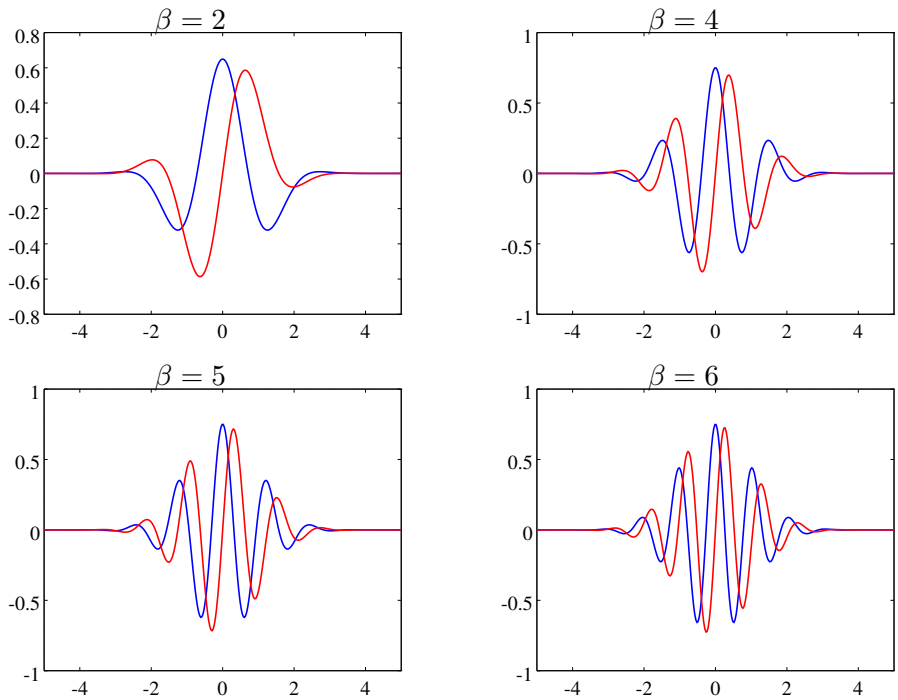


Figura 2.2: Funções de Morlet para diferentes valores de β.

β	c_ψ	Energia
1	$07,621477$	0.423146
2	$28,651145$	0.918742
3	$35,812882$	0.997782
4	$36,406837$	0.999988
5	$36,423521$	1.000000
6	$36,423694$	1.000000
10	$36,423694$	1.000000
20	$36,423694$	1.000000

Tabela 2.1: Tabela de testes da função de Morlet

escala indica uma maior frequência e vice-versa, e do mesmo modo que o gráfico do valor absoluto da transformada janelada é chamado de espectrograma, o gráfico do valor absoluto da transformada *wavelet* é chamado de escalograma.

Assim como as outras transformadas, a CWT também possui uma fórmula para a sua inversa:

$$f(t) = \frac{1}{c_\psi} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{a^2} \mathfrak{W}_f^\psi(a,b) \psi_{a,b}(t) da db \tag{2.5}$$

Onde a constante c_ψ é a constante de admissibilidade definida anteriormente. Uma prova da fórmula da transformada inversa está no apêndice B na página 140.

A transformada pode ser interpretada como a convolução do sinal com várias *wavelets* de diferentes tamanhos (escalas) e em diferentes momentos (translações), obtendo a similaridade da função com tal *wavelet* . Esta característica da CWT de não possuir uma janela fixa faz com que, em intervalos do sinal onde a frequência é baixa (menos variações), as janelas de tempo são maiores (escalas maiores), e o sinal é avaliado menos vezes (diminuindo a redundância). Já quando o ocorre o oposto, ou seja, as variações são muito altas (frequências altas), as janelas de tempo são menores, pegando melhor as informações do

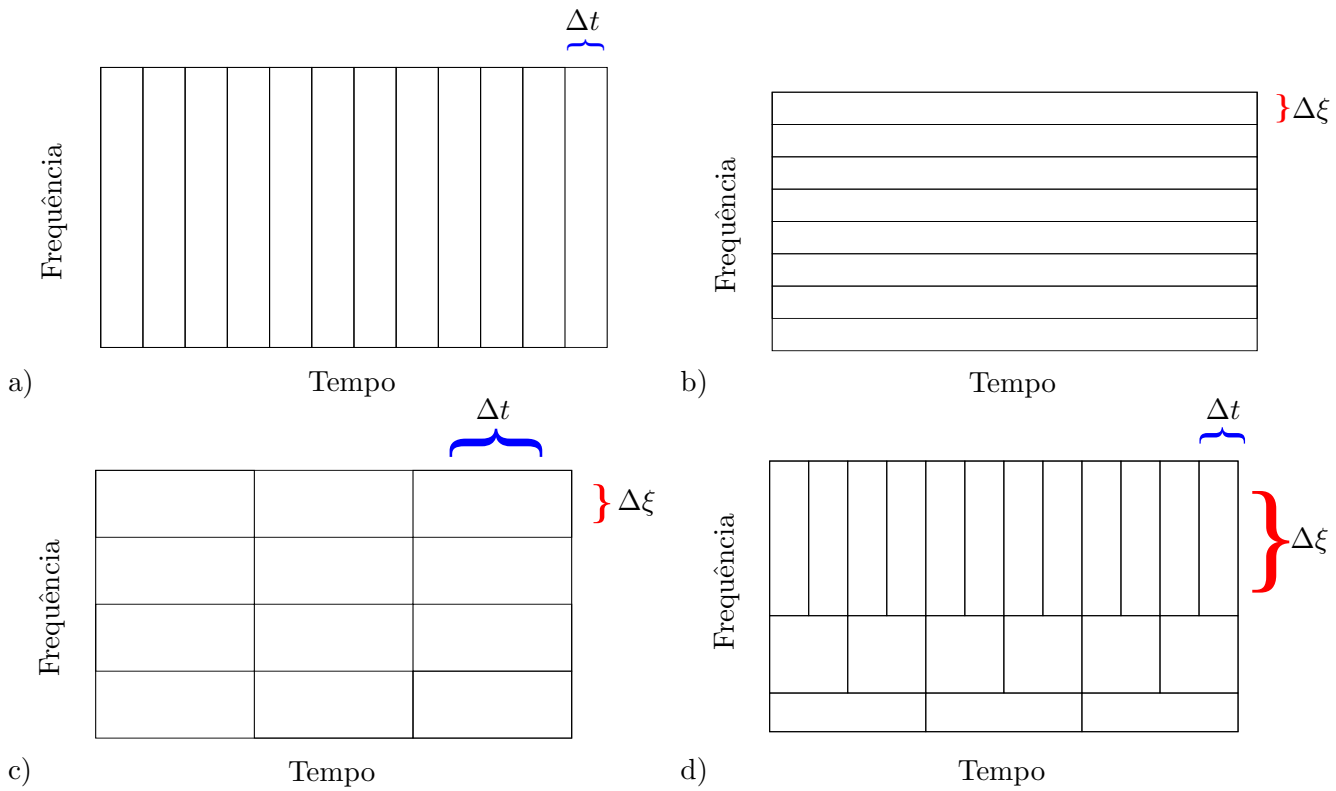


Figura 2.3: Resoluções no tempo (Figura a), na frequência (Figura b), da transformada janelada (Figura c) e da transformada *wavelet* (na Figura d).

sinal. Isto é o que torna as *wavelets* muito mais eficientes em analisar sinais não estacionários.

2.2.1 Dilatações e translações das funções *Wavelet*

Para demonstrar o efeito da mudança das variáveis a e b da transformada, será usada novamente a *wavelet* de Morlet. É possível notar que o número de oscilações não muda, elas apenas acontecem mais ou menos rapidamente e em momentos diferentes.

Nas três figuras de cima, o parâmetro b é mudado de -5 até 5, e nas figuras de baixo, o parâmetro a é variado de 0.5 até 3, demonstrando que estes parâmetros realmente atuam no sentido de transladar ou dilatar/contrair a função. Além disso, como a escala está relacionada a frequência da *wavelet* analisadora, é possível que se defina uma frequência associada a escala, a "pseudo-frequência", chamada aqui de ξ_a. Ela é definida como:

$$\xi_a = \frac{\xi_\psi}{a\Delta t} \tag{2.6}$$

Onde Δt é o período de amostragem e ξ_ψ é a frequência central associada a *wavelet* . É possível ver o significado dessa frequência central ao analisar o espectro de *Fourier* de uma *wavelet* para vários valores de a na Figura 2.5.

É possível ver que o espectro tende a se centralizar em uma certa frequência para cada valor de escala.

2.2.2 Transformada *wavelet* discreta

Assim como as outras transformadas, é definida [8] uma forma discreta da transformada *wavelet* , dada por:

$$d_k^j = 2^{-\frac{j}{2}} \int f(t)\psi_k^j(t)dt,$$

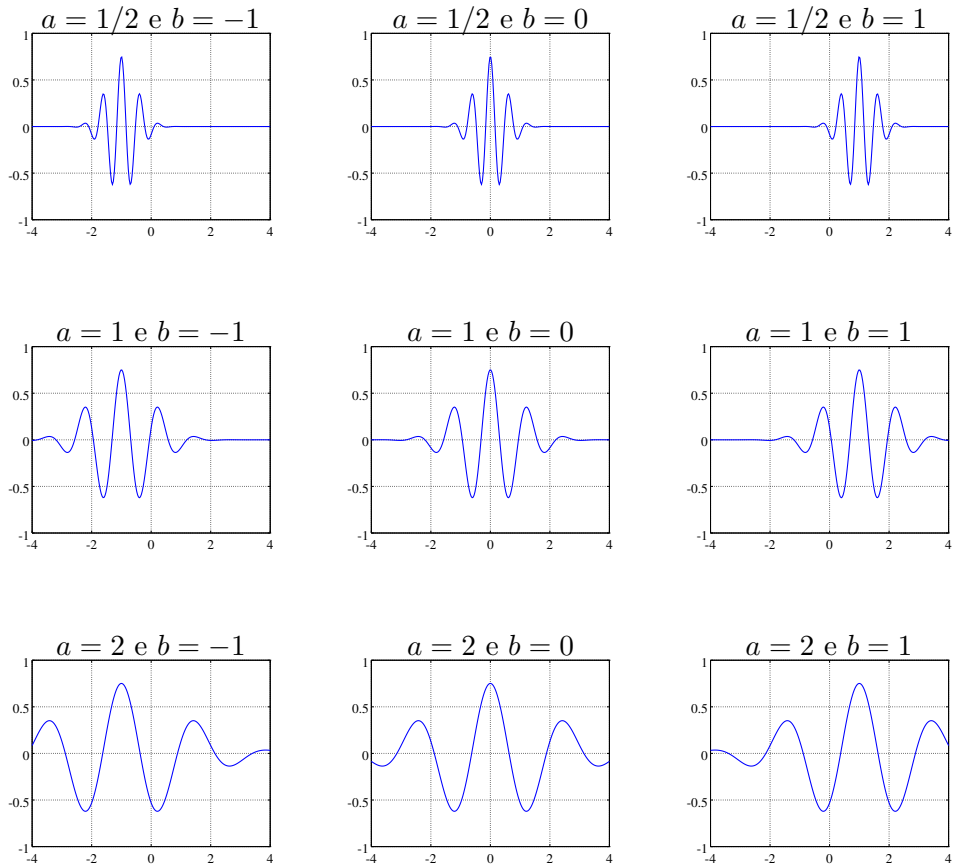


Figura 2.4: Translações de dilatações de uma *wavelet* de Morlet

em que

$$\psi_k^j(t) = 2^{-\frac{j}{2}}\psi(2^{-j}t - k)$$

são as funções *wavelet* e d_k^j os coeficientes. Estes conjuntos de funções *wavelets* são ortogonais, e as variáveis j e k representam, respectivamente, as variáveis de escala e translação.

2.2.3 Escolha das *wavelets*

O número de possíveis transformadas *wavelet* é tão grande quanto o número de funções *wavelet* existentes, e cada uma gera um escalograma diferente com, com diferentes características e usos. Por exemplo, caso se deseja estudar mudanças de amplitude e fase, *wavelets* complexas (como a de Morlet, mostrada nos exemplos) podem ser mais adequadas, pois capturam melhor o comportamento oscilatório dos dados [5].

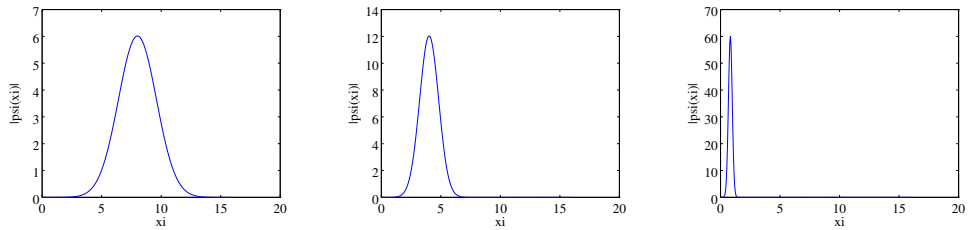


Figura 2.5: Espectro de *Fourier* da *wavelet* de Morlet para $a = \frac{1}{10}$, $a = \frac{1}{5}$ e $a = 1$

2.3 Comparações de análises de tempo-frequência

Nesta seção, serão feitas algumas comparações sobre a visualização de análises de tempo-frequência.

2.3.1 Sinal com mudança brusca de frequência

Será mostrado agora para exemplificar, escalogramas de um sinal similar ao analisado com um espectrograma nos capítulos anteriores.

Pode se perceber que na *wavelet* de Morlet, perde-se um pouco a resolução no tempo se olhar o gráfico da amplitude. Entretanto, pode-se analisar também o gráfico de suas fases, onde é possível recuperar tais informações.

Fica bem claro (principalmente na CWT feita com a *wavelet* de Morlet) como as resoluções mudam de acordo com a escala. Além disso, pode-se observar como ao mesmo tempo que as duas transformadas possuem as mesmas características globais seus detalhes são diferentes. Por esses detalhes é que cada *wavelet* possui suas aplicações específicas, pois cada uma demonstra características do sinal que são necessários em cada aplicação. Calculou-se a CWT acima utilizando a *wavelet* do chapéu mexicano, definida como:

$$\psi_\sigma(t) = \frac{2}{\sqrt{3\sigma}\pi^{1/4}}\left(1 - \frac{t^2}{\sigma^2}\right)e^{-\frac{t^2}{2\sigma^2}} \tag{2.7}$$

Onde σ é uma constante que depende da normalização, sendo igual a $\frac{1}{8}$ neste exemplo (enquanto o β na usado na *wavelet* de Morlet é 5). O código que o gerou é uma implementação simples da CWT feita no GNU/Octave, e também está disponível no apêndice, na página 153.

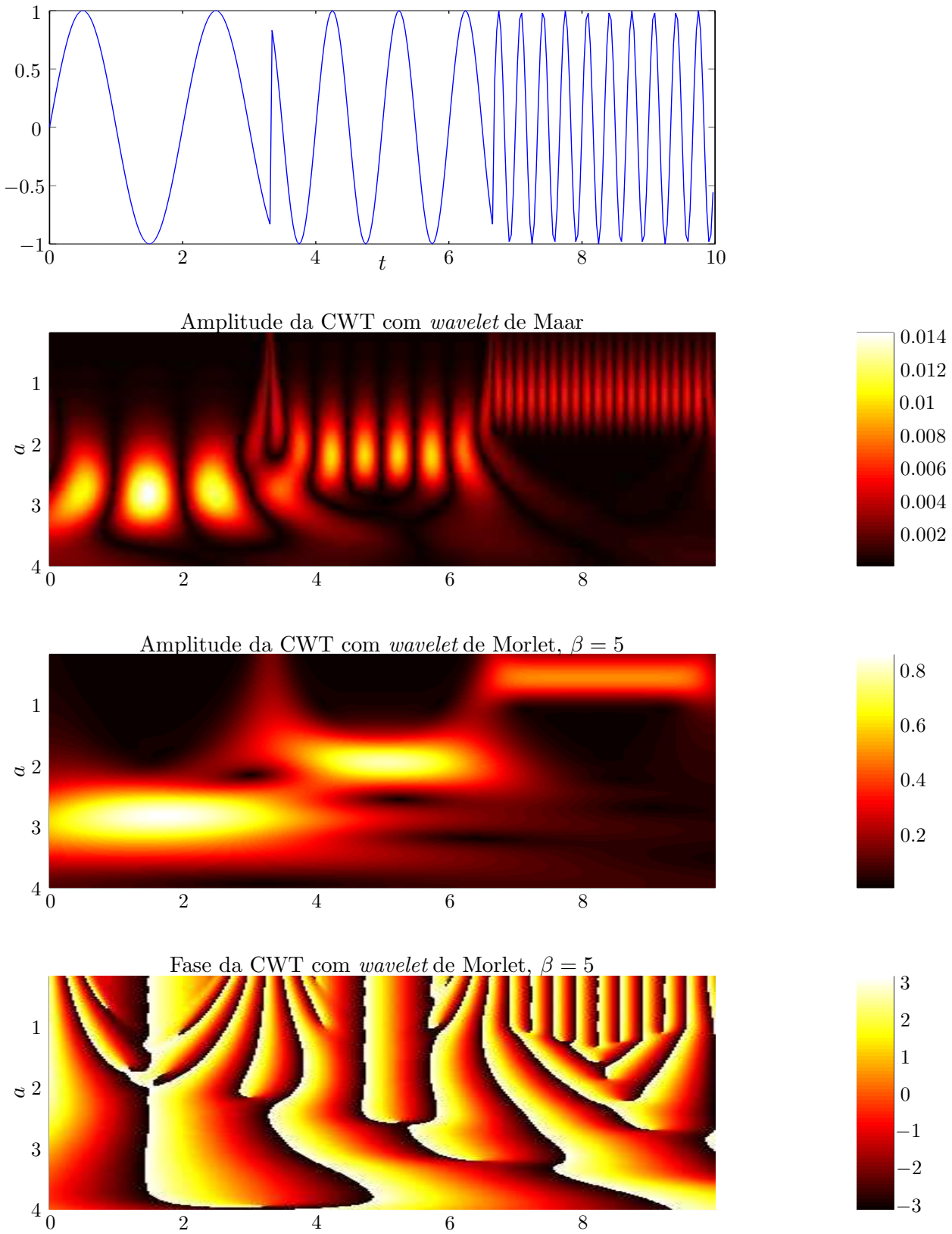


Figura 2.6: Sinal analisado: Na primeira figura acima, o sinal no domínio do tempo. Logo abaixo, módulo da CWT utilizando a *wavelet* do chapéu mexicano, e nas últimas duas, o módulo e a fase usando a *wavelet* de Morlet com $\beta = 5$

2.3.2 Chirp Linear

Serão feitas algumas análises em relação uma função chirp linear. Primeiramente, serão mostrados os espectrogramas usando duas janelas diferentes, a janela de Hanning e a janela retangular. Em seguida, serão mostrados os escalogramas do mesmo sinal utilizando-se das *wavelets* do chapéu mexicano e de Morlet. Todos os escalogramas usados a partir deste exemplo foram feitos com a toolbox YAWTb (*Yet Another Wavelet Toolbox*). Detalhes de sua instalação no GNU/Octave serão dados no apêndice, na página D.

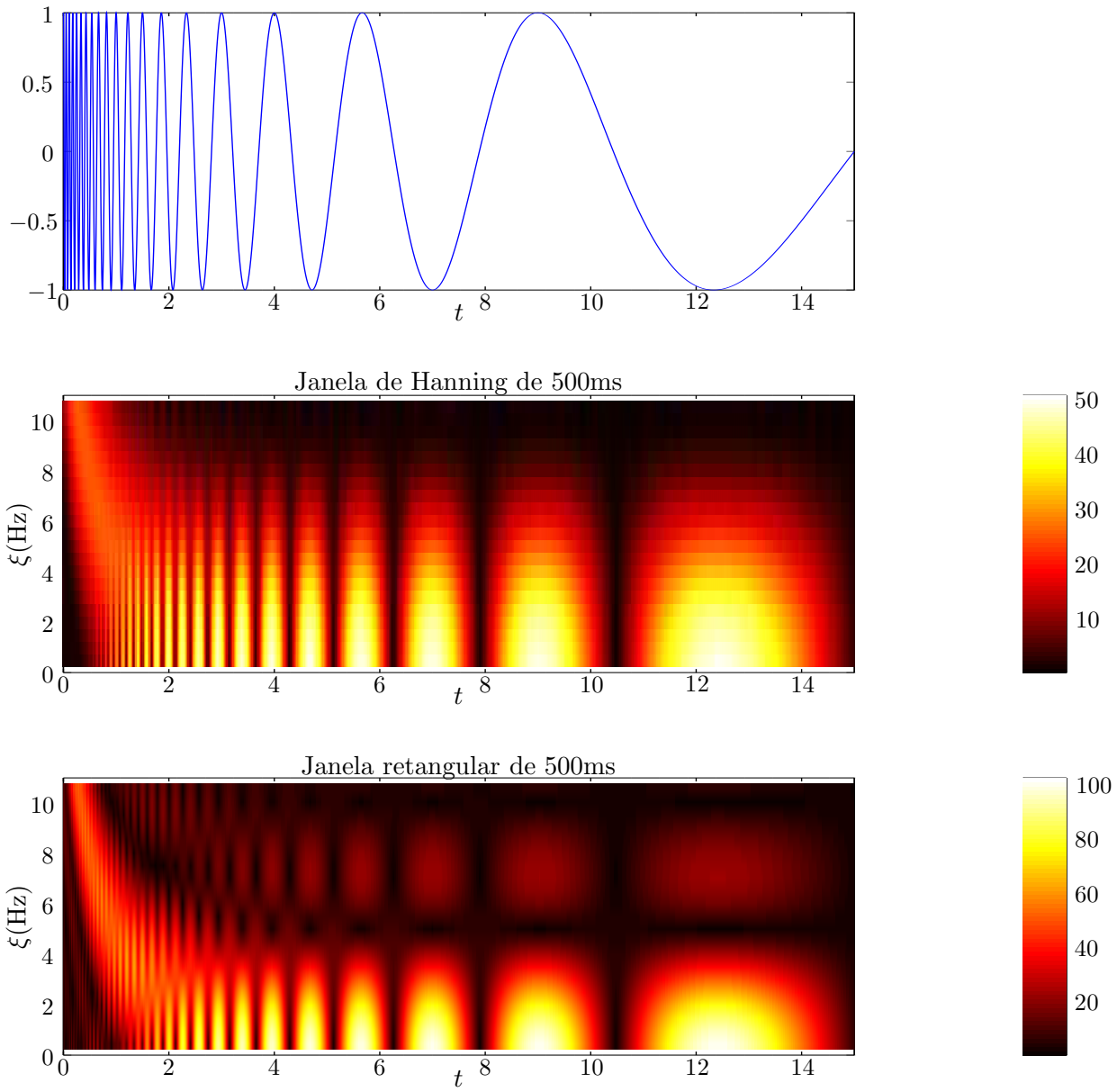


Figura 2.7: Espectrogramas de um chirp linear.

Neste exemplo, pode ser notado que, apesar da visualização com as transformadas *wavelet* são melhores, os espectrogramas ainda conseguem dar uma visualização razoável sobre os espectros deste sinal. Observa-se, no entanto, que a *wavelet* tem mais facilidade em analisar o sinal nas grandes escalas (baixas frequências), diferentemente da transformada janelada.

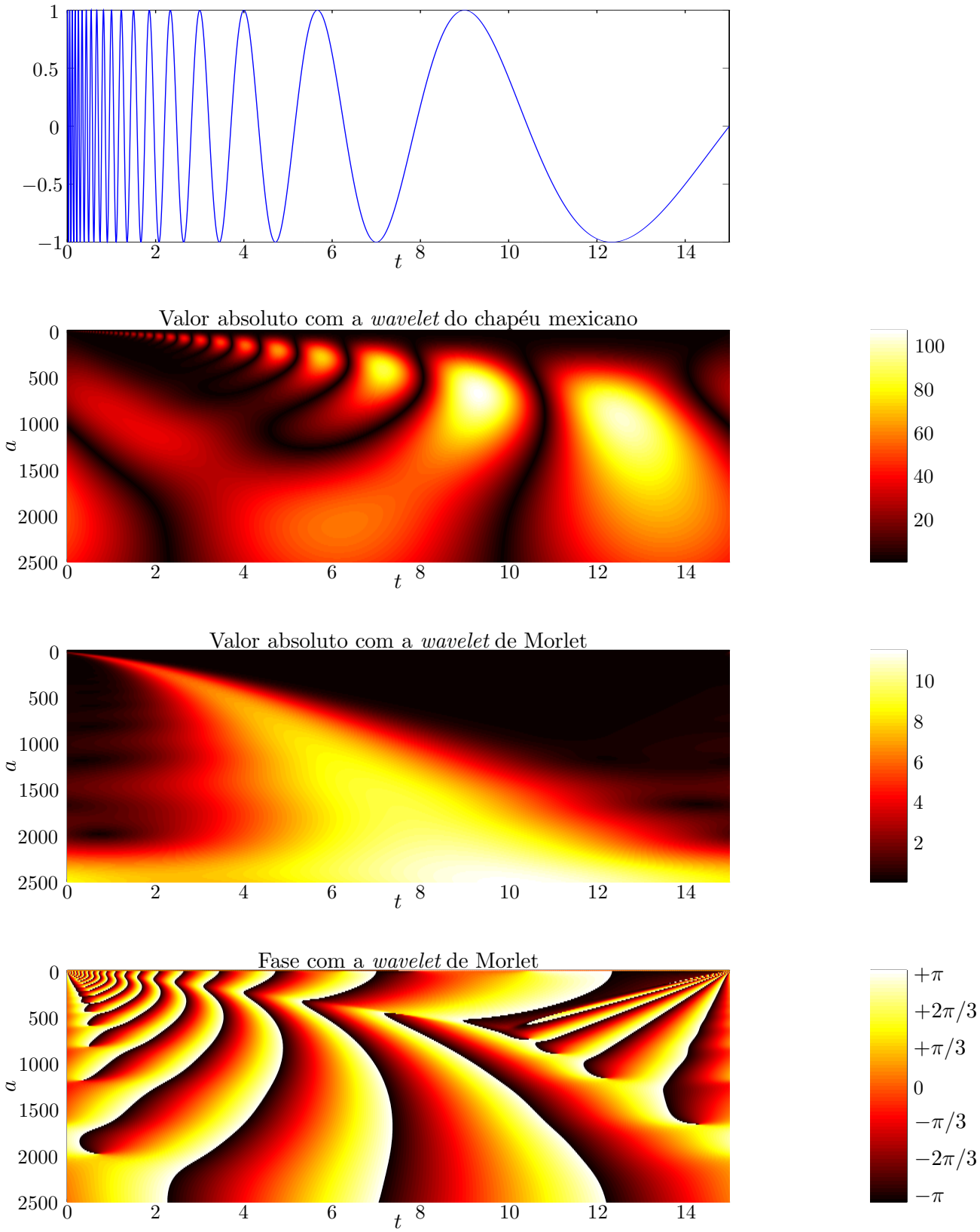


Figura 2.8: Escalogramas de um chirp linear.

2.3.3 Chirps Hiperbólicos

Neste exemplo, será indicada a facilidade maior que a transformada *wavelet* possui em relação a transformada janelada quando o sinal a ser analisado muda muito bruscamente de frequência.

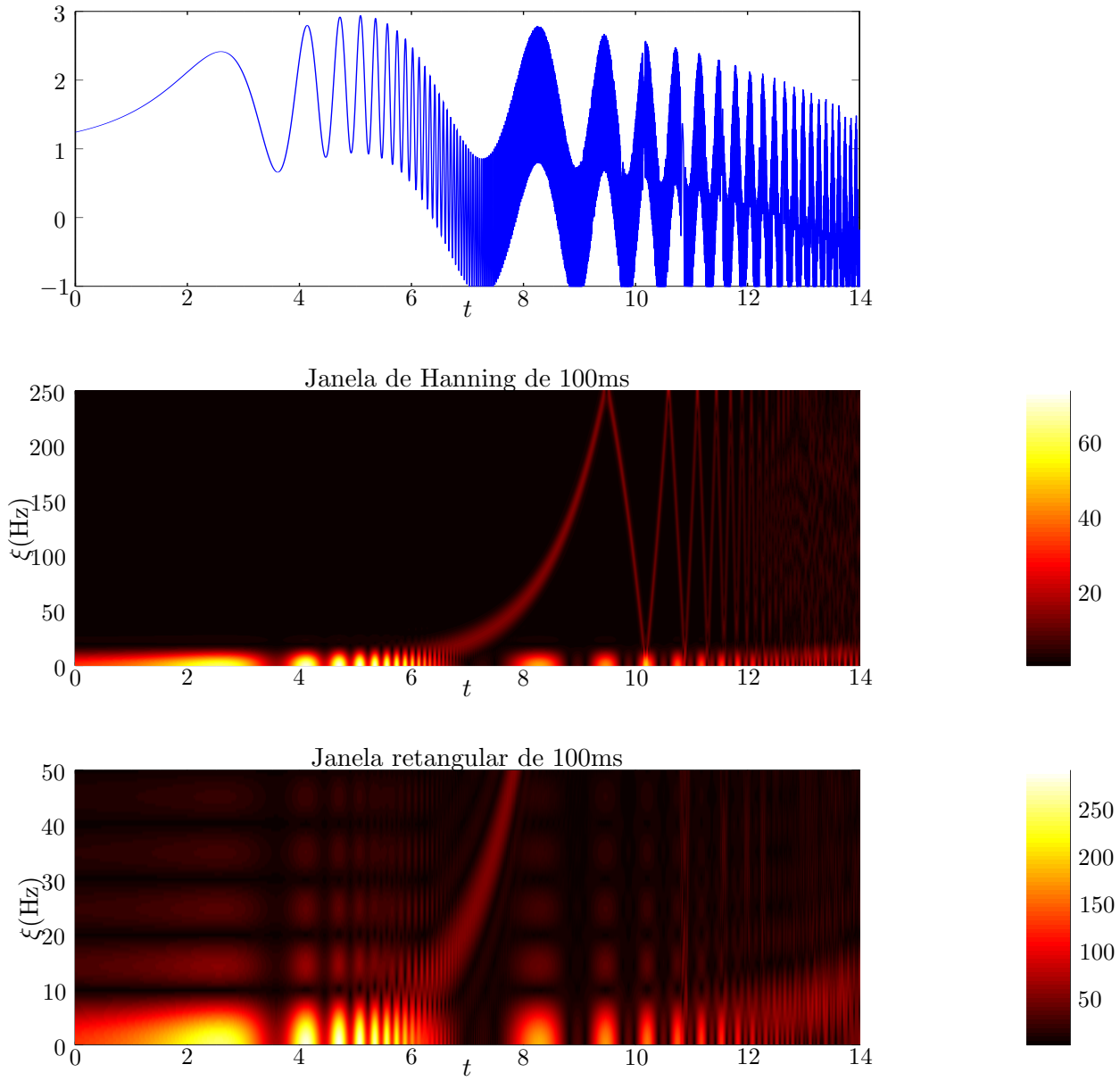


Figura 2.9: Espectrogramas de uma soma de chirps hiperbólicos.

Pode ser vista claramente a necessidade da transformada *wavelet*. Com os espectrogramas, pouca ou nenhuma informação útil pode ser retirada quanto se trata de sinais como estes, pois uma janela de tamanho fixo, de qualquer tamanho que seja, não irá produzir uma boa representação de sinais com variação tão alta.

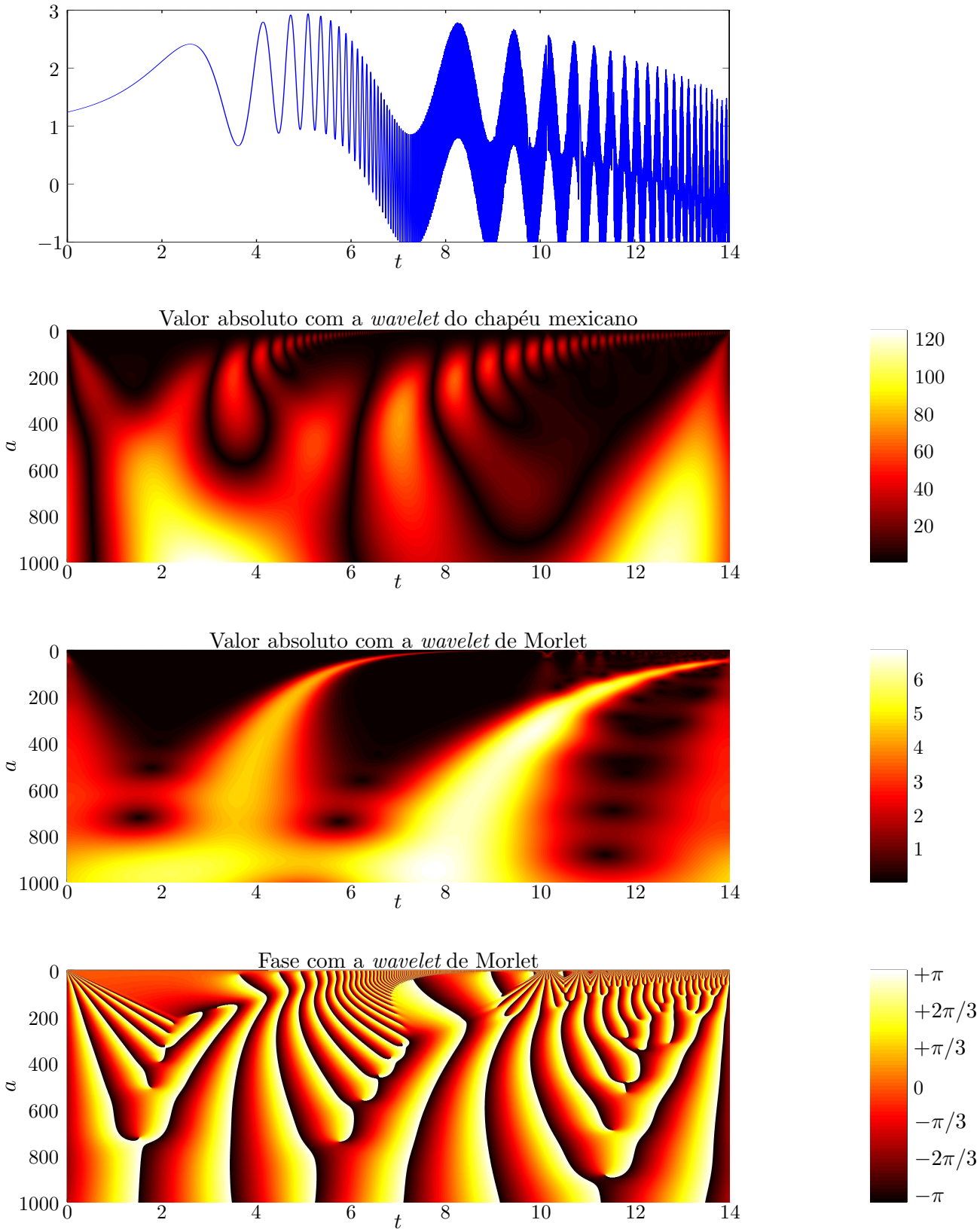


Figura 2.10: Escalogramas de uma soma de chirps hiperbólicos.

2.3.4 Função de Cantor (Devil's Staircase)

Outra função interessante de se analisar, é a função de Cantor, conhecida como "Escadaria do Diabo" (ou *Devil's Staircase*). Ela é uma função fractal que construída por iterações, com o código na página 156

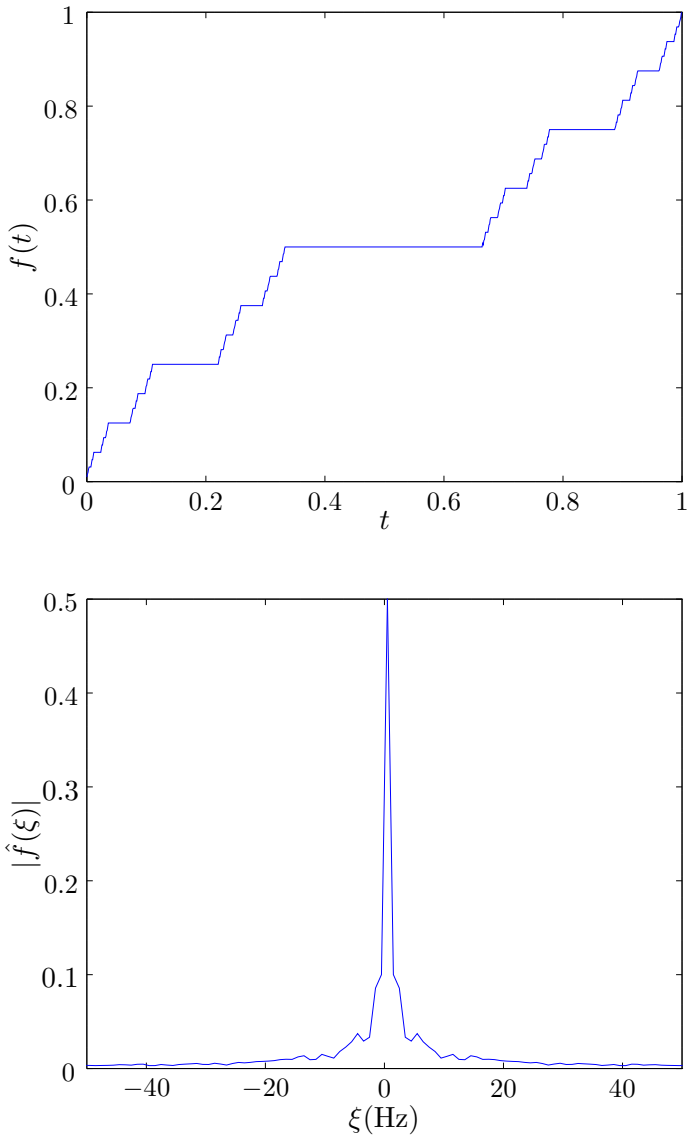


Figura 2.11: Acima, função de Cantor. Abaixo, seu espectro de *Fourier* .

De seu espectro, dá para se notar que a função parece ter uma baixa, porém presente, energia em todas as frequências (ela não se anula rapidamente). As imagens dos espectrogramas mostram que a transformada janelada de *Fourier* não consegue distinguir quais as frequências presentes nesta função. Para cada tamanho diferente da janela, a WFT indica frequências diferentes. Com as transformadas *wavelet* , isto não ocorre, e elas mostram que a função também possui estrutura fractal em sua representação de tempo-escala, o que pode ser visto na Figura 2.13 e 2.14

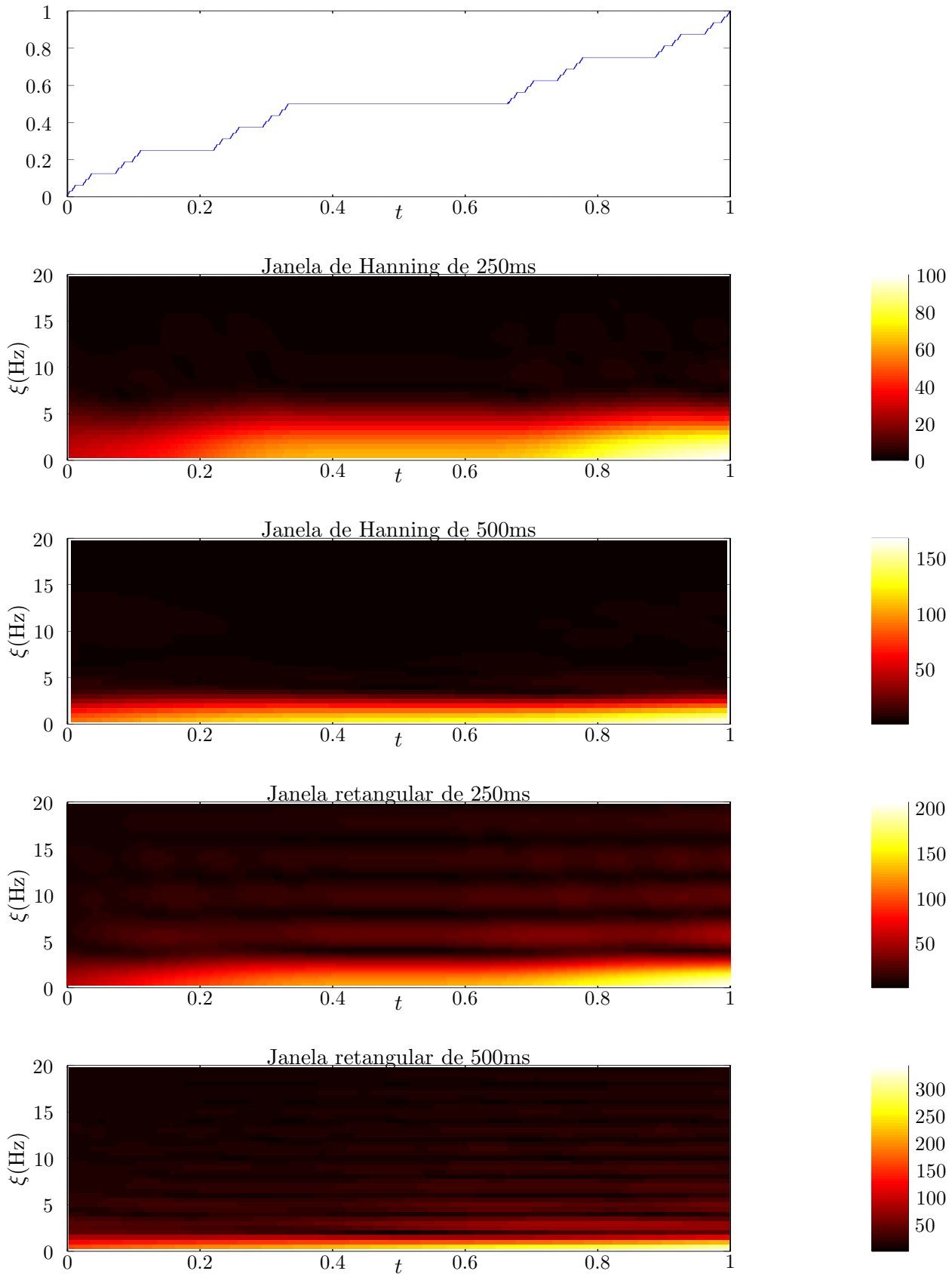


Figura 2.12: Espectrogramas da função de Cantor. Pode ser observado destas imagens que cada janela obteve frequências distintas (os eixos da escala se mantiveram os mesmos, e ainda assim no gráfico as frequências se acumularam mais para baixo (nas maiores escalas) ou para cima).

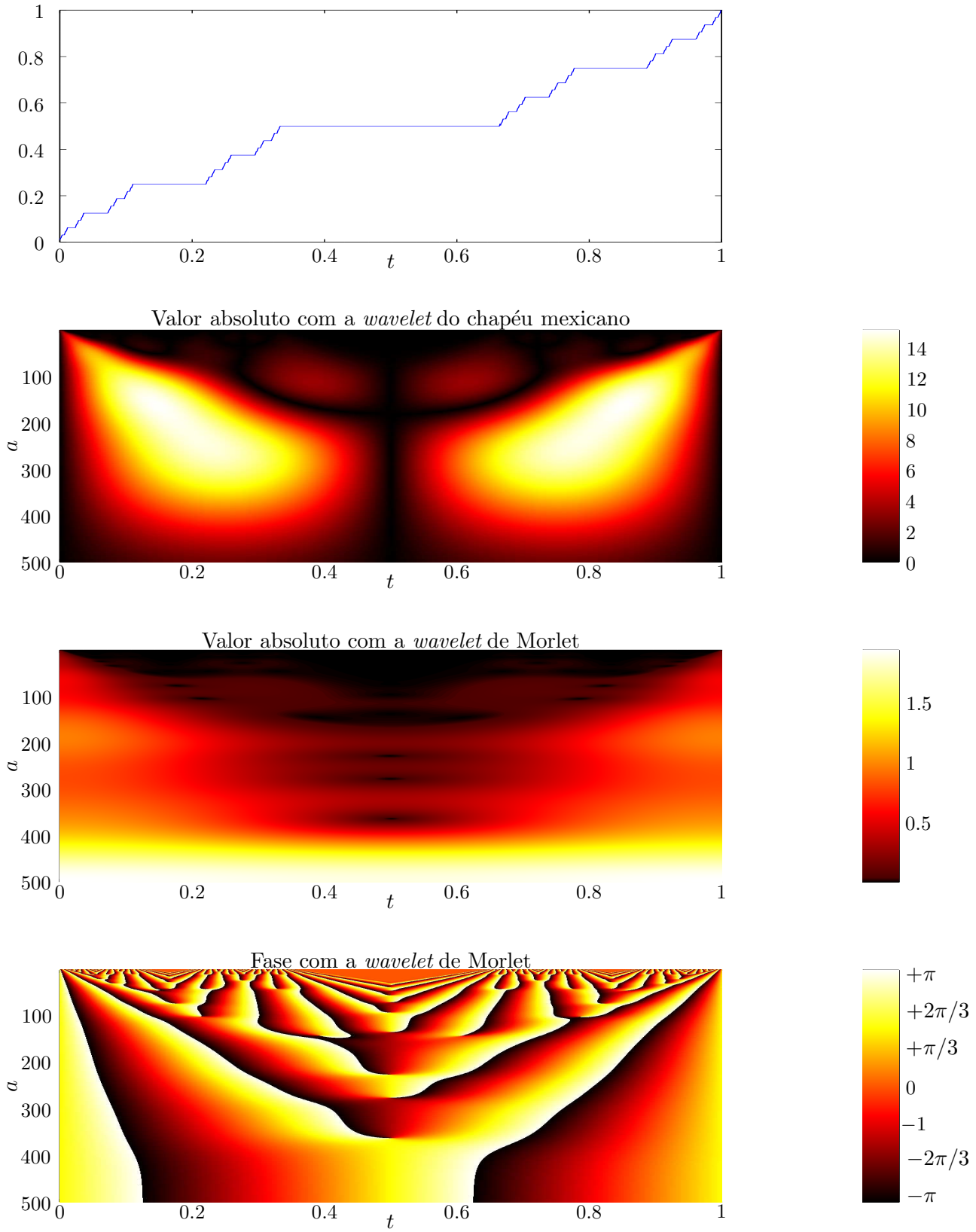


Figura 2.13: Escalogramas da função de Cantor.

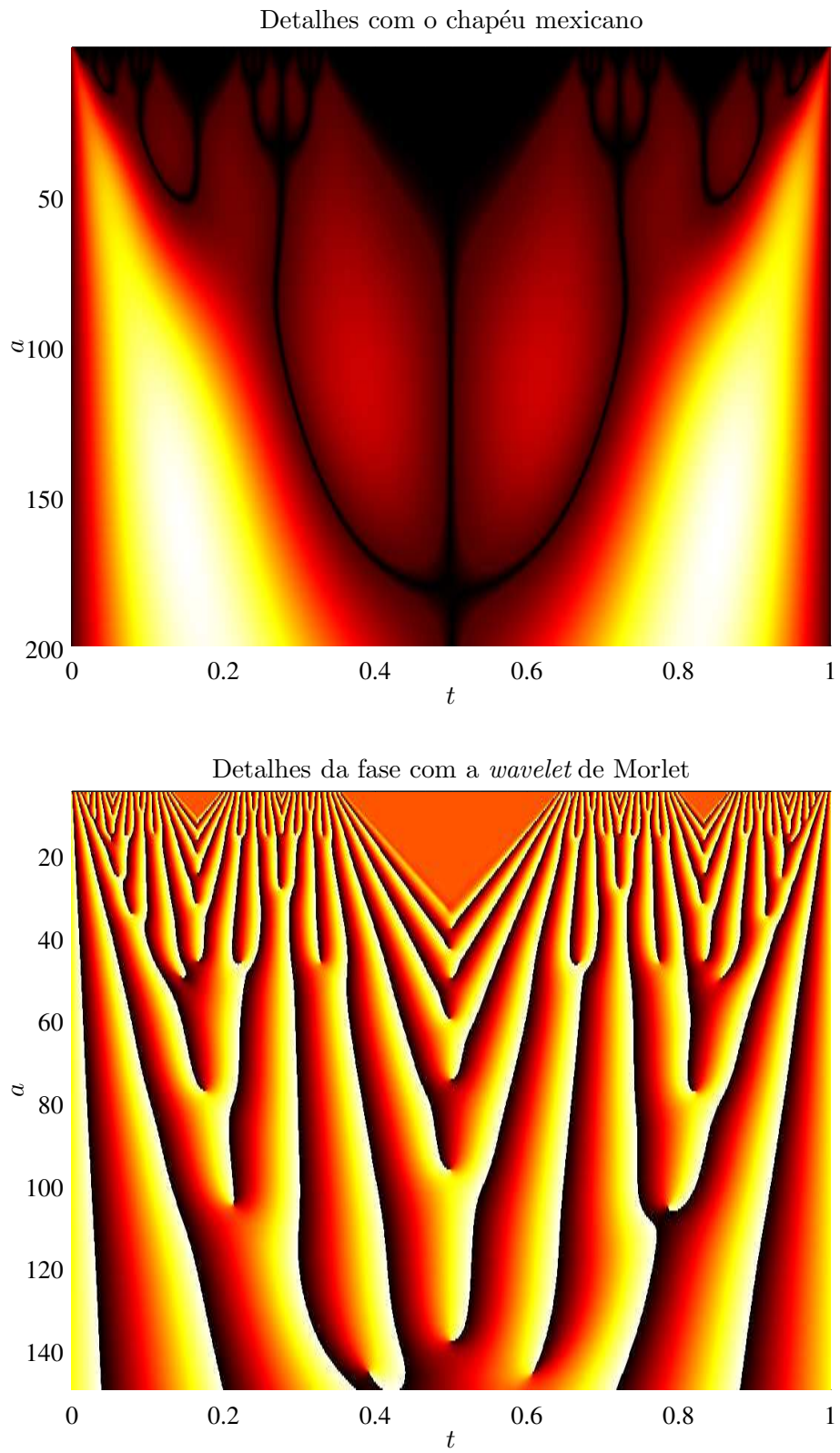


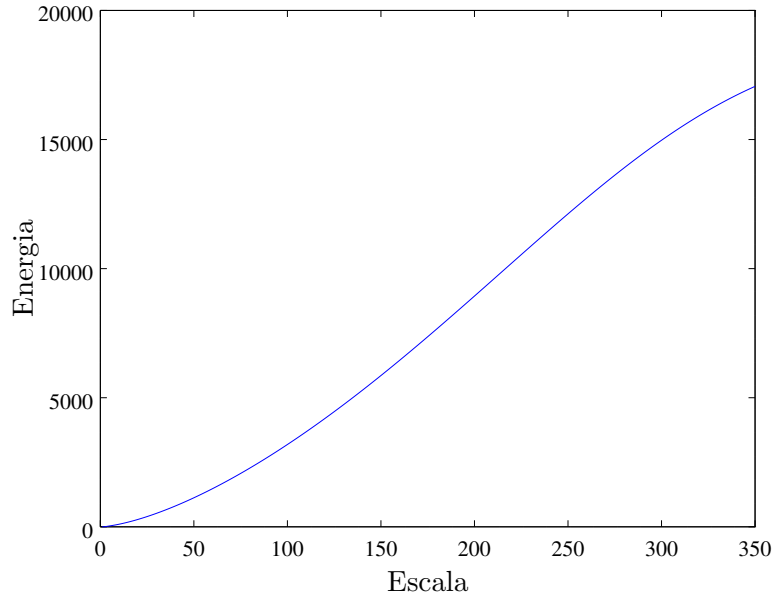
Figura 2.14: Acima, aproximação do valor absoluto da CWT com chapéu mexicano. Abaixo, aproximação da fase com a *wavelet* de Morlet. Observa-se que ambas possuem estrutura fractal. Observe que a palheta de cores usada na aproximação da transformada com chapéu mexicano é um pouco diferente, destacando melhor as estruturas presentes.

2.4 Escalograma Global

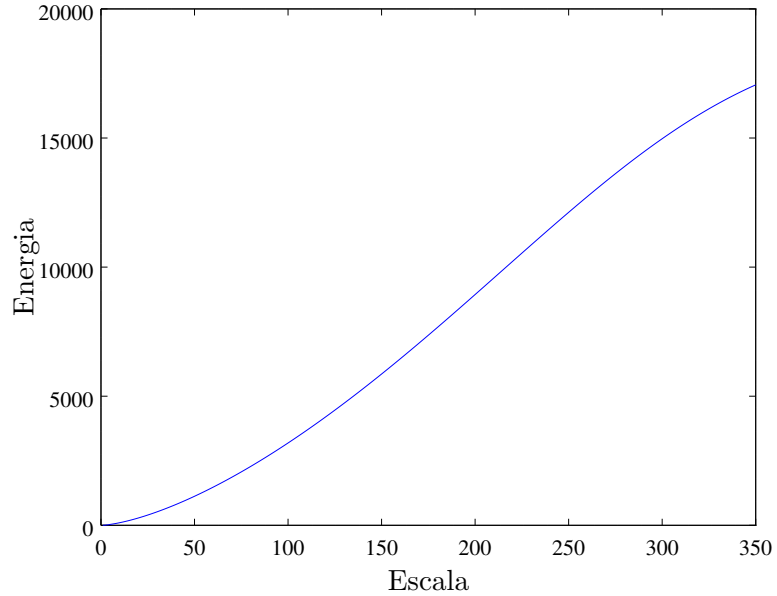
Uma informação que pode ser útil da transformada, é o escalograma global. Ele indica a energia que o sinal possui em cada escala. Pode ser calculado como:

$$E(a) = \int |\mathfrak{W}_f^{\psi}(a,b)|^2 db \tag{2.8}$$

Ao fazer isto, perde-se a informação temporal e mantém-se uma representação somente de escala. De certo modo, é análogo ao escalograma comum da mesma maneira que um espectro de frequências é análogo a um espectrograma.

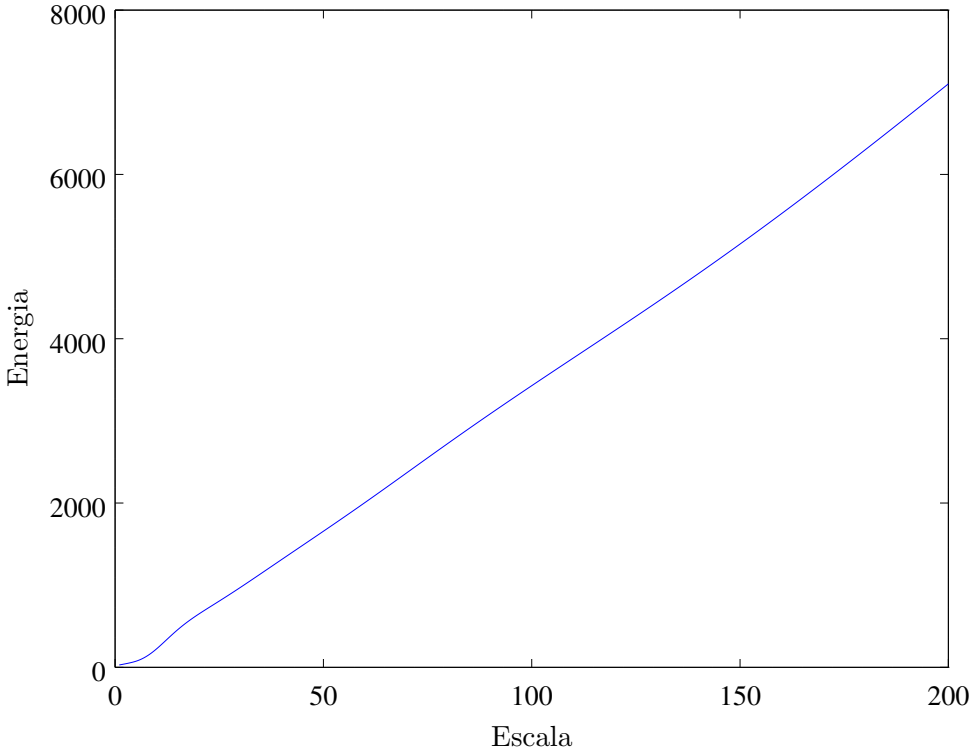


a)

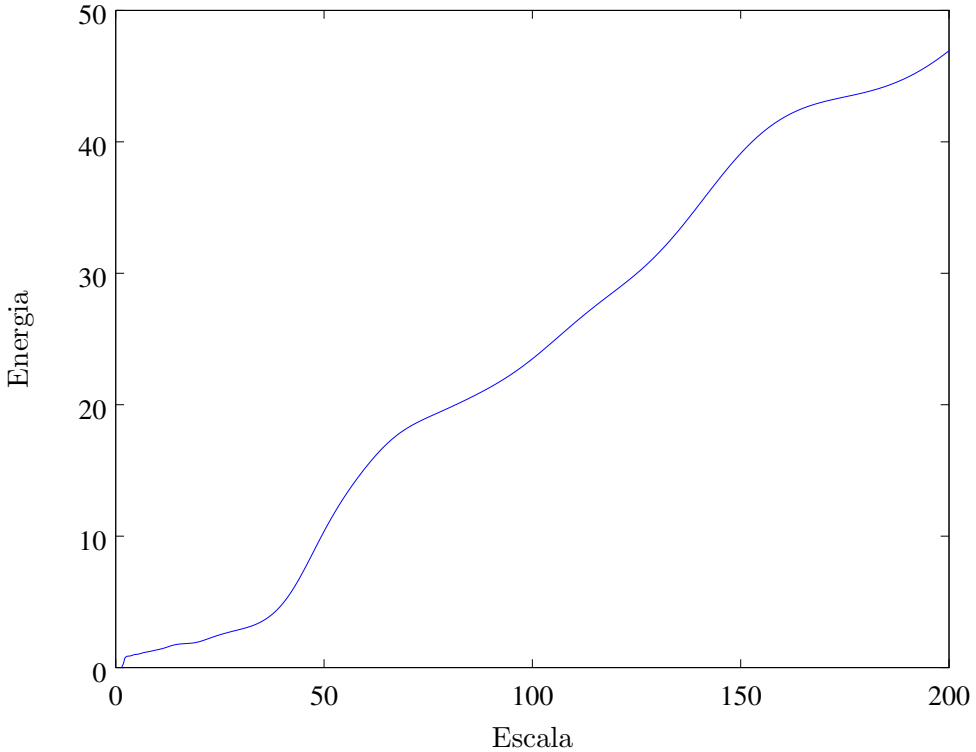


b)

Figura 2.15: Escalogramas globais de um chirp linear com a *wavelet* do chapéu mexicano (Figura a) e a *wavelet* de Morlet (Figura b).

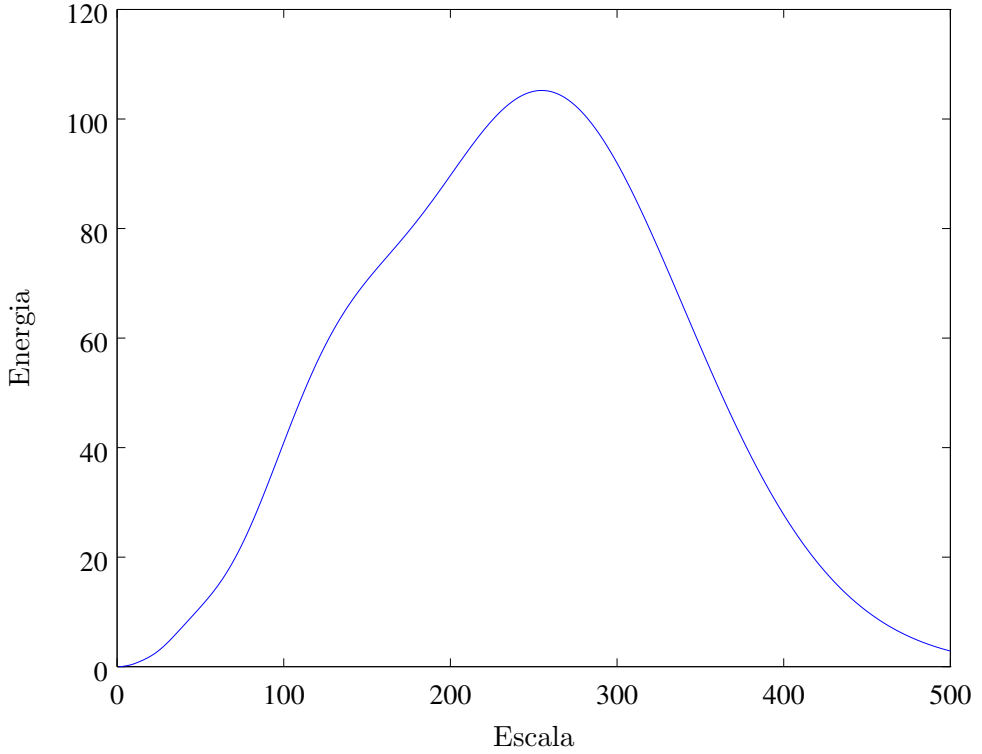


a)

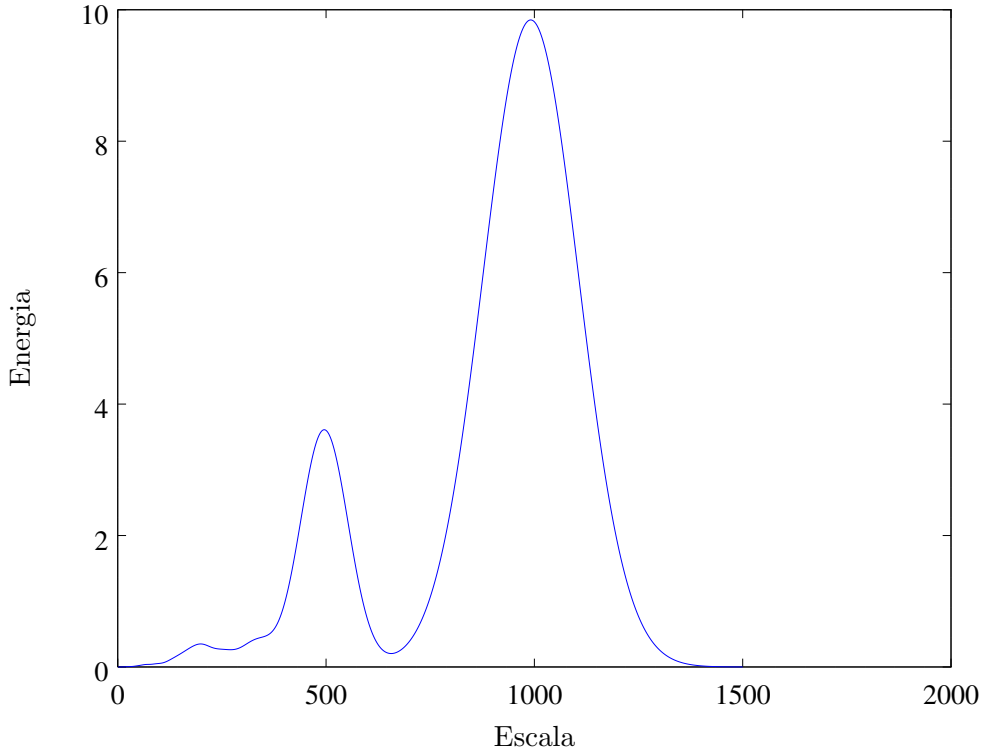


b)

Figura 2.16: Escalogramas globais de uma soma de chirps hiperbólicos com a *wavelet* do chapéu mexicano (Figura a) e a *wavelet* de Morlet (Figura b).



a)



b)

Figura 2.17: Escalogramas globais da função de Cantor com a *wavelet* do chapéu mexicano (Figura a) e a *wavelet* de Morlet (Figura b).

2.5 Transformada *wavelet* com normalização diferente

Uma possibilidade também na hora de se calcular a transformada *wavelet* é a de se fazer uma normalização diferente. Olhando novamente a fórmula

$$\mathfrak{W}_f^{\psi}(a,b) = \int f(t)\bar{\psi}_{a,b}(t)dt \quad a > 0, \tag{2.9}$$

pode ser feita, diferentemente de como é definido antes, a seguinte mudança:

$$\psi_{a,b} = \frac{1}{a}\psi\left(\frac{t-b}{a}\right)$$

Onde $\frac{1}{a}$ é escolhido no lugar do usual $\frac{1}{\sqrt{a}}$ Isto gera transformadas um pouco diferentes, em que se destrói a capacidade da transformada de manter a energia do sinal, rejeitando a relação de Parseval, porém é mantida a amplitude.

2.5.1 Exemplo

Novamente será analisado o chirp linear, porém comparando com esta nova normalização. Na Figura 2.18, vê-se uma comparação das transformada com duas *wavelets* reais, e com as duas normalizações. A primeira é a *wavelet* do chapéu mexicano, a outra é a *wavelet* de Shannon definida como:

$$\psi(t) = \operatorname{sinc}\left(\frac{t}{2}\right)\cos\left(\frac{3\pi t}{2}\right) \tag{2.10}$$

em que $\operatorname{sinc}(t)$ denota $\operatorname{sen}(\pi t)/(\pi t)$.

Na Figura 2.19 foram usadas a *wavelet* de Morlet e a versão complexa da *wavelet* de Shannon, definida agora como:

$$\psi(t) = \operatorname{sinc}(t)e^{-\imath 2\pi t} \tag{2.11}$$

e na Figura 2.20 se mostra as fases usando as mesmas *wavelets* complexas. Observe que, na CWT feita com a *wavelet* real de Shannon, observa-se este "resíduo"curvo na imagem. Isto acontece pelo formato da *wavelet* , que como pode ser visto em seu gráfico na página 19, não possui suporte compacto e continua oscilando por um bom tempo antes de se anular (ou melhor, se aproximar o suficiente de zero).

Nota-se que nas imagens dos valores absolutos, muda-se um pouco as cores e seus valores, mas como a amplitude e a energia estão relacionadas, a forma *global* dos gráficos não muda. Nas imagens das fases, não é notada diferença alguma entre as normalizações.

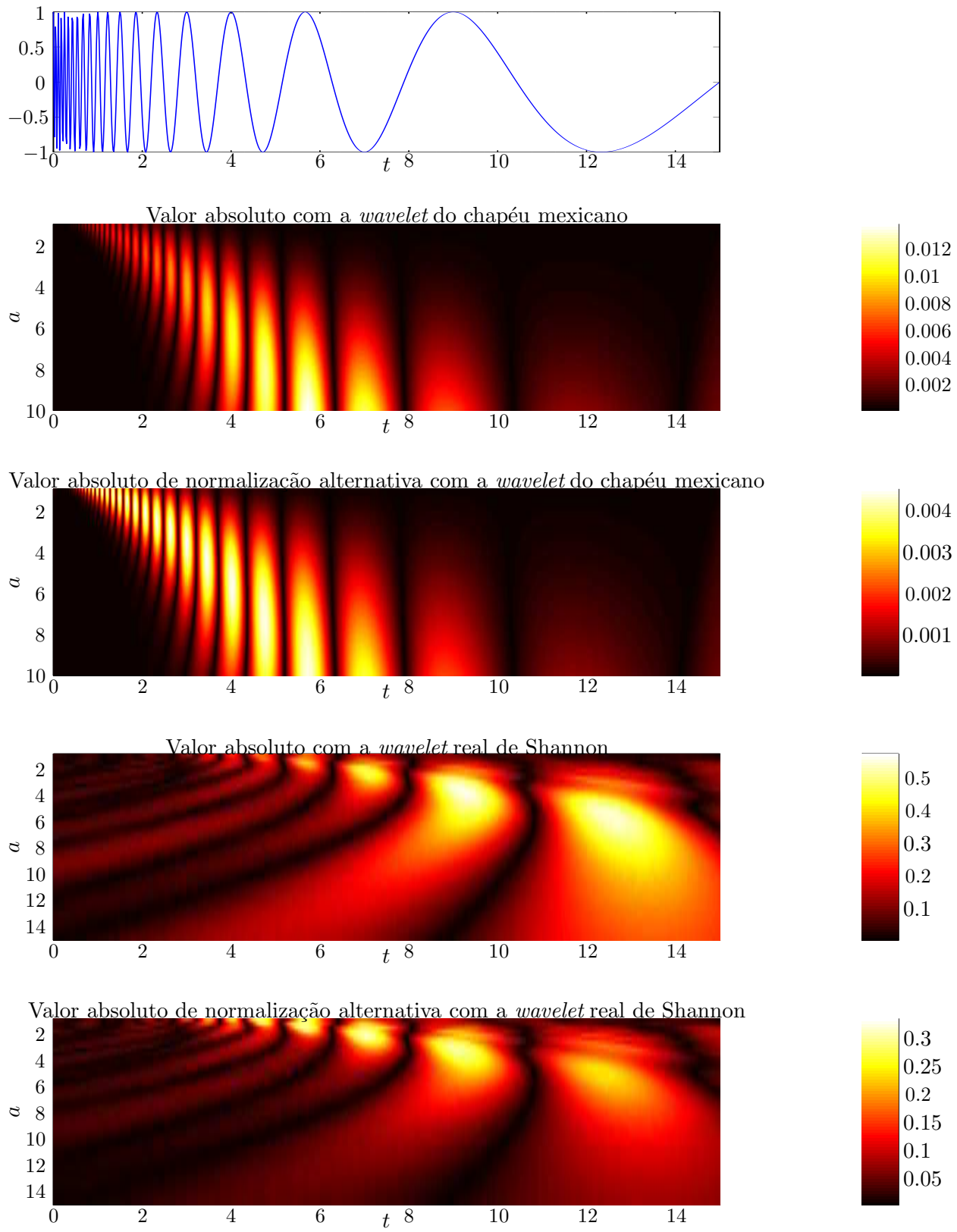


Figura 2.18: Comparação dos escalogramas de um chirp linear com a *wavelet* do chapéu mexicano e a *wavelet* real de Shannon, ambas com duas normalizações diferentes

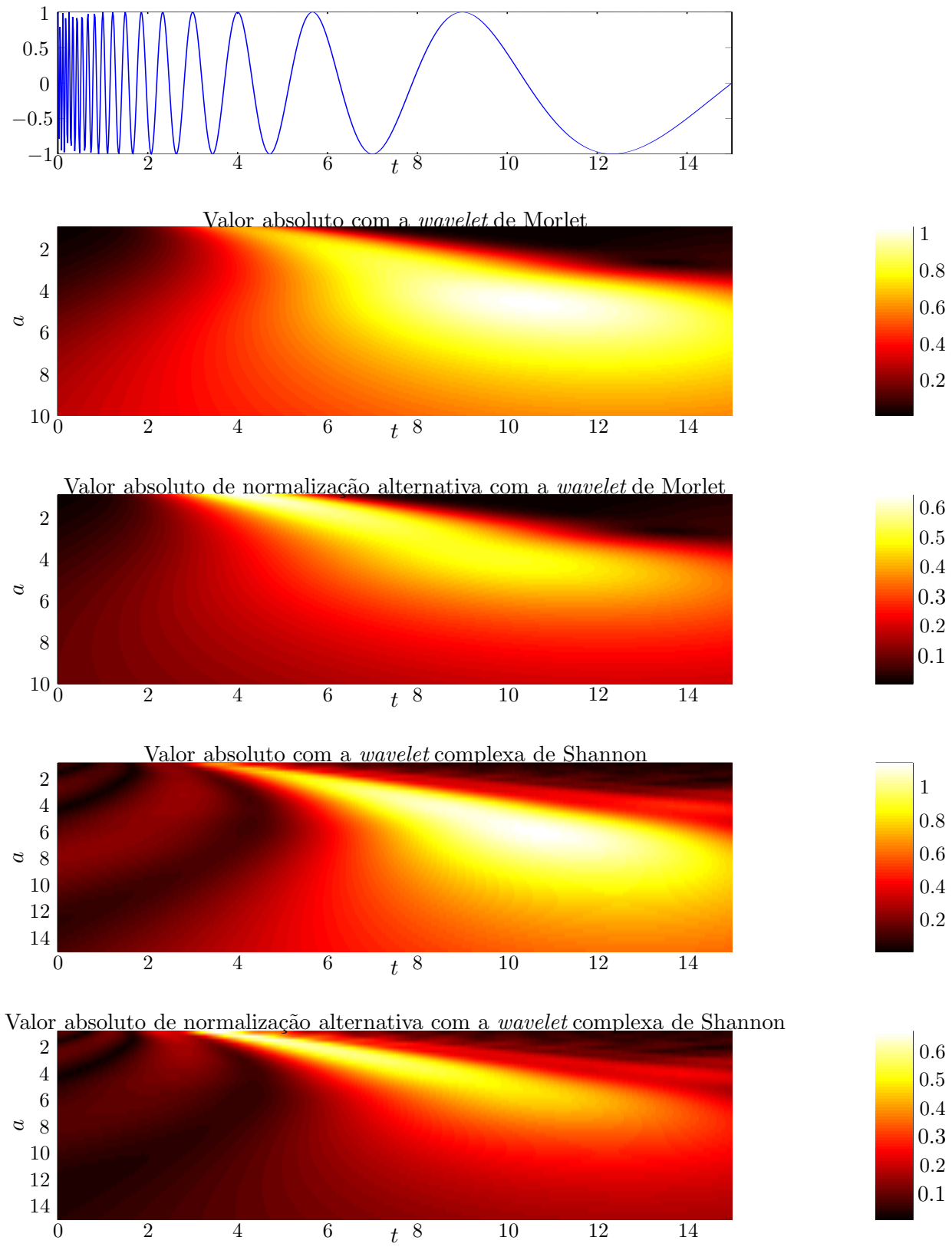


Figura 2.19: Comparação dos escalogramas de um chirp linear com as *wavelet* de Morlet e a *wavelet* complexa de Shannon, ambas com duas normalizações diferentes

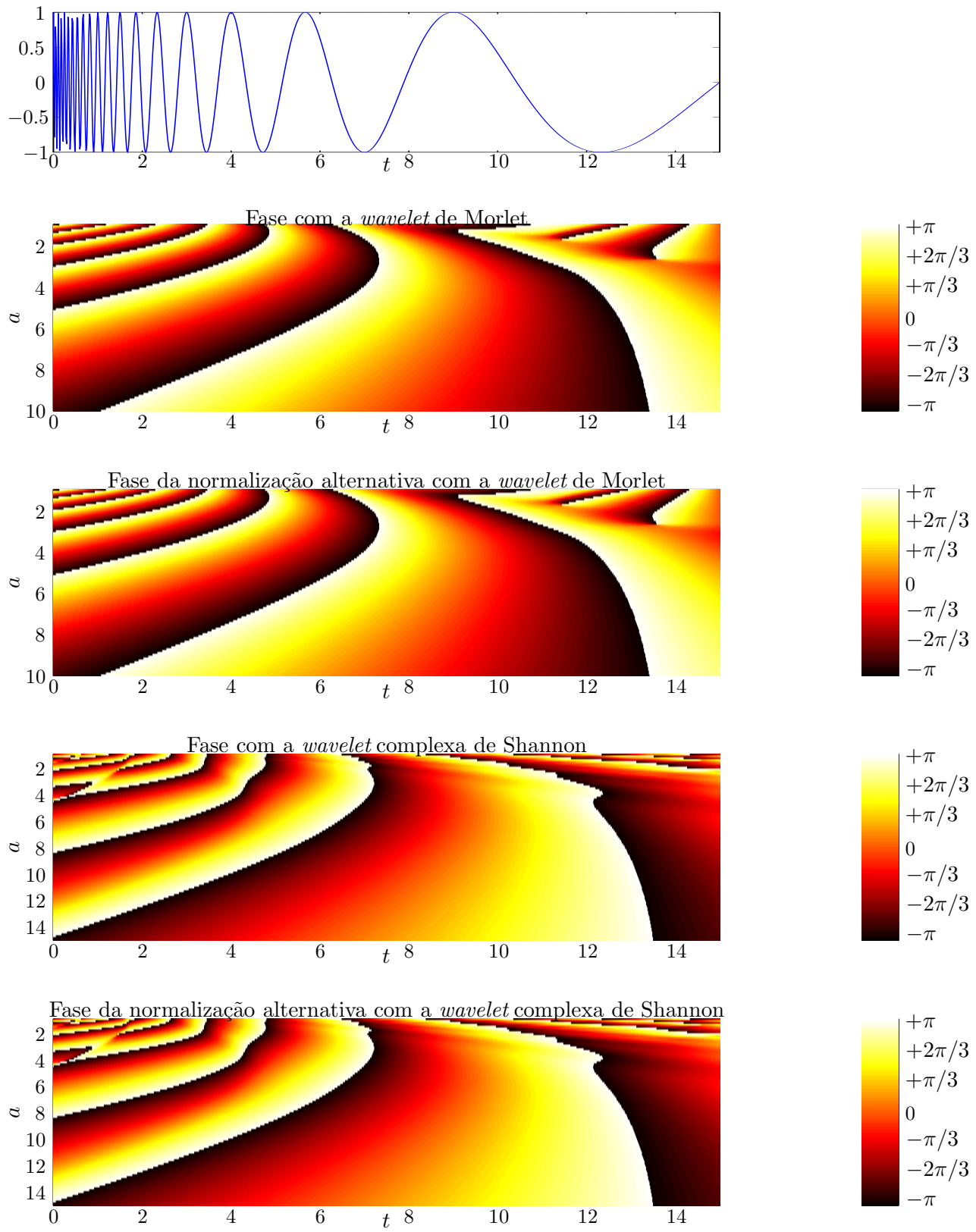


Figura 2.20: Comparação das fases de um chirp linear com as *wavelet* de Morlet e a *wavelet* complexa de Shannon, ambas com duas normalizações diferentes

Capítulo 3

Palhetas de Cores

Uma coisa importante também para se analisar ao usar um gráfico para demonstrar um escalograma, é a palheta de cores usada, pois palhetas de cores diferentes podem demonstrar melhor certas características presentes em um gráfico que se deseja demonstrar.

Este capítulo explicará resumida e brevemente as ideia de demonstração de imagens científicas bidimensionais de Marie Farge [6], que por sua vez é baseada na teoria das cores de Johannes Itten, para a escolha de uma palheta de cores objetiva que consiga passar a informação sem ambiguidades.

3.1 Comparação de algumas palhetas

Para demonstrar a importância da escolha das palhetas, serão analisadas varias palhetas simples de cores para o mesmo detalhe do escalograma da função de Cantor com *wavelet* de chapéu mexicano. Os gráficos das palhetas possuem 4 curvas. Três delas indicam a intensidade de cada uma das cores do formato RGB de síntese aditiva, padrão muito usado nos computadores. A outra é a curva que indica a média entre as três curvas anteriores, indicando a luminosidade total da palheta.

Primeiramente, tem-se a palheta padrão do MATLAB e GNU/Octave, a palheta *Jet*. A palheta *Jet* não é uma boa opção para este caso porque, olhando seu gráfico, vemos que a intensidade média não muda muito, indo de 0.2 até 0.7 aproximadamente, e além disso, seus valores máximo e mínimo possuem a mesma intensidade. Ela poderia ser aproveitada, talvez, para apresentar gráficos onde se deseja distinguir valores negativos e positivos, mas este não é o caso do exemplo. Em seguida, tem-se a palheta *HSV*. Esta palheta apresenta o mesmo problema da palheta anterior no que diz respeito a intensidade de seus valores máximo e mínimo, porém é muito pior pelo fato de sua intensidade média ser oscilante. Com ela, dificilmente pode ser observado algum detalhe útil nesta imagem. A comparação destas duas palhetas está na Figura 3.1

A terceira palheta será a que foi escolhida como padrão de uso durante os exemplos anteriores deste trabalho, a palheta *Hot*. Dela pode ser observado claramente que nenhum dos dois problemas que existem nas palhetas anteriores estão presentes. E isso não se dá ao fato do tom avermelhado da palheta, e para demonstrar isso, foi usada a webpage colormap.org para criar duas palhetas customizadas. Elas são análogas a palheta *Hot*, porém com tons azulados (chamada de "*Blue*"). A comparação de ambas está na Figura 3.2.

A palheta utilizada no detalhe do exemplo da função de Cantor, entretanto, foi uma modificação da palheta *Hot*, que pode ser vista na comparação da Figura 3.3. Ela foi modificada para aumentar a intensidade média no início dos valores, para que alguns detalhes mais fracos apareçam. Foi colocada também os mesmos escalogramas nestas duas palhetas em tamanho maior na Figura 3.4, para uma comparação melhor.

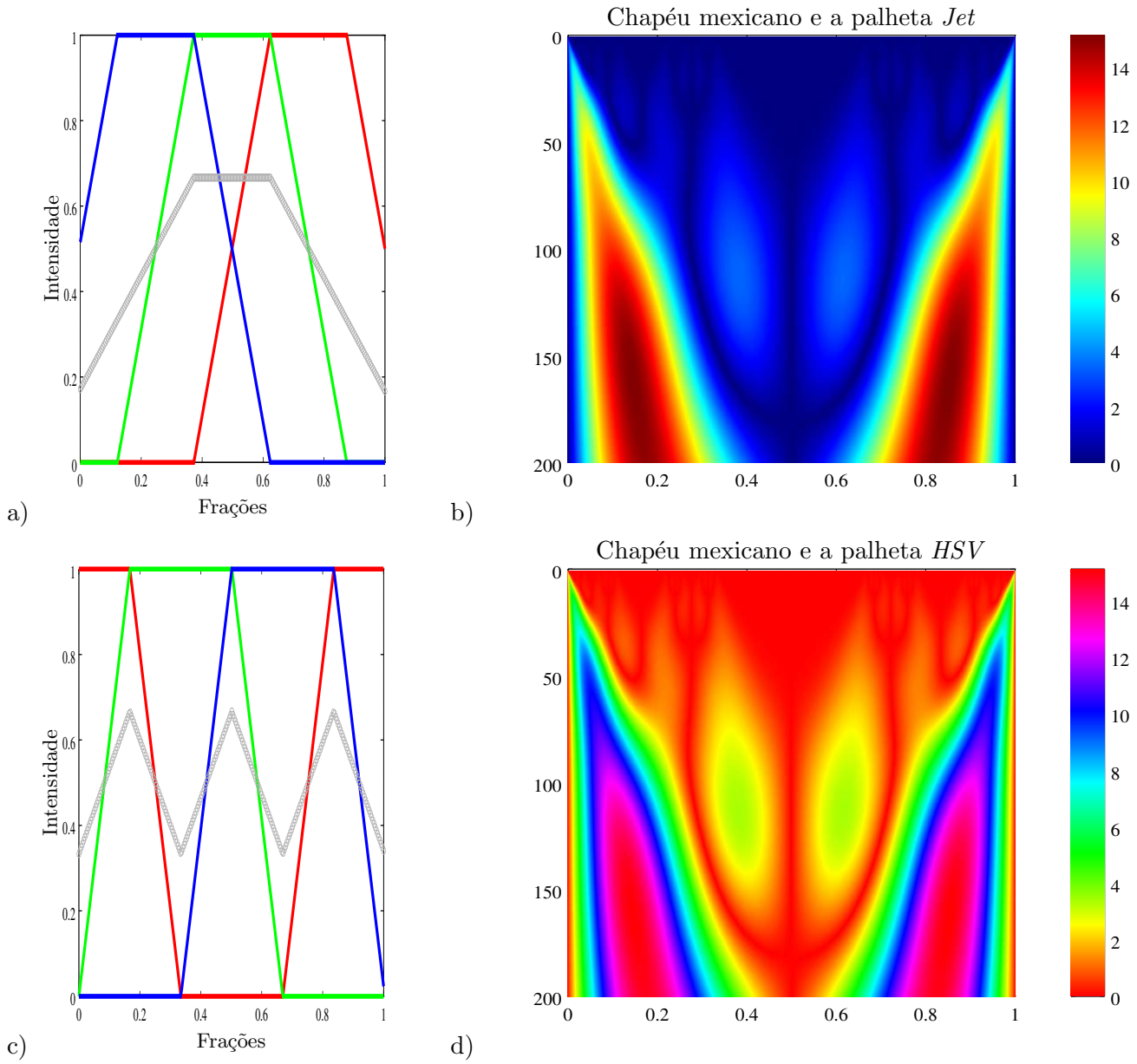


Figura 3.1: Comparação do uso das palhetas *Jet* e *HSV*. Na Figura a, está a palheta *Jet*, e na Figura b, seu uso no escalograma da função de cantor. Na Figura c, está a palheta *HSV*, e na Figura d, seu uso para o mesmo escalograma.

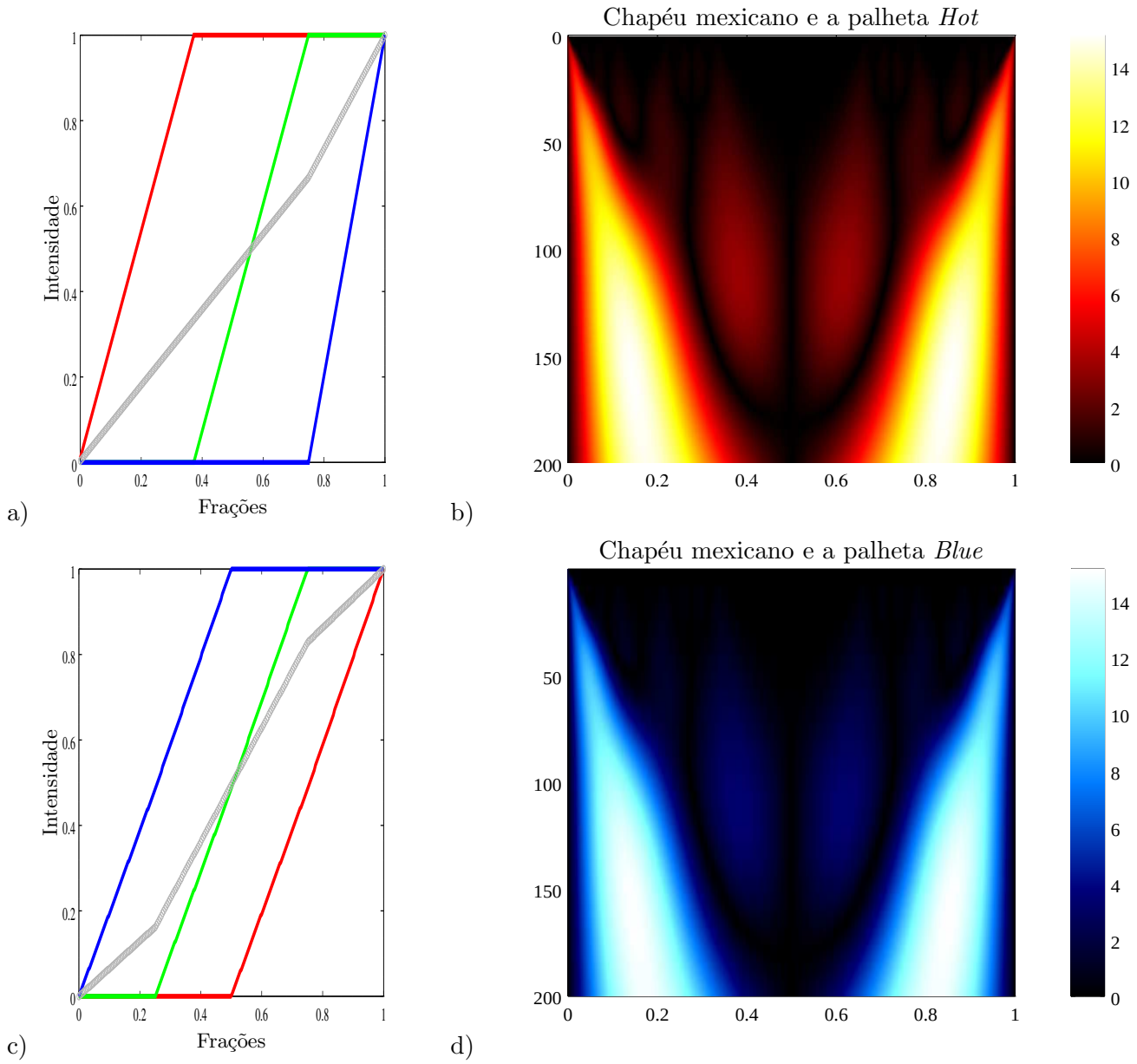


Figura 3.2: Comparação do uso das palhetas *Hot* e *Blue*. Na Figura a, está a palheta *Hot*, e na Figura b, seu uso no escalograma da função de cantor. Na Figura c, está a palheta *Blue*, e na Figura d, seu uso para o mesmo escalograma.

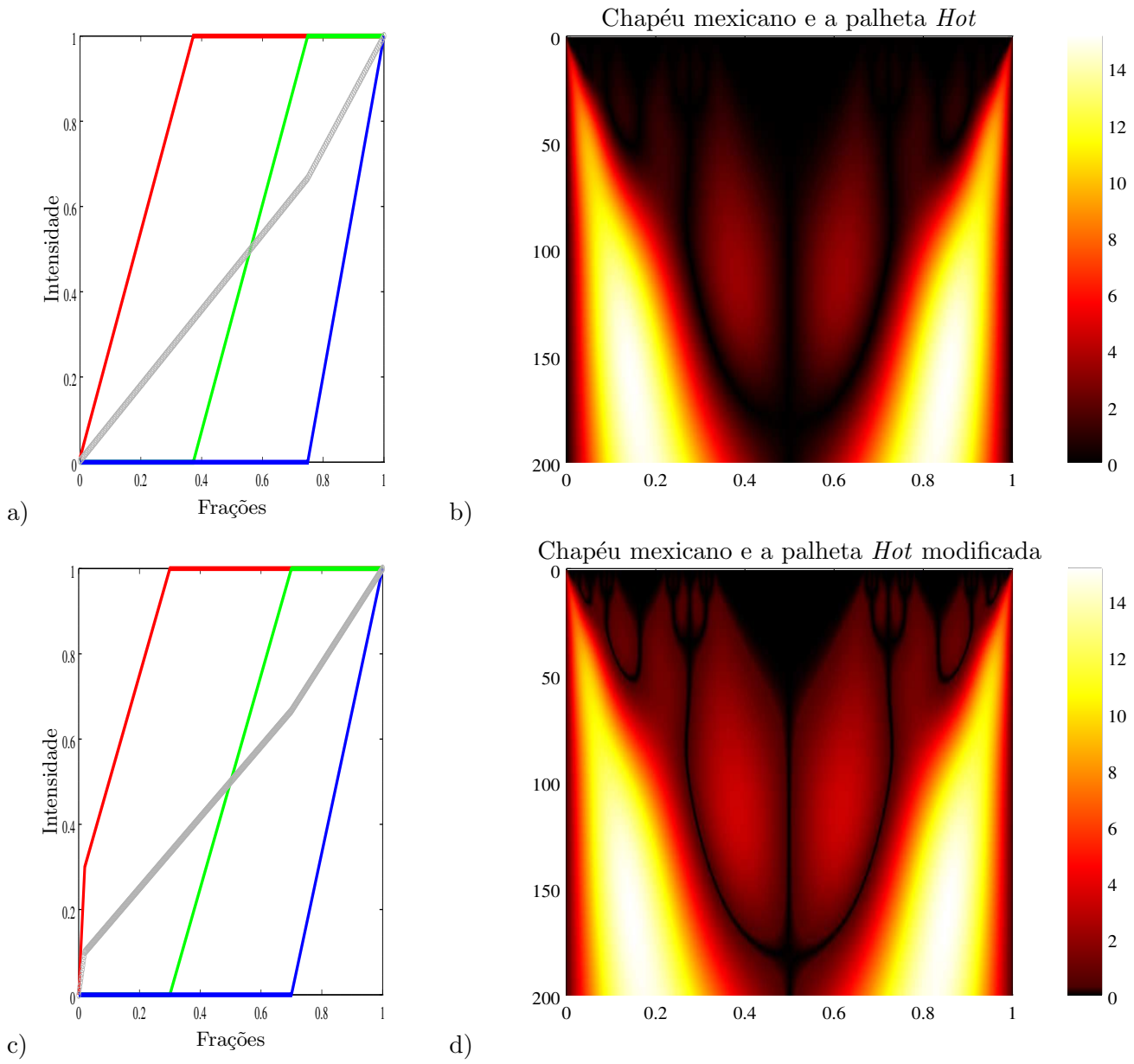


Figura 3.3: Comparação do uso das palhetas *Hot* nas versões original e modificada. Na Figura a, está a palheta original, e na Figura b, seu uso no escalograma da função de cantor. Na Figura c, está a palheta modificada, e na Figura d, seu uso para o mesmo escalograma. Nota-se como a palheta modificada teve sua curva de vermelho "levantada" na parte inicial dos valores.

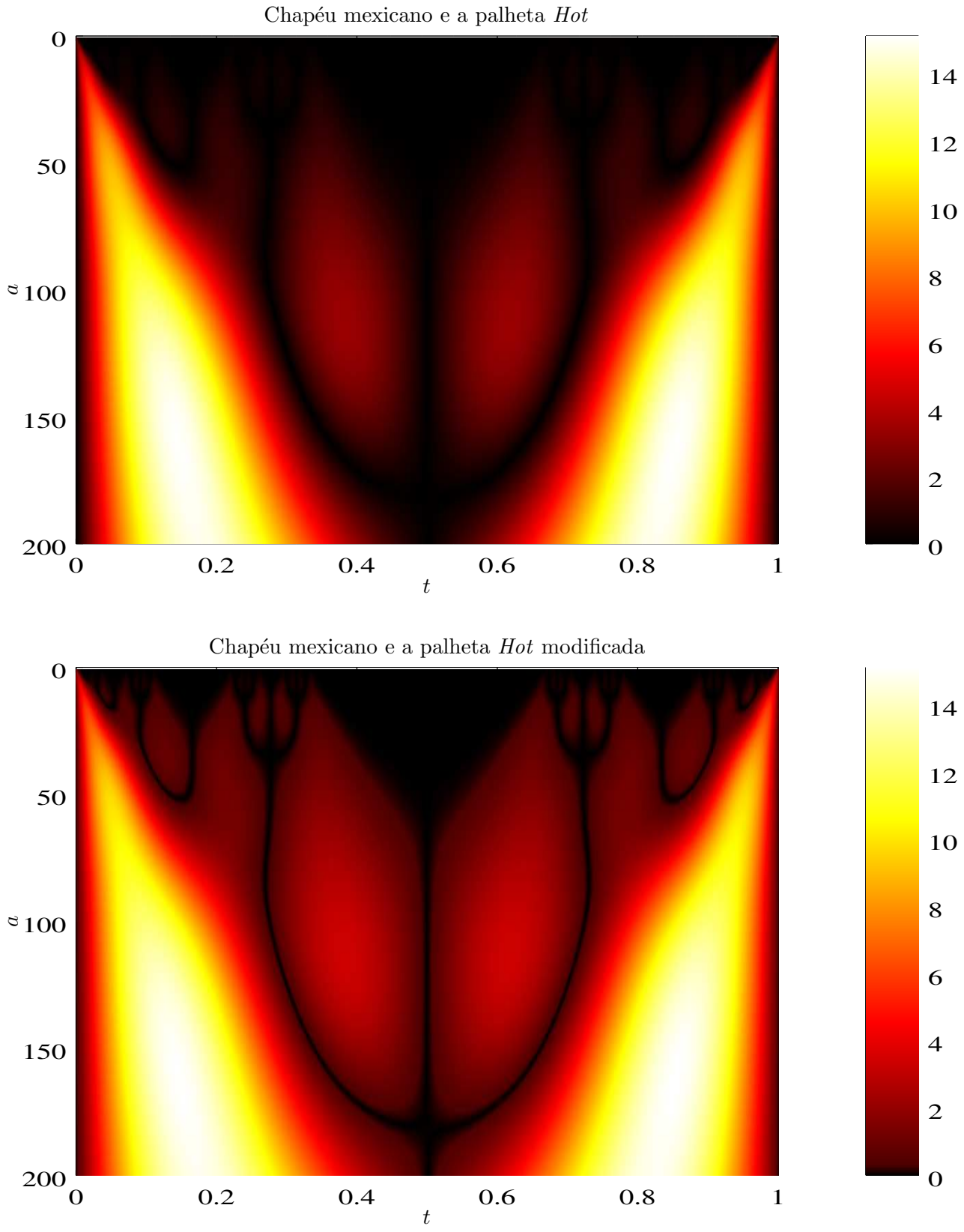


Figura 3.4: Comparação do uso das palhetas *Hot*. É possível perceber que, apesar da palheta original já indicar a presença destes formatos de tridente, a palheta modificada os deixa ainda mais evidentes.

3.2 Teoria das Cores

3.2.1 Definições

Espectro de cores

Cores monocromáticas de espectro discreto (Figura 3.5) porem ser obtidas pela decomposição de um raio de luz em um prisma, e indicam raios de luz de apenas uma frequência. São aproximadamente iguais as cores do arco iris.

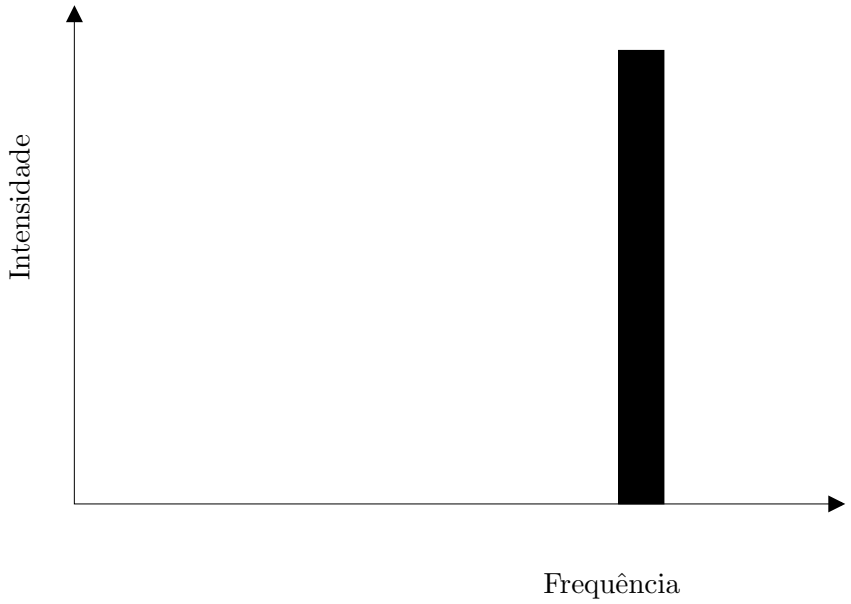


Figura 3.5: Cor monocromática de espectro discreto.

Cores não monocromáticas de espectro discreto (Figura 3.6) são cores que não estão no arco iris e se formam com a junção de outras cores monocromáticas de espectro discreto.

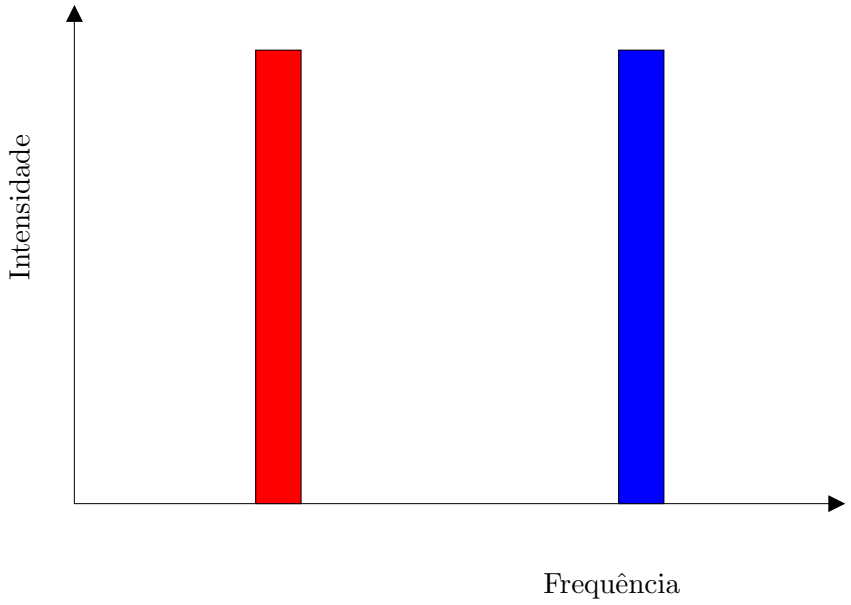


Figura 3.6: Cor não monocromática de espectro discreto.

Cores de espectro contínuo (Figura 3.7) são cores que possuem vários espectros continuamente e pode se apresentar monocromaticamente o espectro se comporte como uma frequência v_{max}, e a cor branca é uma cor de espectro contínuo com energia igual em todas as frequências.

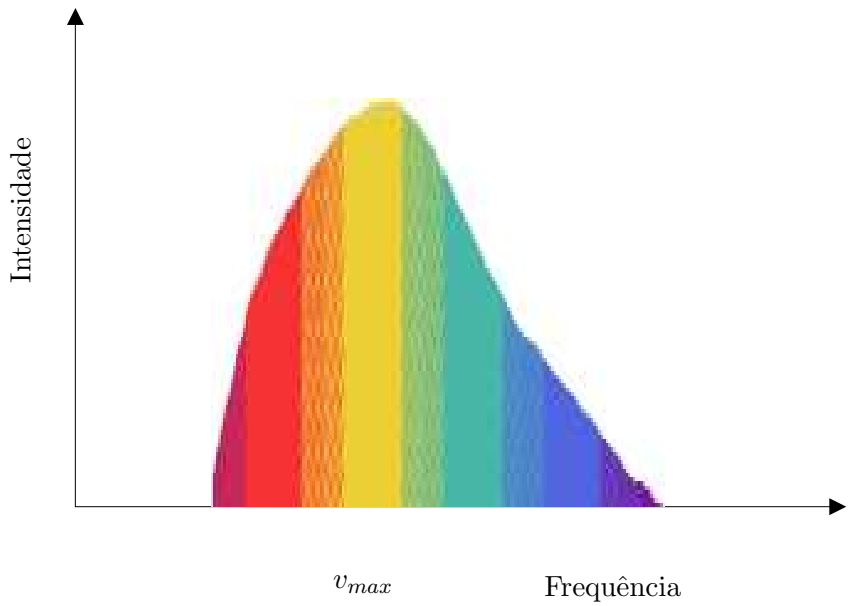


Figura 3.7: Cor de espectro contínuo.

Dimensões das cores

A *Luminância* é a integral da intensidade luminosa de um raio, e serve de indicação da intensidade de iluminação.
A *Crominância* corresponde à frequência dominante v_d, é ela que determina a cor, e normalmente é diferente da frequência v_{max}, a máxima do espectro. Ela pode inclusive ser uma frequência não presente no espectro, pois a mistura de duas cores discretas monocromáticas, por exemplo, pode resultar em uma intermediária.
A *Saturação*, ou "grau de pureza" da cor corresponde à quantidade de cor branca ou preta misturada na cor monocromática para a obtenção de uma cor de espectro contínuo. É definida como:

$$S = \frac{L(v_d)}{L(v_d) + L(w)} \tag{3.1}$$

em que $L(v_d)$ é a luminância da cor dominante e $L(w)$ é a luminância da cor branca.

De acordo com o postulado da tri variância visua, estas 3 dimensões são suficientes para definir uma cor.

Sínteses aditiva e subtrativa das cores

É o modo como é feita a mistura das cores. Um monitor de computador, ou uma televisão, usam a síntese aditiva com 3 cores: a vemelha, verde e azul. Ela funciona misturando raios de várias intensidades de cada uma dessas 3 cores para gerar as outras, e uma mistura total delas gera uma cor branca. Um pintura, por outro lado, usa a síntese subtrativa. Como a pintura não possui luz própria, ela cria cores "roubando" as frequências existentes em um raio refletor de luz. Por exemplo, quanto mais de uma certa tinta que subtrai a cor azul é colocada, menos azul a pintura fica. Uma mistura total das cores neste caso gera uma cor preta.

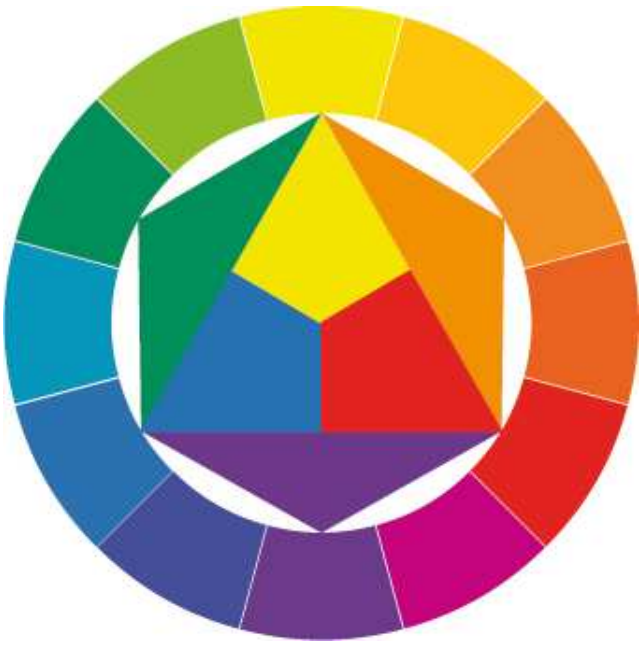


Figura 3.8: Círculo das cores de Itten. No triângulo central, as 3 cores subtrativas primárias,os 3 triângulos adjacentes, as cores subtrativas secundárias, e no círculo em volta, estas 6 cores com as outras 6 cores intermediárias entre outras duas, as chamadas terciárias.

A escolha será dada pela síntese subtrativa de cores, por serem mais intuitivas as misturas entre as cores devido ao fato de que as pessoas geralmente são acostumadas com ela desde que são crianças, pois é assim que a mistura de tintas funciona.

Diminuição das dimensões do espaço das cores

Para simplificar a representação das cores, serão usadas a todo momento apenas cores vivas, ou seja, de saturação máxima. Isto não só diminui o número de dimensões necessárias para as cores (pois agora a saturação é constante) como também faz sentido pois é melhor de se trabalhar com cores saturadas em um computador.

Divisão das crominâncias

Em geral, as pessoas envolvidas no meio científico não possuem nenhum treinamento especial para a sensibilidade das cores, portanto a crominância foi subdividida em apenas 12 cores.

3.3 Teoria de Itten

Como o olho humano é bem limitado em relação a banda de frequências que consegue enxergar, sendo esta menor do que uma oitava (diferentemente do ouvido, por exemplo), fica difícil uma formalização concreta e definitiva das cores, e por isso a teoria de Johannes Itten foi a escolhida.

Diferentemente de outras teorias, como a de Kandinsky, que para uma análise mais artística das cores requer muita sensibilidade de quem a estuda para entender as nuances que ele descreve (como secas, lisas, doces, etc...), e ainda cria muita subjetividade em cima da visualização, a teoria de Itten é mais simples. E uma teoria mais simples torna mais fácil esta classificação formal de cores e como usá-las. A teoria é baseada em 7 tipos de contrastes:

3.3.1 Tipos de contrastes

Contraste de crominância

Contraste entre as cores primárias. Será usada aqui pois proporciona uma distinção com bastante clareza, principalmente em nossa escolha de 12 cores discretas.

Contraste de luminância

Contraste entre claro e escuro. Para se visualizar características locais (qualitativas) de uma imagem, uma luminância que cresce continuamente é mais desejada. Por outro lado, uma luminância descontínua daria uma melhor visualização das características globais por causar variações bruscas. A escolha neste caso então é a de uma luminância contínua, para detectar padrões locais, pois a variação discreta já escolhida para a crominância já permitirá a visualização destas características globais.

Contraste de quente e frio

Contraste entre cores que dão sensação de frio (como tons azulados) ou calor (como tons quentes). É usada para se distinguir valores negativos de positivos.

Contraste de cores complementares

Contraste entre cores que, quando somadas, dão origem a uma cor branca. Estas cores são chamadas complementares entre si, e podem ser usadas para aumentar a luminosidade das outras.

Contraste de simultaneidade

A lei do contraste de simultaneidade diz que, para uma certa cor, o olho exige que a cor complementar da mesma esteja no campo de visão, pois se não encontrada, ele mesmo a produz. O contraste de simultaneidade faz as cores "perderem suas características objetivas", de acordo com Itten, e o contraste de claro e escuro acaba com este contraste. Por estes motivos, este tipo de contraste será evitado com o intuito de criar uma visualização bem objetiva.

Contraste de saturação

Uma cor, ao ser clareada com uma branco fica mais fria, enquanto que misturada com preto fica mais quente, e ainda, se misturada com sua complementar, fica mais amena. Uma cor mais ou menos saturada (menos ou mais pura) pode parecer mais ou menos forte perto de outra cor, mas isto é muito relativo (depende demais de quais cores está se analisando). Por isto, não será usado este contraste.

Contraste de quantidade

Indica o contraste entre duas ou mais cores relativa a quantidade de cada cor presente. Uma cor mais presente que outra acaba por criar uma impressão de ser mais ou menos forte, e para evitar isto, a luminância de cada cor será calibrada adequadamente.

3.3.2 Algebrização das cores

Farge organiza as cores do seguinte modo:

- *Crominâncias subtrativas primárias (ordem 1)*: são as cores vermelho, amarelo e azul, representadas, respectivamente, pelos valores inteiros ímpares $v_1 = 1, 3$ e 5.

- *Crominâncias subtrativas secundárias (ordem 2)*: são as cores laranja, verde e violeta, representadas, respectivamente, pelos valores inteiros pares $v_2 = 2, 4$ e 6.
- *Crominâncias subtrativas terciárias (ordem 3)*: são as cores intermediárias entre duas cores c_1 e c_2, de ordens 1 e 2, como vermelho-violeta, etc... . São representadas numericamente pelos valores $v_3 = \frac{v_1+v_2}{2}$.
- *Cores de ordem mais alta*: seriam as cores intermediárias entre estes valores, como por exemplo, entre um cor de ordem 3 e 2. Não serão usadas.

3.4 Escolha das palhetas

A pergunta feita por Farge é, *"como escolher as palhetas de cores para visualizar J campos escalares, caracterizados cada um por um conjunto de I crominâncias em equilíbrio harmonioso (sem contrastes de saturação) umas contra as outras?"*.

Cada palheta deve identificar um campo usando um conjunto de I crominâncias que lhe são próprias, sendo que ela deve se distinguir o máximo possível das outras $J-1$ combinações de I crominâncias presentes nos outros $J-1$ campos. Com geometrização das cores definidas a partir da classificação das crominâncias diametralmente opostas, a regra de Itten para a escolha das palhetas pode ser expressa assim:

I palhetas estão em equilíbrio harmonioso se se eles correspondem aos I pontos de um polígono inscrito dentro do círculo de cores (Figura 3.8). Como foi definida a álgebra simples do espaço de cores, esta regra pode ser transformada em um algoritmo. Assim, o valor da cor de número i da palheta de número j em equilíbrio complementar com as $I-1$ outras crominâncias presentes é encontrada pela fórmula:

$$v(i,j) = \left(v(1,1) + 6\left(\frac{i-1}{I}\right) + 6\left(\frac{j-1}{JI}\right)\right) \mod 6 \tag{3.2}$$

com i variando de 1 a I, j de 1 a J, e $v(1,1)$ o valor da crominância inicial.

Assim, se deseja duas palhetas diferentes (para representar o módulo e a fase de um escalograma, por exemplo) com 4 crominâncias cada, e supondo $v(1,1) = 1$, a matriz conseguida é:

$$v(i,j) = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} \tag{3.3}$$

que corresponde a uma matriz de cores:

$$c(i,j) = \begin{bmatrix} \text{vermelho} & \text{amarelo} & \text{azul} \\ \text{laranja} & \text{verde} & \text{violeta} \end{bmatrix} \tag{3.4}$$

Foram escritos algoritmos para a criação desta matrix de cores de Farge, para sua conversão para valores aditivos de RGB (para que o computador possa reproduzir), e para a criação de colormaps do GNU/Octave a partir desses valores RGB. Todos os códigos estão disponíveis no apêndice C.3, na página 158.

3.4.1 Exemplo

Serão feitas agora imagens com a palheta *Hot* e palhetas geradas por estes algoritmos, para a comparação. Como desejo demonstrar 2 gráficos, um para o valor absoluto e outro para a fase, usarei $J = 2$, e para o número de crominâncias escolhi $I = 5$ cores em cada palheta. Além disso, o valor escolhido para a cor inicial da primeira palheta foi o azul, que na álgebra de cores equivale ao valor $v(1,1) = 8$. A matriz de valores que o algoritmo gerou foi:

$$v(i,j) = \begin{bmatrix} 2.0 & 3.2 & 4.4 & 5.6 & 0.8 \\ 2.6 & 3.8 & 5.0 & 0.2 & 1.4 \end{bmatrix}$$

cuja matriz de cores correspondente é:

$$c(i,j) = \begin{bmatrix} \text{laranja} & \text{amarelo-verde} & \text{verde-azul} & \text{azul-violeta} & \text{violeta-vermelho} \\ \text{laranja-amarelo} & \text{amarelo-verde} & \text{azul} & \text{violeta-vermelho} & \text{vermelho laranja} \end{bmatrix}$$

Na Figura 3.9 foi usada a a palheta *hot* e a primeira palheta gerada com esses valores para o valor absoluto da transformada, e na Figura 3.10 foi usada novamente a palheta *Hot* com a segunda palheta gerada pelo algoritmo para plotar a fase.

Pode se reparar que, enquanto a palheta *Hot* possui um espectro de cores que faz com que a iluminação média entre as cores aditivas cresce continuamente para aumentar sua força nos valores mais altos, as palhetas de Farge mantêm suas iluminações médias quase constantes, eliminando assim o contraste de saturação, e usando a diferença cromática para demonstrar os formatos da imagem.

Repare, entretanto, que a palheta gerada pelo algoritmo na Figura 3.10 tem um problema, que também está presente na palheta *Hot*. Ao se plotar o gráfico de fases, deve ser levado em consideração que os extremos do gráfico ($-\pi$ e $+\pi$) valem ambos o mesmo valor, logo deveriam ser representados pela mesma cor. Para contornar isto, a palheta escolhida pode ser modificada para que seus extremos tenha mesmo valor de cor:

$$v(i,j) = \begin{bmatrix} 2.0 & 3.2 & 4.4 & 5.6 & 0.8 \\ 2.6 & 3.8 & 5.0 & 0.2 & 2.6 \end{bmatrix}$$

cuja matriz de cores correspondente é:

$$c(i,j) = \begin{bmatrix} \text{laranja} & \text{amarelo-verde} & \text{verde-azul} & \text{azul-violeta} & \text{violeta-vermelho} \\ \text{laranja-amarelo} & \text{amarelo-verde} & \text{azul} & \text{violeta-vermelho} & \text{laranja-amarelo} \end{bmatrix}$$

o que corrige o problema. A demonstração disto está na Figura 3.11.

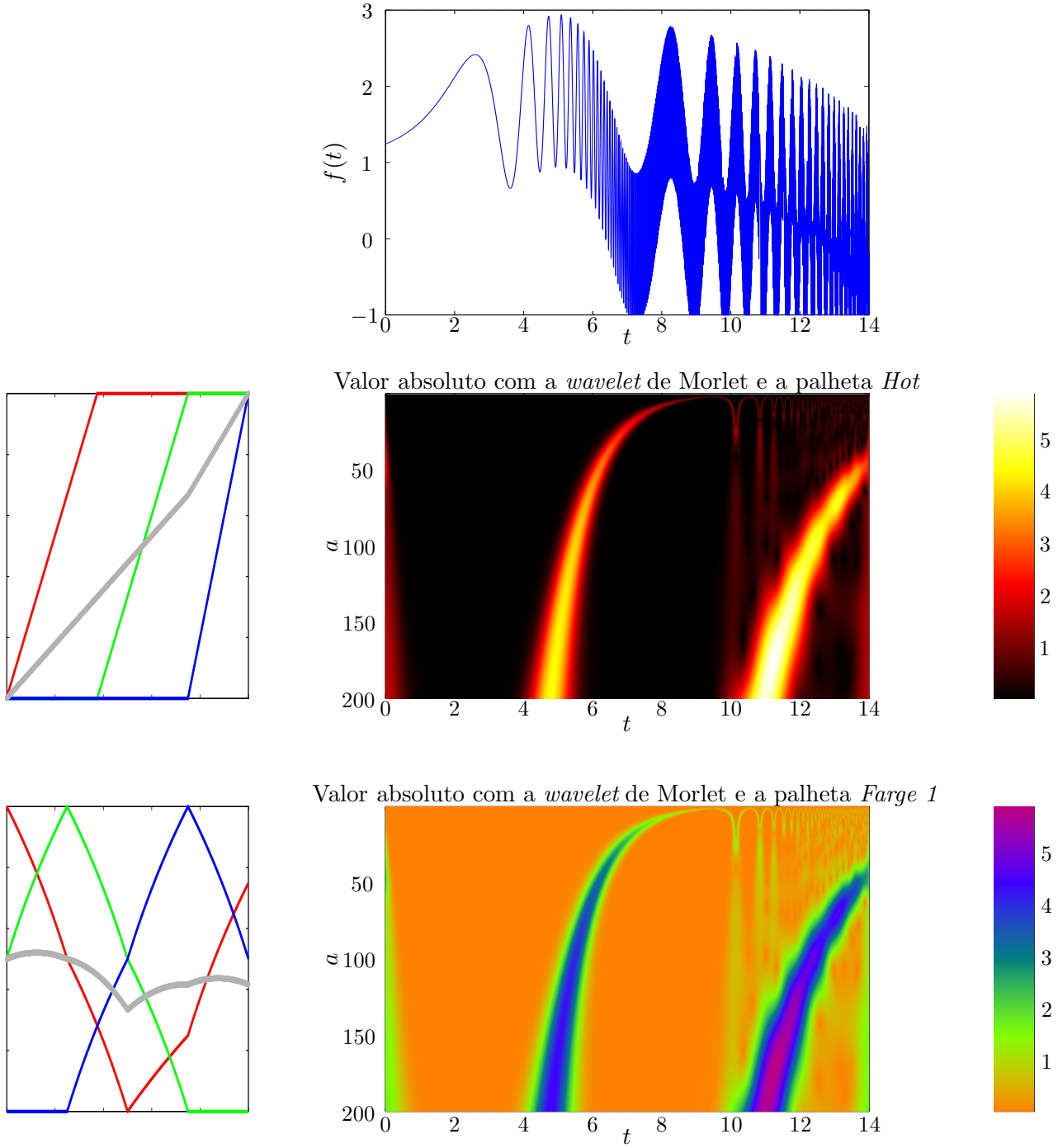


Figura 3.9: Escalograma da soma de chirps hiperbólicos com a *wavelet* de Morlet, comparando a colormap *Hot* com a gerada pelo algoritmo de Farge.

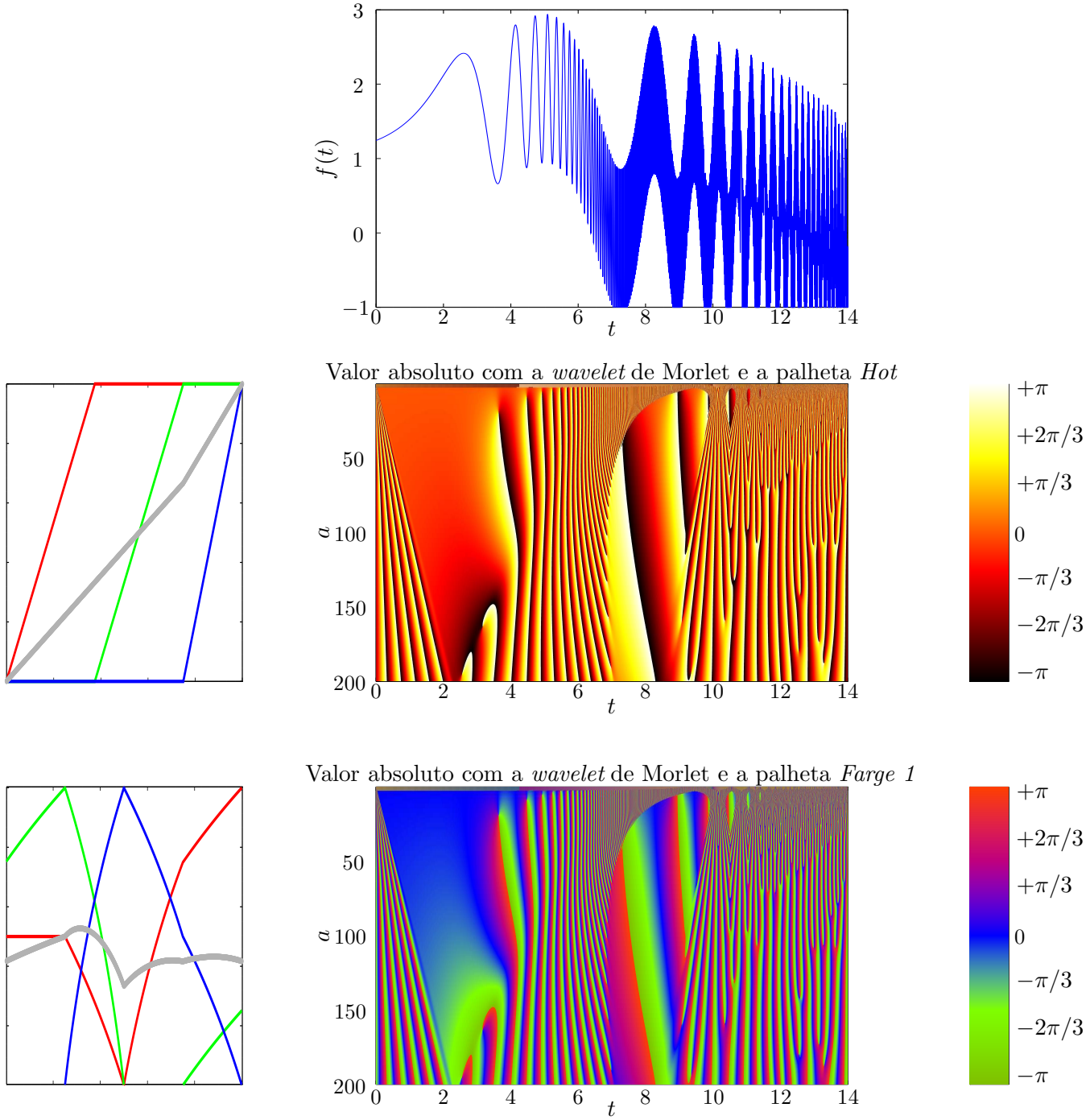


Figura 3.10: Fase da transformada da soma de chirps hiperbólicos com a *wavelet* de Morlet, comparando a colormap *Hot* com a gerada pelo algoritmo de Farge.

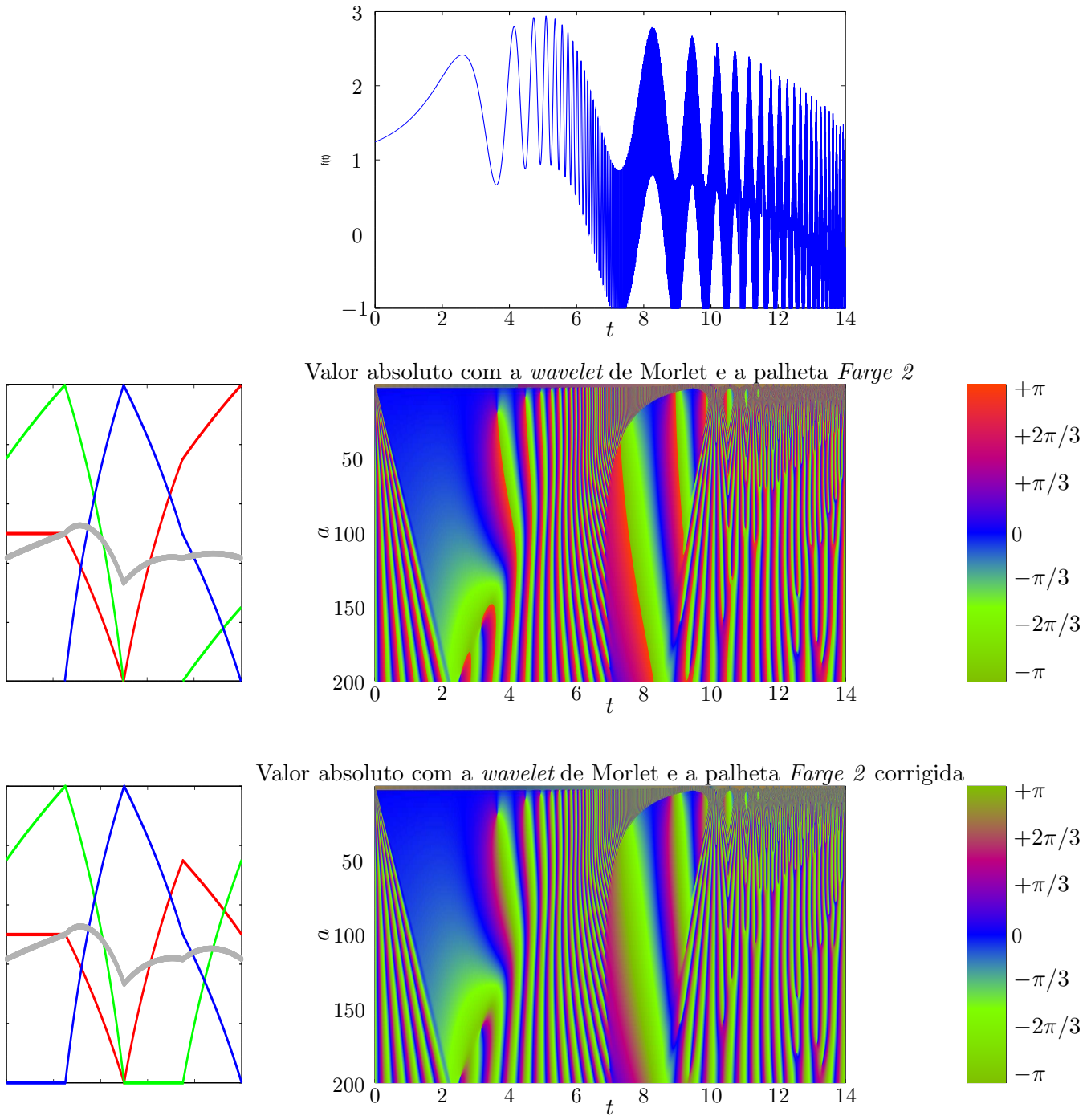


Figura 3.11: Comparação da segunda palheta gerada com sua correção.

3.5 Deficiência visual de cor (daltonismo)

Uma grande parcela da população mundial sofre com daltonismo, o que as impede de distinguir entre certas cores. Algumas possuem total deficiência cromática e enxergam apenas em tons de uma só cor (tons de cinza), e estas pessoas devem ser levadas em conta também para a criação de palhetas de cor. Para tais pessoas com visão totalmente monocromática, a única saída é o uso de uma palheta que tenha iluminação constante (como a palheta *hot* usada).

Nesta seção, serão discutidos casos de deficiência não totalmente monocromática, e será proposta uma palheta especial para ajudar no caso do tipo mais comum de daltonismo.

3.5.1 Tipos de daltonismo

Não existem exatamente "níveis" de daltonismo, mas existem tipos. Os tipos são:

Monocromatismo

- Total indistinção entre crominâncias, percebe apenas tons de cinza.

Protanopia

- Ausência de cones de ondas **longas**.
- Deficiência das cores **verde**-**amarelo**-**vermelho**.
- Cerca de 1% de homens e 0.01% de mulheres.

Deuteranopia

- Ausência de cones de ondas **intermediárias**.
- Deficiência das cores **verde**-**amarelo**-**vermelho**.
- Cerca de 6% de homens e 0.4% de mulheres.

Tritanopia

- Ausência de cones de ondas **curtas**.
- Deficiência das cores **amarelo**-**azul**.
- Raro, atinge cerca de 0.001 a 0.02% das pessoas, independente do sexo.

Como foi dito, os daltonismos do tipo deuteranopia e protanopia possuem efeitos muito parecidos e são bem mais comuns que os outros (principalmente em homens), e portanto eles serão estudados.

Comparação das palhetas Nesta seção, as palhetas usadas até agora serão analisadas de acordo com seus desempenhos para visualização por pessoas com daltonismo. Será dada prioridade, neste primeiro momento, para as síndromes de deuteranopia e protanopia, porém imagens de simulação para tritanopia também serão demonstradas para simples comparação. As simulações de daltonismo foram feitas usando o software ImageJ com o plugin Vischeck.

Na Figura 3.12, pode ser observada a dificuldade total com que ela apresenta as estruturas presentes no sinal para pessoas com deficiência. Não só ela usa as cores mais problemáticas (tons de vermelho) para expor metade dos seus valores, como aqui a característica dela de não possuir luminosidade crescente também atrapalha.

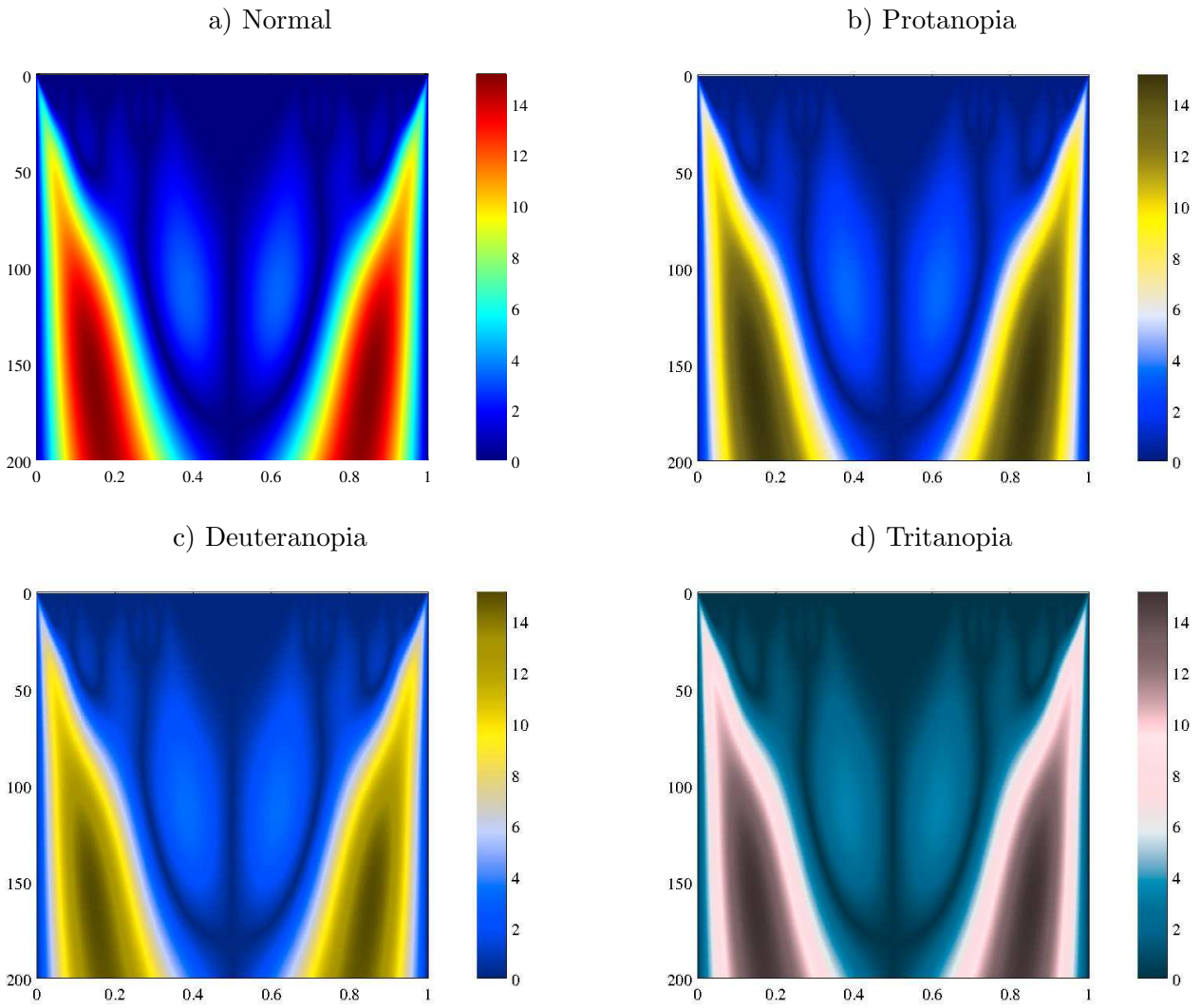


Figura 3.12: Simulação da percepção da palheta *Jet* por pessoas com daltonismo.

Nas Figuras 3.13 e 3.14, a palheta *Hot* demonstra também uma grande dificuldade em expor seus dados. Quem possui problemas de verde-amarelo-vermelho enxerga todo o espectro de cores da imagem como um espectro amarelo monocromático.

Na Figura 3.15, já pode ser visto que a palheta *Blue* é uma boa opção para substituir a palheta *Hot* (pelo menos nos casos mais comuns), já que ela evita as cores quentes problemáticas das palhetas *Jet* e *Hot*.

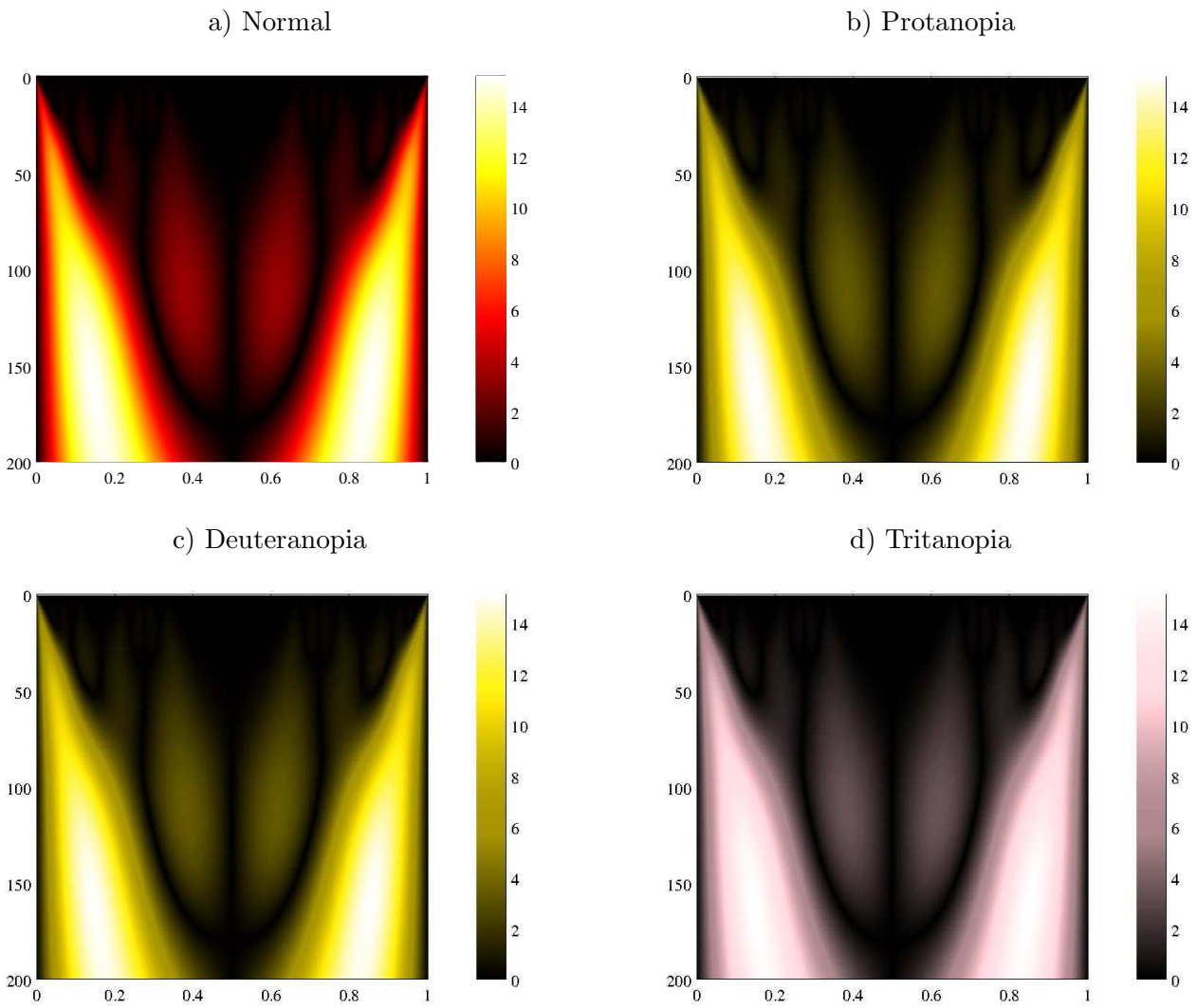


Figura 3.13: Simulação da percepção da palheta *Hot* por pessoas com daltonismo.

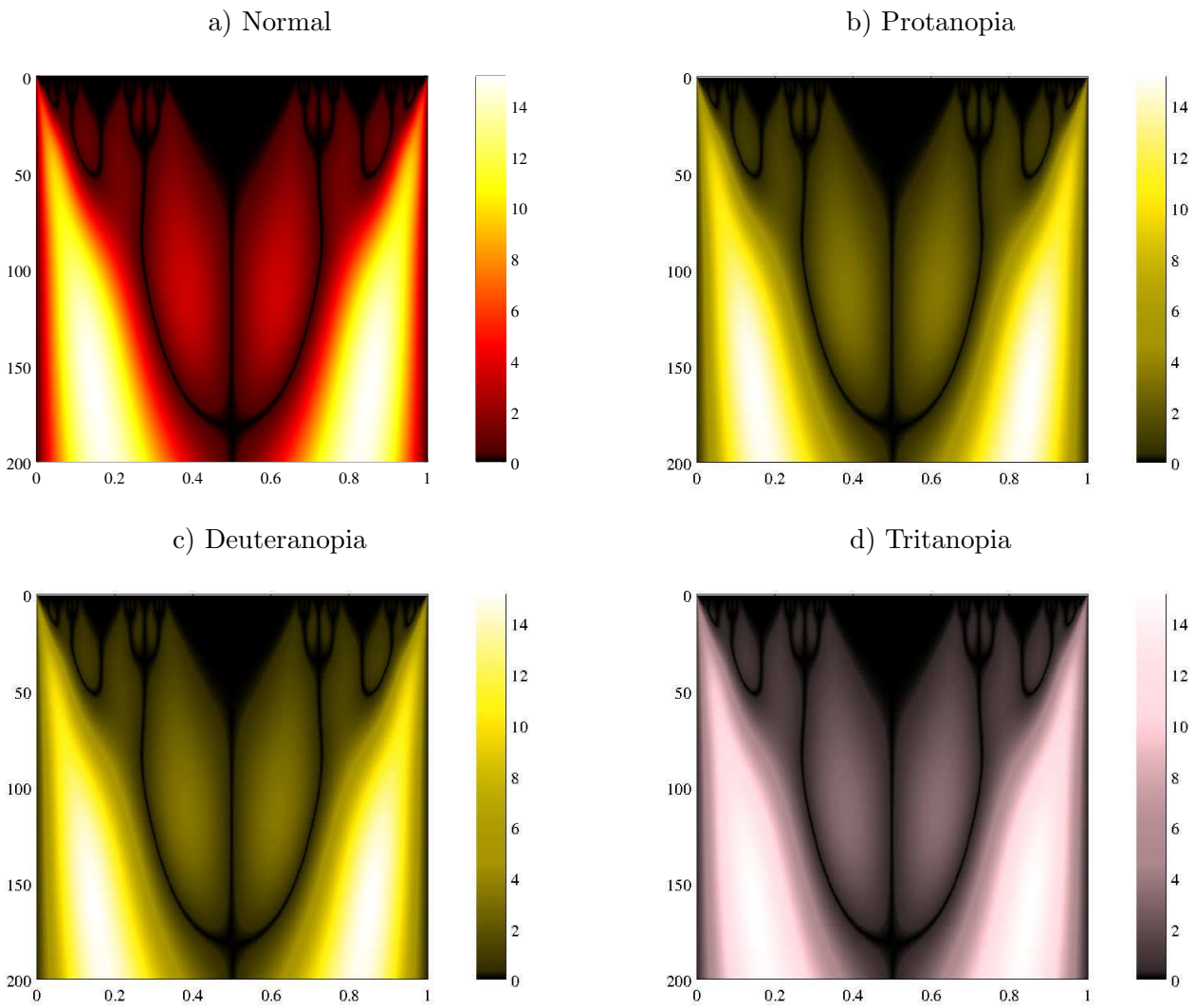


Figura 3.14: Simulação da percepção da palheta *Hot* modificada por pessoas com daltonismo.

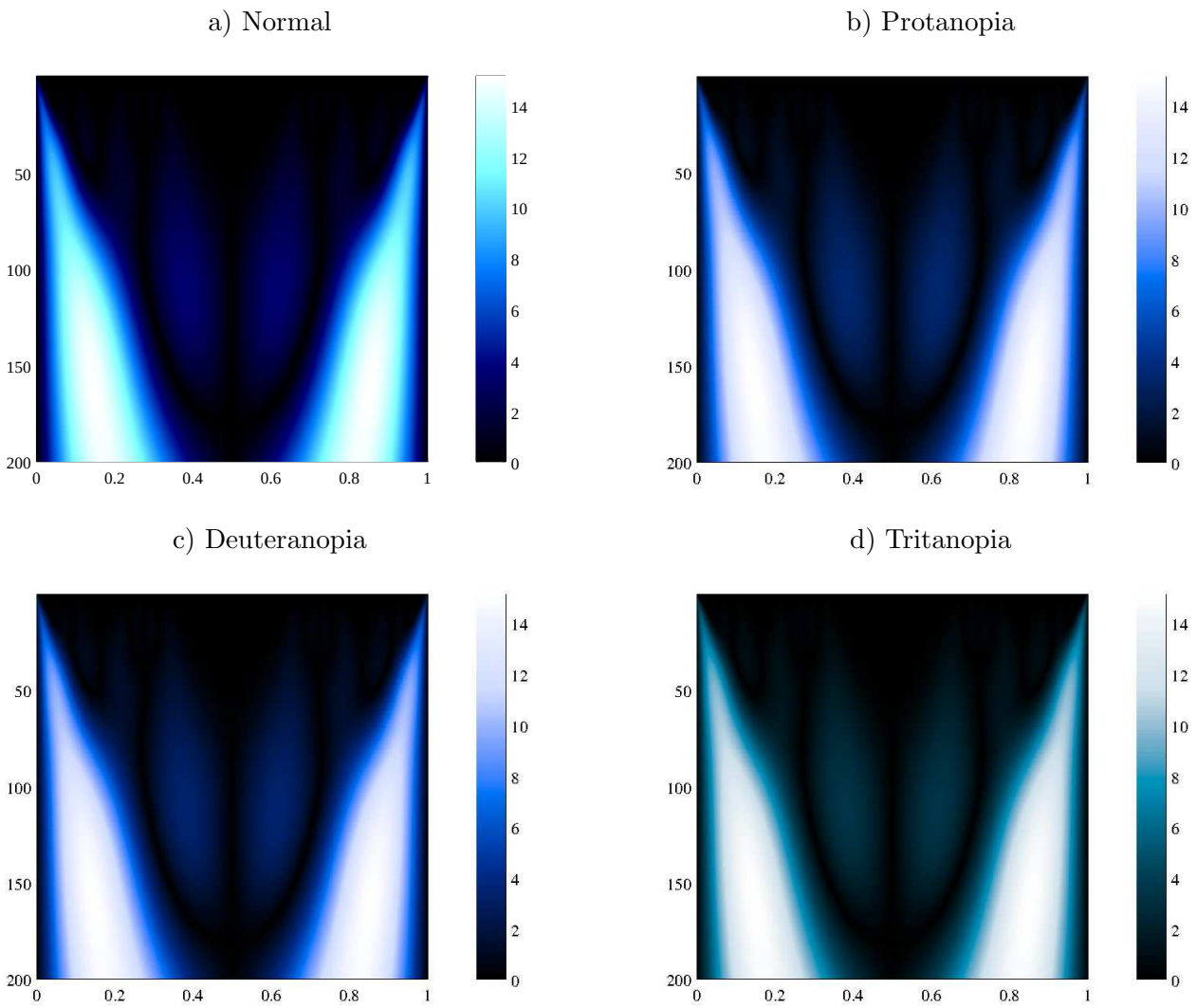


Figura 3.15: Simulação da percepção da palheta *Blue* por pessoas com daltonismo.

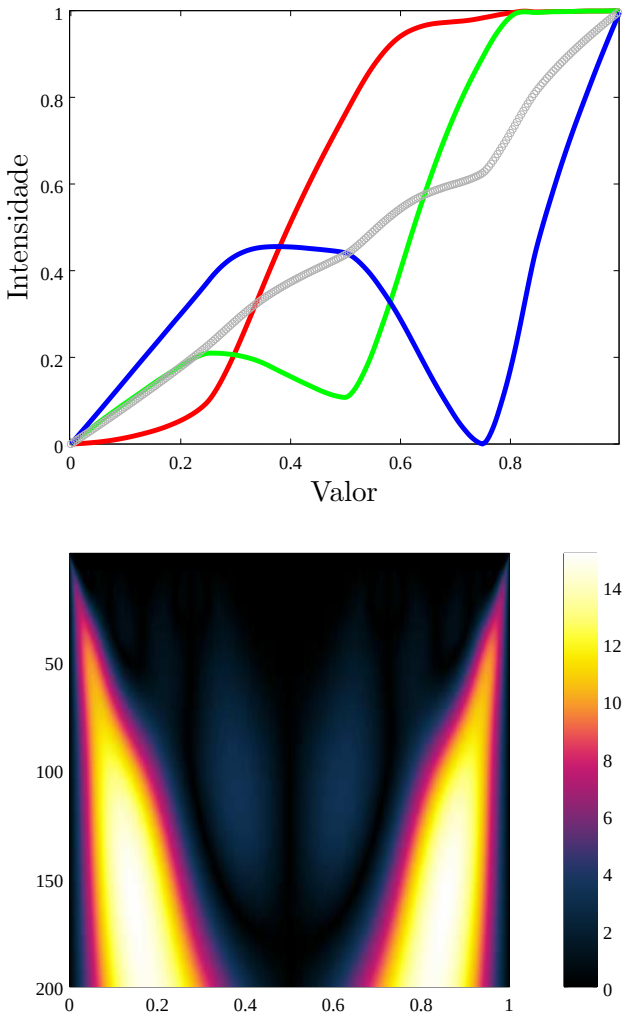


Figura 3.16: Palheta Morgenstemning e um exemplo de seu uso.

Uma palheta interessante para amenizar este problema é a palheta *Morgenstemning*, proposta em [7]. Ela está ilustrada na Figura 3.16. Ela foi feita com os seguintes princípios:

- Luminosidade crescente e linear, para que a iluminação de uma boa ideia da intensidade da imagem;
- Cores azuladas, para evitar o uso de cores vermelhas que poderiam inviabilizar a visualização para portadores de protanopia e deuteranopia;
- Alguns tons especiais de violeta, para que quem possui sua visão normal consiga tirar um proveito ainda maior da palheta.

Seu melhor desempenho para quem possui deuteranopia e protanopia pode ser visualizado na simulação da Figura 3.17.

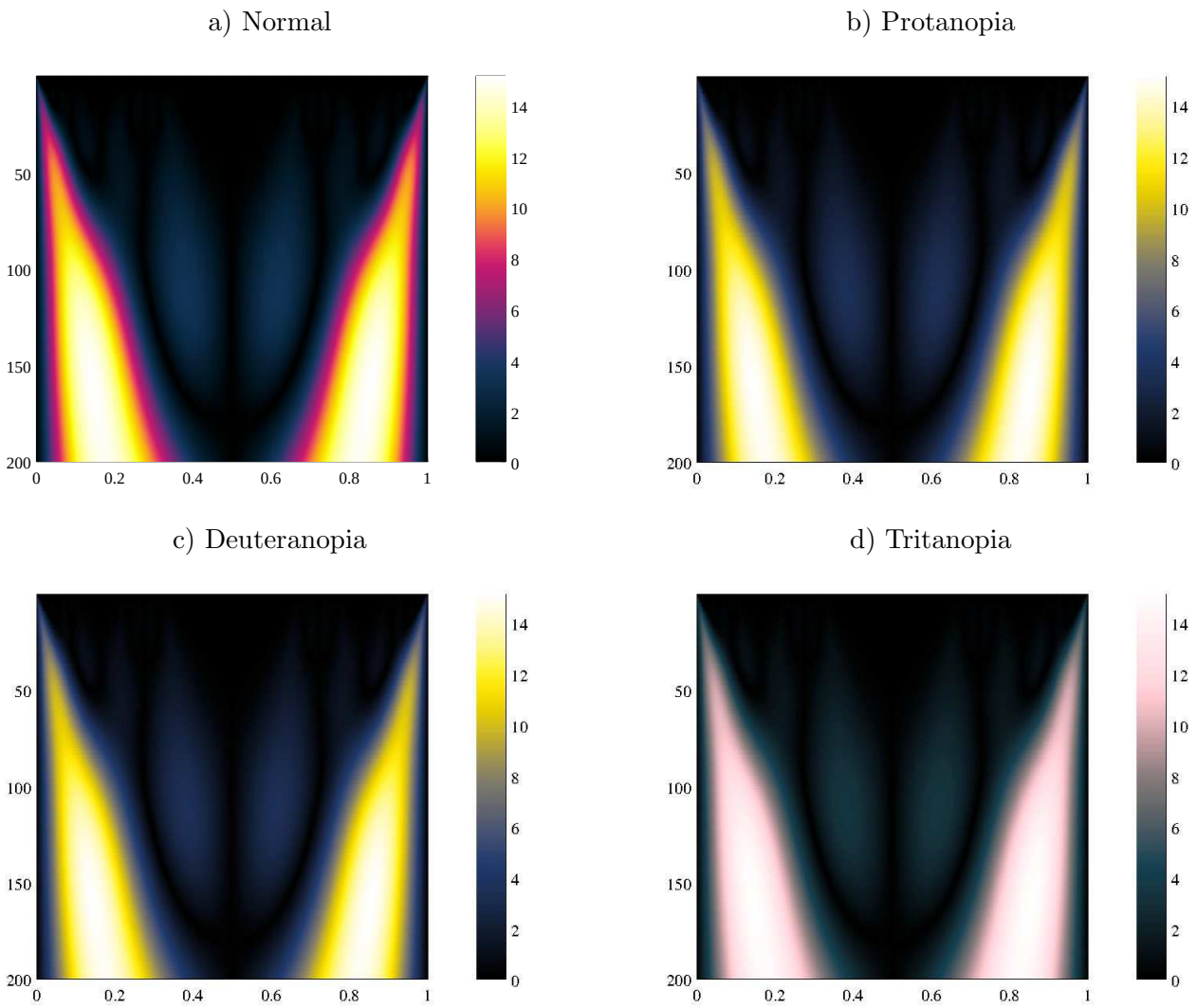


Figura 3.17: Simulação da percepção da palheta *Morgenstemning* por pessoas com daltonismo.

Capítulo 4

Sinais Analíticos

Suponha um sinal $f(t)$ que possui transformada de *Fourier* $\hat{f}(\xi)$. Se sua transformada for nula para todo $\xi < 0$, este sinal é chamado de "sinal analítico". A ideia por trás de se tratar destes sinais em especial, é que a transformada de *Fourier* de uma função real sempre será simétrica, fazendo com que as componentes negativas de frequência sejam redundantes. Desta maneira, a parte negativa da transformada pode ser descartada sem perda de informação. Para exemplificar, utilizarei o software GNU/Maxima para calcular analiticamente o espectro de *Fourier* da função $f(t) = e^{-t^2}$, como visto na Figura 4.1.

4.1 Exemplo: Transformada da função real

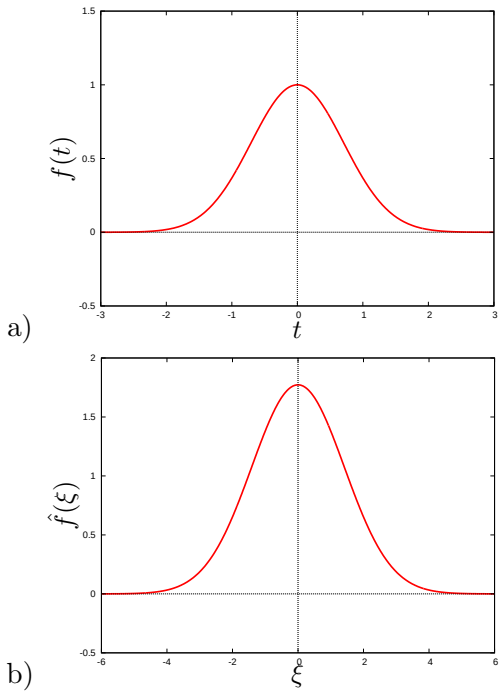


Figura 4.1: Figura a): a função original. Figura b): seu espectro de frequências.

Como pode ser observado nesta imagem, as partes real e imaginária da transformada gerada realmente são simétricas, a parte real sendo par e a parte imaginária sendo ímpar. Isto pode ser demonstrado analiticamente usando a definição da transformada contínua. Como $f(t) \subset \mathbb{R}$, a sua fórmula pode ser

manipulada da seguinte maneira:

$$\hat{f}(\xi) = \int f(t) \cdot e^{-2\pi\imath\xi t} dt$$
$$\hat{f}(\xi) = \int f(t) \cdot [\cos(2\pi\imath\xi t) - \imath \sin(2\pi\imath\xi t)]\, dt$$
$$\hat{f}(\xi) = \int f(t) \cdot \cos(2\pi\imath\xi t) dt - \imath \int f(t) \cdot \cos(2\pi\imath\xi t) dt$$

Em que ambas as integrais são reais, além de que:

$$Re\{\hat{f}(\xi)\} = \int f(t) \cdot \cos(2\pi\imath\xi t) dt \quad \text{é par em } \xi$$
$$Im\{\hat{f}(\xi)\} = -\int f(t) \cdot \sin(2\pi\imath\xi t) dt \quad \text{é ímpar em } \xi$$

4.2 Exemplo: transformada de um sinal analítico

Isto não pode ser demonstrado para sinais complexos, e portanto não é esperado que estes sigam tal comportamento. Será analisada deste vez a transformada de uma função complexa e veremos o comportamento do seu gráfico.

Observa-se na Figura 4.2 que neste caso a transformada de *Fourier* do sinal possui (salvo uma pequena distorção causada pelo cálculo numérico no GNU/Octave) parte negativa nula, porém isso só ocorreu porque o sinal escolhido foi um sinal analítico. Isto não é uma regra para sinais complexos gerais.

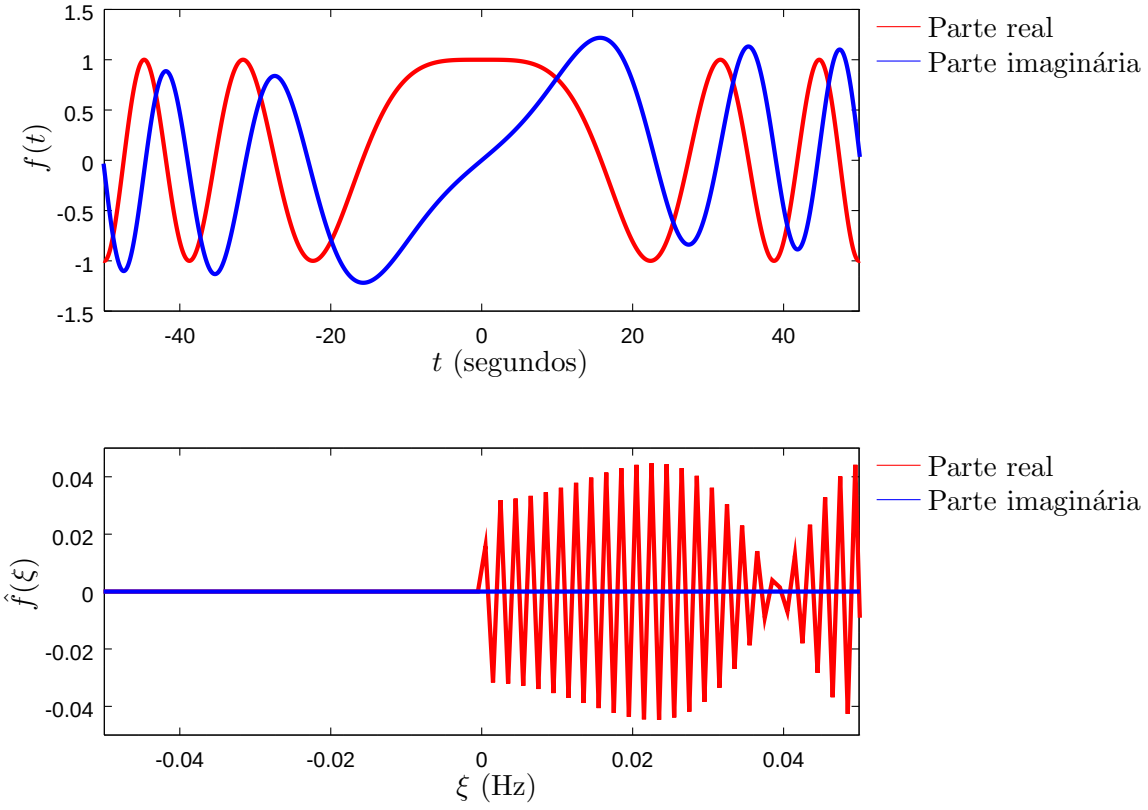


Figura 4.2: Sinal complexo e sua transformada de *Fourier* .

Uma outra utilidade de um sinal analítico, é a extração de sinais em modulação, que também será exemplificada a seguir.

4.3 Exemplo: extração de um sinal modulado

Caso um receptor queira analisar o sinal modulado e extrair o sinal original, ele fazer isto encontrando sua representação analítica (técnica que será demonstrada posteriormente). Para ilustrar, supõe-se um sinal $f(t) = \sin(10 \cdot 2\pi t) + \sin(30 \cdot 2\pi t)$, ou seja, possuidor de duas frequências (10 e 30Hz) e a seguir faz-se a modulação com uma onda portadora de 50Hz. O sinal obtido é então:

$$f_{mod}(t) = [\sin(10 \cdot 2\pi t) + \sin(30 \cdot 2\pi t)] \cos(50 \cdot 2\pi t)$$

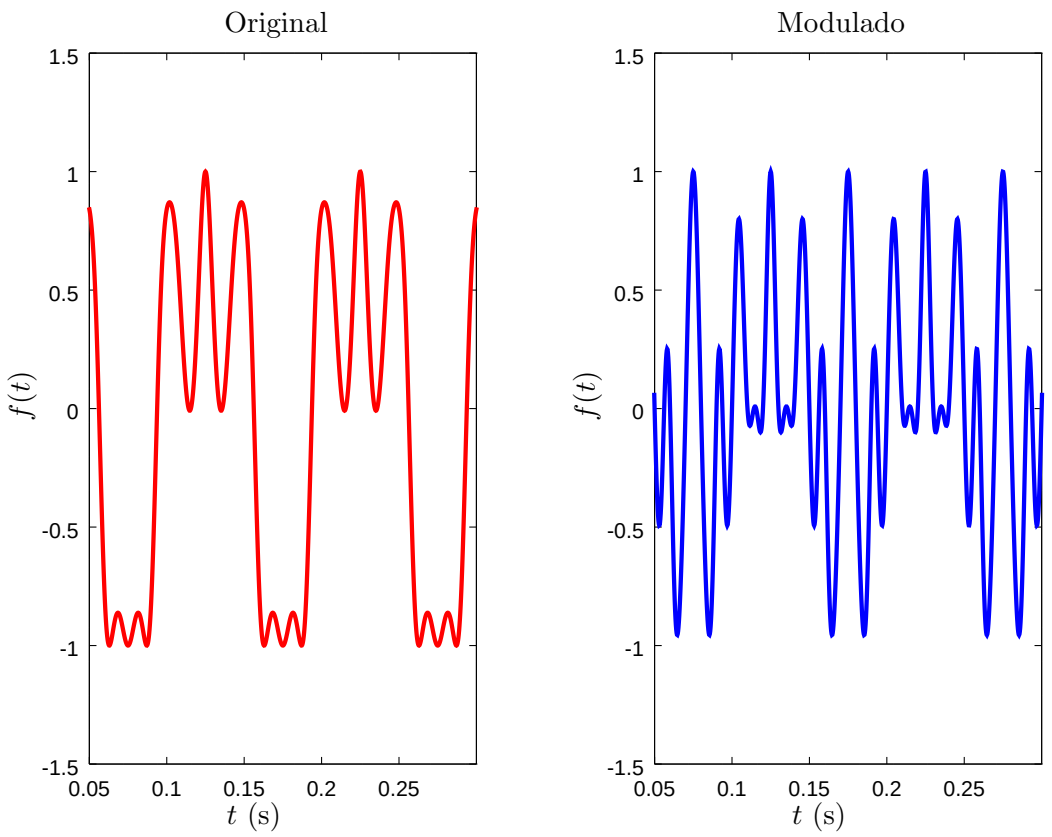


Figura 4.3: Sinais original e do modulado

O receptor, ao receber o sinal modulado, calcula sua representação analítica, e a partir deste ponto, o objetivo é extrair o sinal original do sinal modulado. Calculando o módulo da representação analítica do sinal modulado, com o módulo do sinal original o Octave, observa-se a relação que eles possuem.

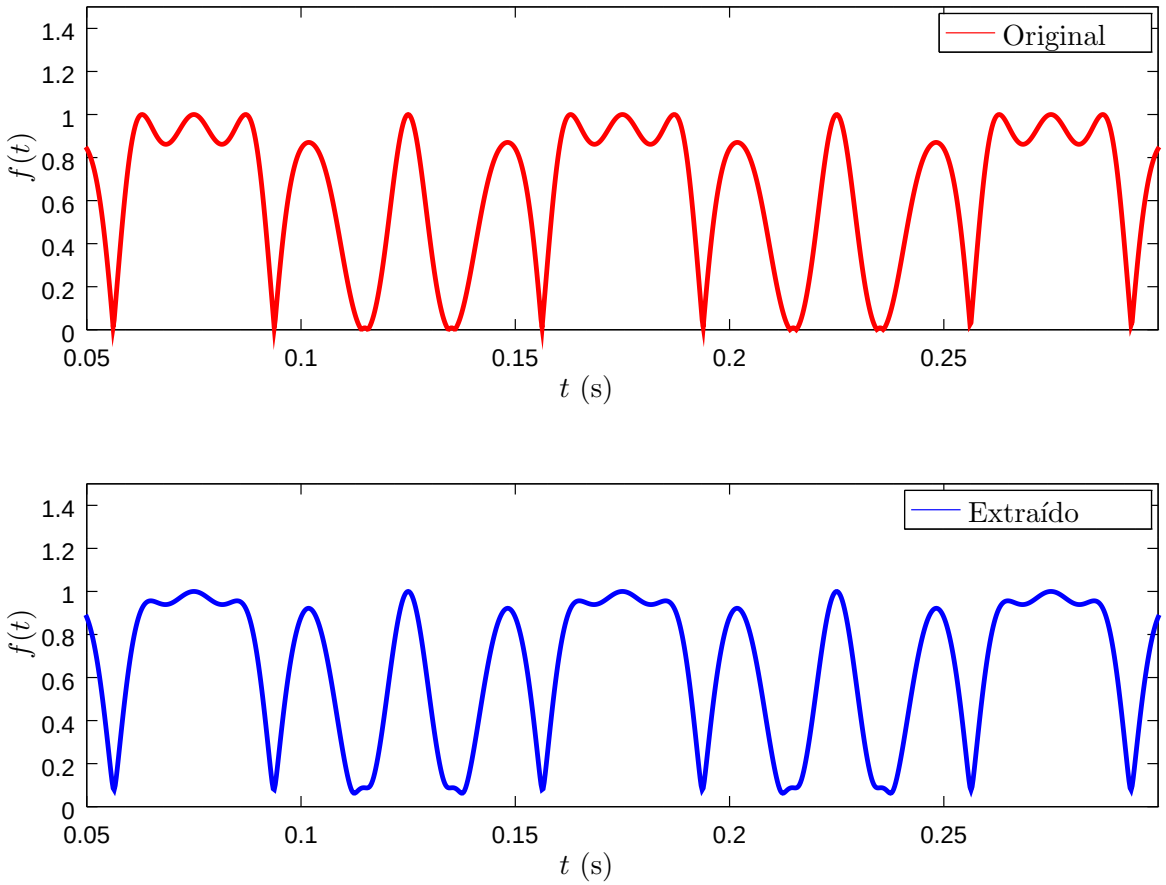


Figura 4.4: Sinal original e extraído do sinal modulado

Apesar das pequenas diferenças percebidas (geradas pelo cálculo numérico no Octave), observa-se que eles são iguais. De fato, a representação analítica do sinal modulado é:

$$f_{mod,a}(t) = [\sin(10 \cdot 2\pi t) + \sin(30 \cdot 2\pi t)] \cdot e^{50 \cdot 2\pi t}$$

Ou seja, a multiplicação do sinal original com um exponencial com a mesma frequência da onda portadora:

$$f_{mod,a}(t) = f(t) \cdot e^{50 \cdot 2\pi t}$$

Decindo-se usar representações analíticas dos sinais ao invés de suas formas originais, não só facilita o trabalho com alguns sinais reais em muito, como ainda torna muitos deles mais significativos, e isto é fácil de se perceber usando como exemplo uma exponencial complexa, que mais fácil de se trabalhar do que com as funções seno ou cosseno correspondentes. Existem várias aplicações para estes tipos de sinais, mas para que possamos verdadeiramente trabalhar com ele, precisamos desenvolver uma técnica para sua obtenção. Esta técnica é a *Transformada de Hilbert*, que será mostrada a seguir.

Capítulo 5

Transformada de *Hilbert*

5.1 Derivação

Como foi explicado anteriormente, toda vez que a transformada de *Fourier* é aplicada à um sinal real, o resultado é uma representação no domínio da frequência que possui energia tanto para frequências negativas quanto positivas, além de que estes dois "lados" da função da transformada sempre possuem algum tipo de simetria (o que torna a informação um tanto quanto redundante). A ideia é encontrar uma representação alternativa do sinal, que não possua energia em suas frequências negativas. O processo é feito do seguinte modo: Supõe-se um sinal $f(t)$ que possui transformada de *Fourier* $\hat{f}(\xi)$. Este sinal, chamado de $f_a(t)$ possui transformada definida como:

$$\hat{f}_a(\xi) = \begin{cases} 0 & \text{se } \xi < 0 \\ \hat{f}(\xi) & \text{se } \xi = 0 \\ 2\hat{f}(\xi) & \text{se } \xi > 0 \end{cases} \tag{5.1}$$

Notando que o motivo de tal definição é que a parte positiva compense a energia perdida na parte negativa, levando a um sinal com a mesma energia. Utilizando-se de uma função degrau, ela pode ser re-escrita como:

$$\hat{f}_a(\xi) = 2\hat{f}(\xi) \cdot u(\xi)$$

A partir dai, a função procurada $f_a(t)$ pode ser encontrada simplesmente aplicando a transformada inversa nesta equação.

$$\begin{aligned} \hat{f}_a(\xi) &= 2\hat{f}(\xi) \cdot u(\xi) \\ f_a(t) &= \mathcal{F}\{2\hat{f}(\xi) \cdot u(\xi)\} \\ f_a(t) &= 2\mathcal{F}\{\hat{f}(\xi)\} * \mathcal{F}\{u(\xi)\} \end{aligned}$$

Como sabe-se que $f(t) = \mathcal{F}^{-1}\{\hat{f}(\xi)\}$ e $\mathcal{F}^{-1}\{u(\xi)\} = \frac{\delta(t)}{2} + \frac{\imath}{2\pi t}$:

$$\begin{aligned} f_a(t) &= 2f(t) * \left(\frac{\delta(t)}{2} + \frac{\imath}{2\pi t}\right) \\ f_a(t) &= f(t) * \left(\delta(t) + \frac{\imath}{\pi t}\right) \end{aligned}$$

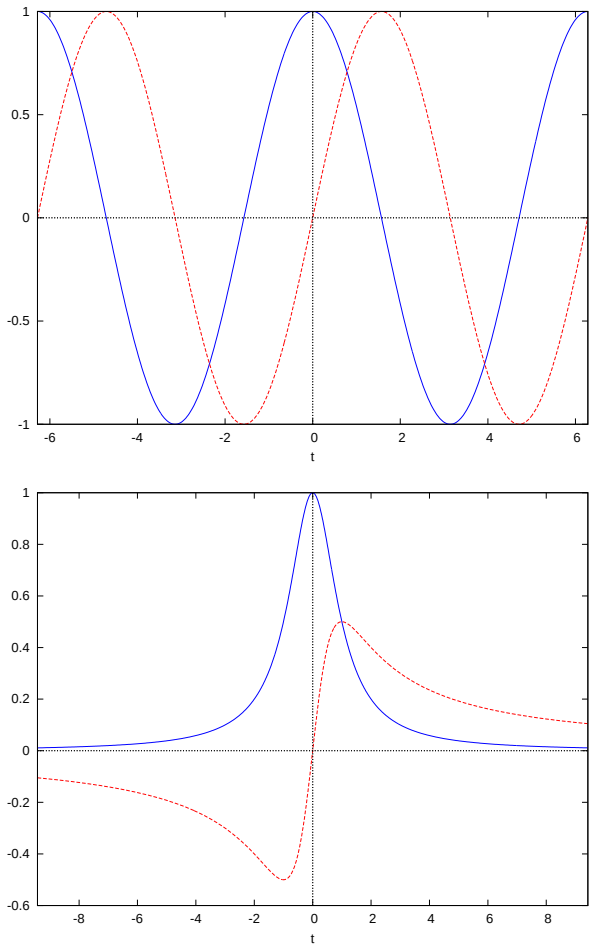


Figura 5.1: Transformadas de *Hilbert* das funções $\cos(t)$ e $\frac{1}{t^2+1}$ (função original em azul e transformada em vermelho)

E então:

$$f_a(t) = f(t) + \imath f(t) * \left(\frac{1}{\pi t}\right) \tag{5.2}$$

Portanto, o sinal que satisfaz a propriedade procurada é o próprio sinal original, somado com uma parte imaginária. Esta fórmula na parte imaginária é conhecida como transformada de *Hilbert* e será denotada por $\mathcal{H}\{f(t)\}$ (note que esta transformada gera uma função no mesmo domínio, já que é simplesmente uma convolução), e o sinal encontrado desta forma é a própria representação analítica do sinal $f(t)$.

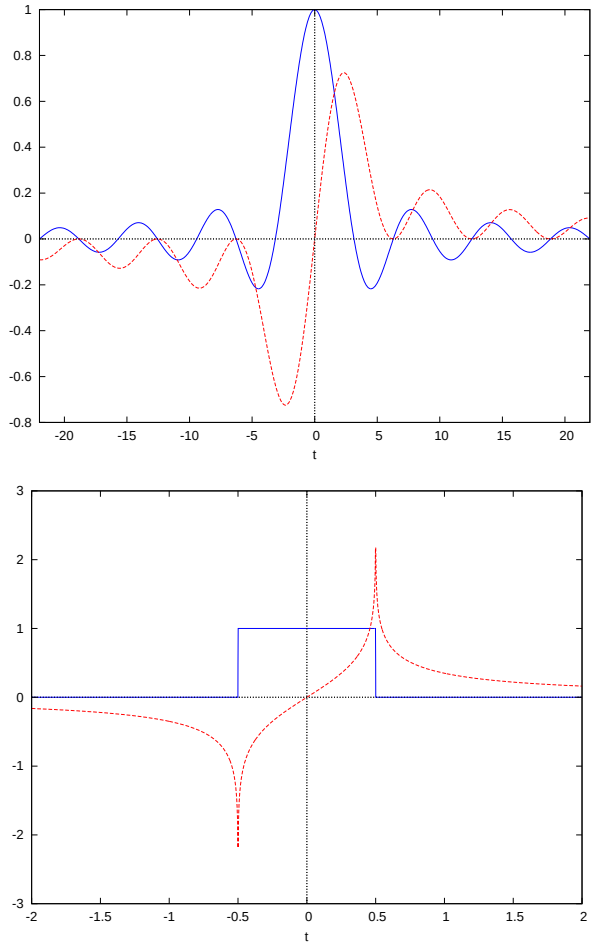


Figura 5.2: Transformadas de *Hilbert* das funções $\frac{\sin(t)}{t}$ e $\mathrm{rect}(t)$ (função original em azul e transformada em vermelho)

A transformada de *Hilbert* também pode ser definida como um filtro $h(t)$ que possui, no domínio da frequência, a expressão:

$$\hat{h}(\xi) = \begin{cases} j & \text{if } \xi < 0 \\ 0 & \text{if } \xi = 0 \\ -j & \text{if } \xi > 0 \end{cases} \tag{5.3}$$

Este filtro ideal de *Hilbert* atua gerando um atraso de fase de $-\pi/2$ radianos (ou $-90°$) em todas as frequências do sinal. O sinal analítico pode ser representado da seguinte forma:

$$f_a(t) = A(t)e^{\imath\phi(t)} \tag{5.4}$$

Onde a grandeza $A(t)$ representa a amplitude instantânea (ou envelope), e $\phi(t)$ a fase instantânea no domínio do tempo:

$$\phi(t) = \text{tg}^{-1}\left(\frac{Im\{f_a(t)\}}{Re\{f_a(t)\}}\right)$$

em que é possível também definir a frequência instantânea como:

$$\xi(t) = \frac{1}{2\pi}\frac{d\phi(t)}{dt} \tag{5.5}$$

Além disso, uma propriedade importante e extremamente útil desta transformada é a seguinte:

Teorema 5.1. *Supondo-se que existam dois sinais $f_l(t)$ de baixa frequência e $f_h(t)$ de alta frequência, sendo que os espectros de tais sinais não se cruzam. Se isto ocorrer, temos a seguinte propriedade:*

$$\mathcal{H}\{f_l(t) \cdot f_h(t)\} = f_l(t) \cdot \mathcal{H}\{f_h(t)\} \tag{5.6}$$

Usando esta propriedade, pode-se separar sinais modulados de suas ondas portadoras. Supondo que a função $f_h(t)$ é uma única alta frequência, a função analítica do sinal completo terá frequência igual a frequência desta onda.

5.1.1 Exemplo: Modulação em Amplitude (AM)

Com esta nova ferramenta, obtêm-se um bom método de se trabalhar com sinais modulados:

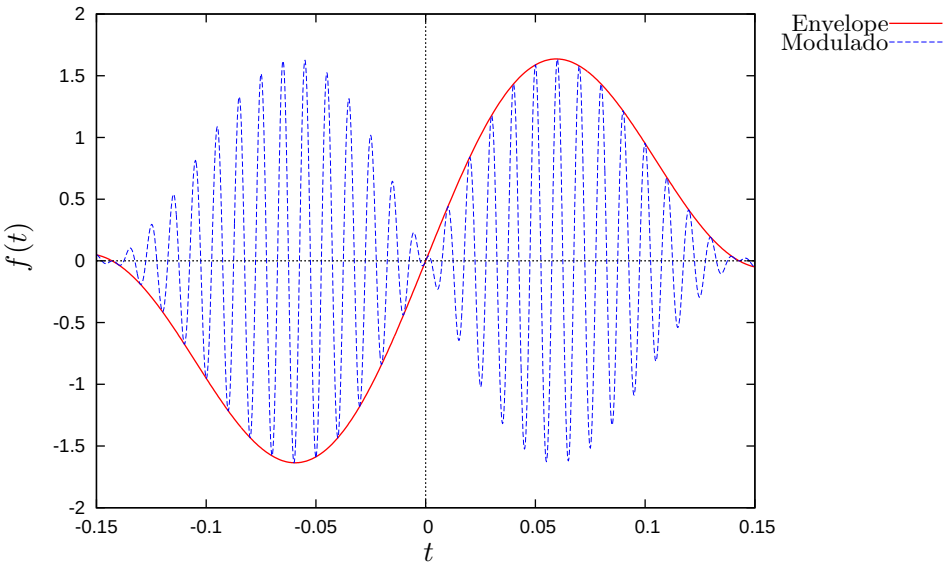


Figura 5.3: Função original plotada como o envelope da função modulada.

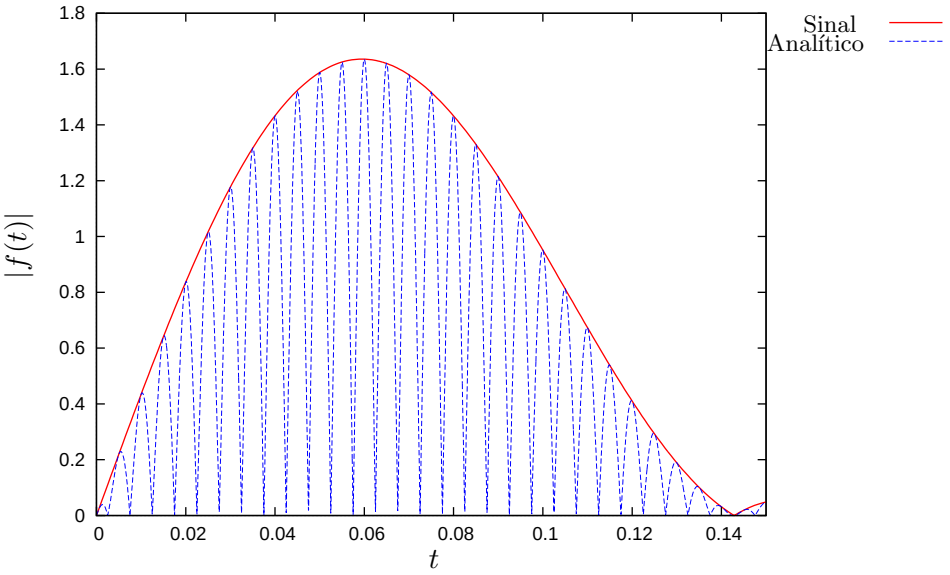


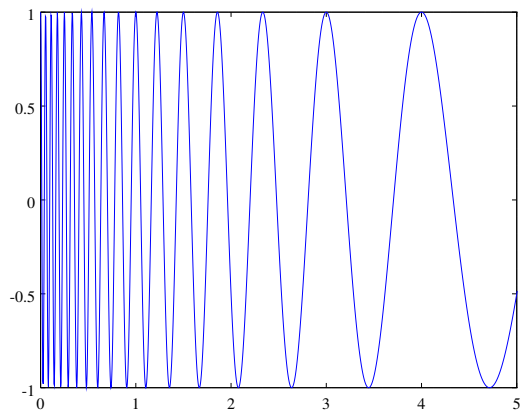
Figura 5.4: Modulo da função analítica é mostrada como o envelope do módulo do sinal modulado.

5.2 Análise de tempo-frequência com a frequência instantânea

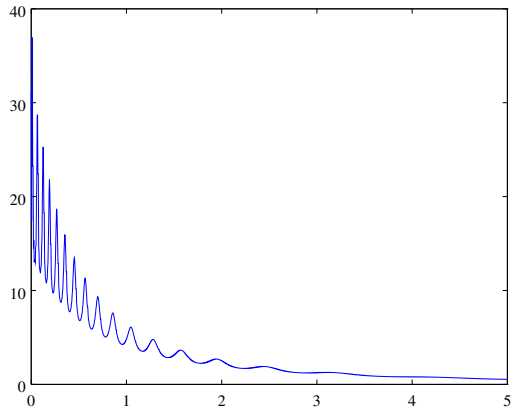
A fórmula definida para a frequência instantânea (como derivada da fase) pode ser usada para a construção de um gráfico de frequência por tempo, porém ela só tem sentido quando o sinal usado não é multicomponente, ou seja, duas frequências não ocorrem ao mesmo tempo. Se este não for o caso, a propriedade de frequência instantânea não possui um sentido físico claro o suficiente. Para exemplificar, será mostrada a curva da frequência instantânea para alguns sinais.

5.2.1 Exemplos

O mesmo chirp já analisado anteriormente com transformadas janeladas e *wavelets* tem agora sua frequência instantânea mostrada em uma curva na Figura 5.5. Nota-se que apesar das oscilações presentes, ainda é possível tirar alguma informação do gráfico. Entretanto, quando duas frequências ocorrem *simultâneamente*, não importa quão simples seja a função, a frequência instantânea não dá uma indicação boa das frequências reais do sinal. Isto pode ser visto na Figura 5.6, onde é mostrada a frequência instantânea para 3 curvas diferentes, uma onda com 1Hz, uma com 3Hz, e uma terceira com as duas ao mesmo tempo. Nas duas primeiras, as curvas obtidas são curvas contantes com a correta frequência da onda. Na terceira curva, obtem-se a média das duas. Isto mostra que, para sinais reais, não faz sentido o cálculo direto da frequência instantânea.



a)



b)

Figura 5.5: Figura a): chirp linear analisado. Figura b): sua frequência instantânea

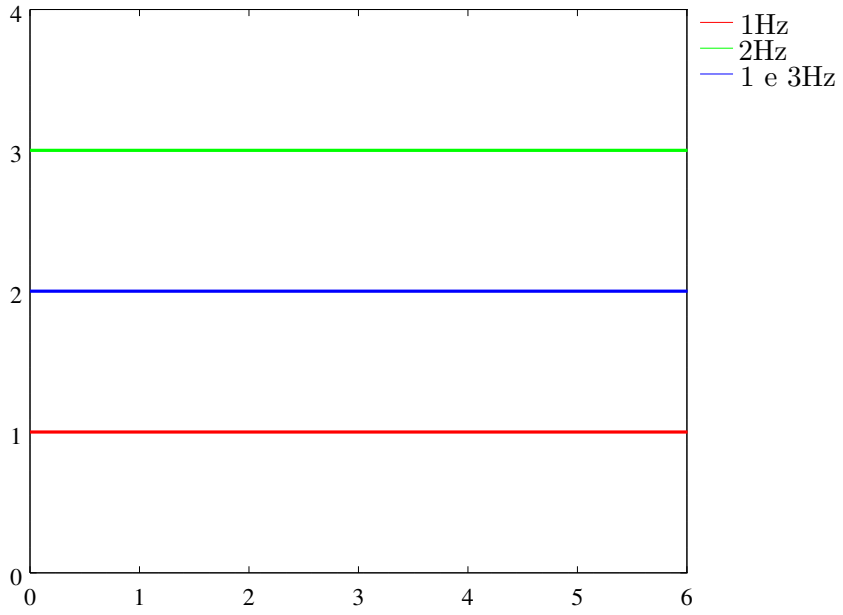


Figura 5.6: Valores obtidos no cálculo da frequência instantânea de 3 cossenos diferentes, um com 1Hz (em vermelho), um com 3Hz (em verde) e uma soma dos dois (em azul).

5.3 Análise com filtros e frequência instantânea

Para que se possa fazer uma representação de tempo-frequência realmente útil com a transformada de *Hilbert* , podem ser feitas filtragens no sinal original com diferentes filtros passa-banda, calcular as frequências instantâneas em cada banda, e usar estes resultados para se construir uma representação de tempo-frequência.

5.3.1 Exemplo de filtro passa-baixas

Para exemplificar o uso de filtros, será feita agora a filtragem de um sinal com dois filtros diferentes. Tendo-se um sinal com duas frequências diferentes, como representado na Figura 5.7, podemos retirar uma de suas frequências usando uma janela conveniente no domínio da frequência. Na Figura 5.8 temos a filtragem do sinal com uma janela de Hanning, e na Figura 5.9 temos a filtragem do mesmo sinal com uma janela retangular. Note que os sinais filtrados possuem perdas de amplitudo mesmo nas frequências que não deveriam ser filtradas, e note também que o filtro retangular, apesar de ser mais simples, fez uma filtragem um pouco pior do que a janela de Hanning.

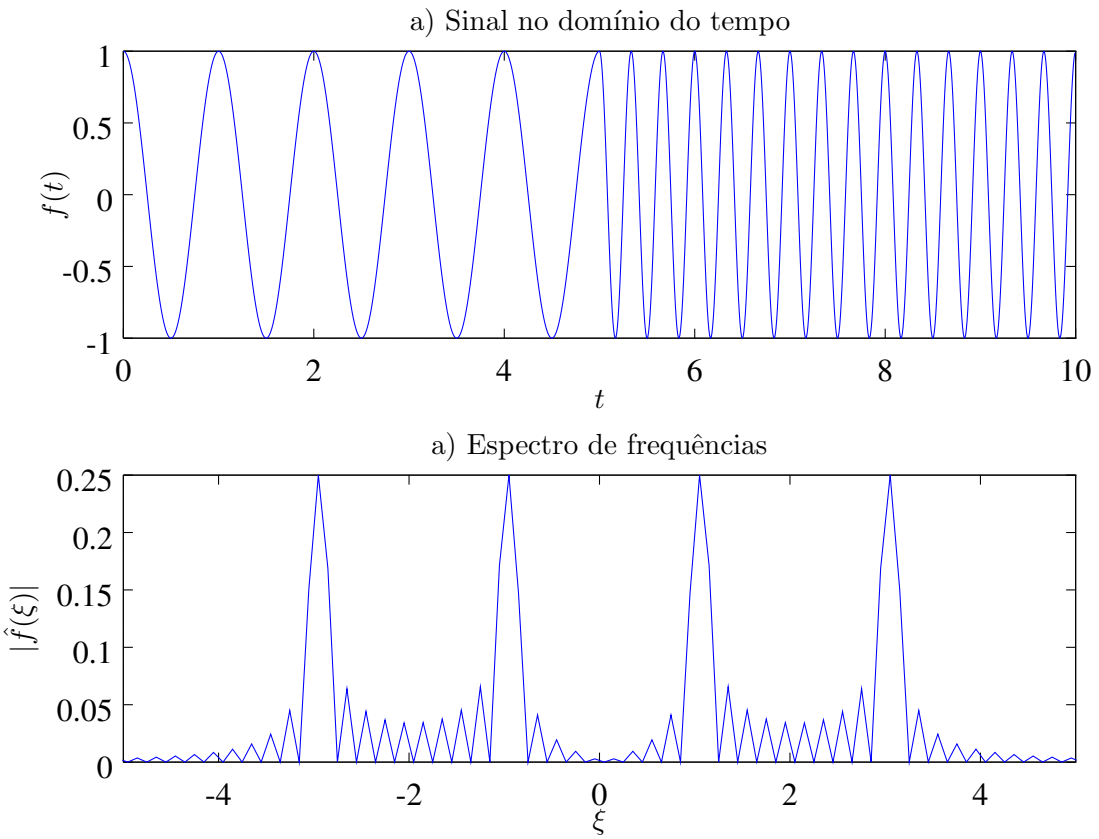


Figura 5.7: Sinal a ser filtrado no domínio do tempo (Figura a) e no domínio da frequência (Figura b).

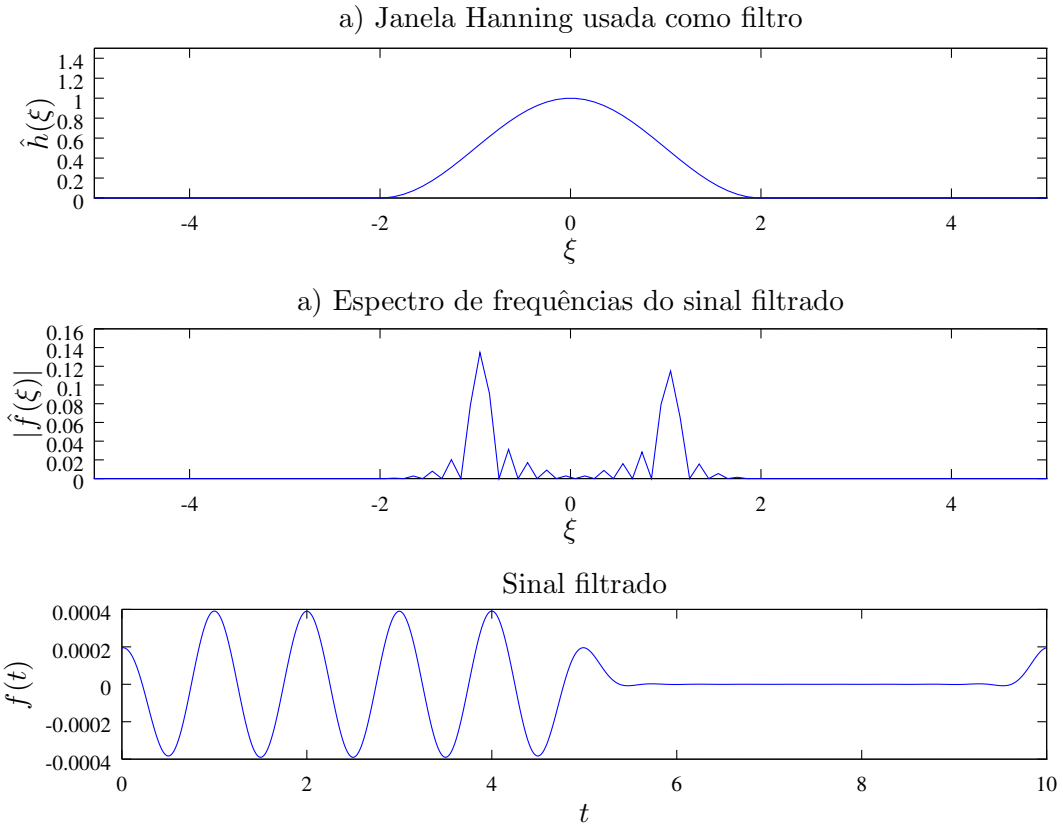


Figura 5.8: Janela usada (Figura a), sinal filtrado no domínio da frequência (Figura b), e sinal filtrado no domínio do tempo (Figure c).

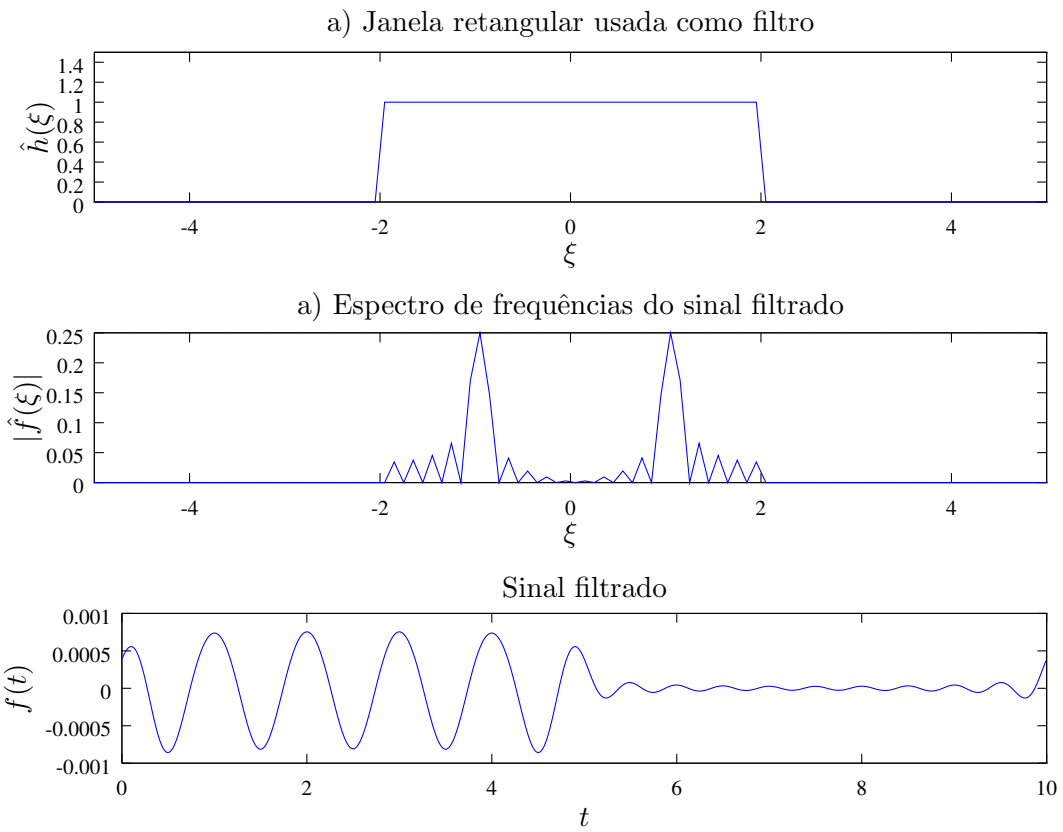


Figura 5.9: Janela usada (Figura a), sinal filtrado no domínio da frequência (Figura b), e sinal filtrado no domínio do tempo (Figure c).

Os filtros usados em conjunto com a transformada de *Hilbert* serão filtros passa-banda, entretando. Diferentemente das janelas usadas aqui, estes filtros teriam uma forma parecida com o filtro da Figura 5.10, caso a janela de Hanning fosse usada como filtro passa-bandas também.

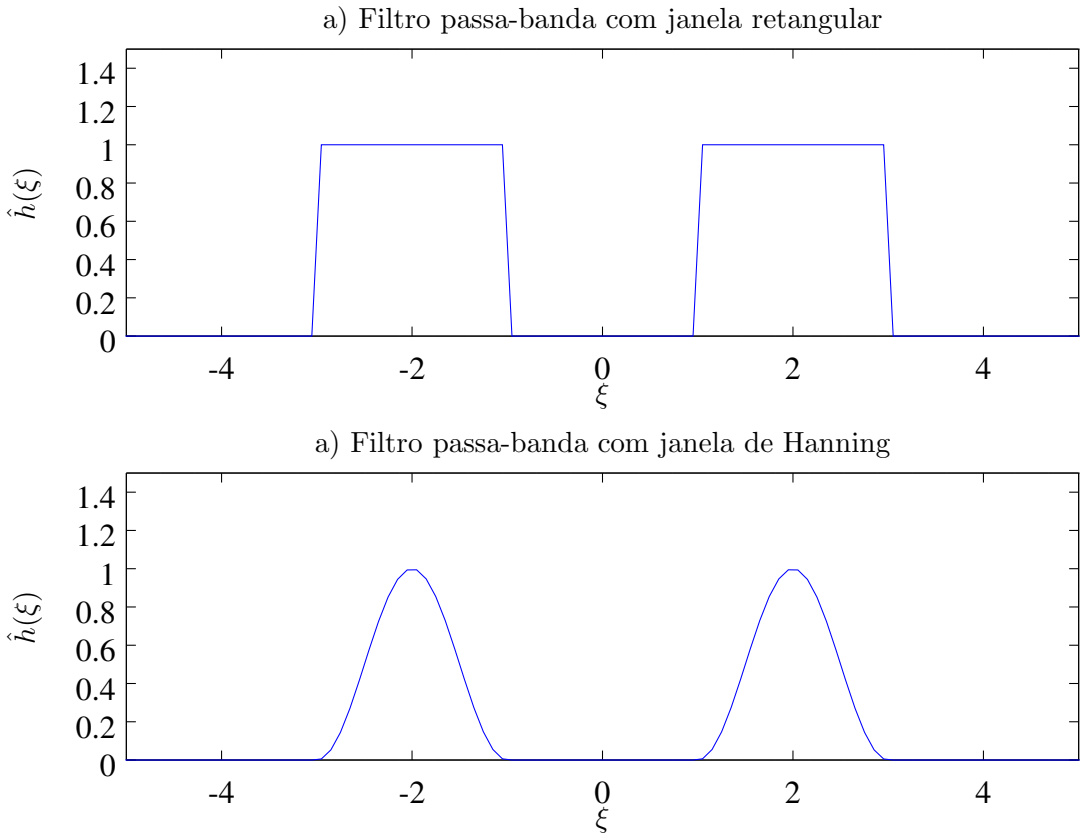


Figura 5.10: Filtros passa-banda. Janela retangular (Figura a) e janela de Hanning (Figura b).

5.3.2 Exemplo de espectro de Hilbert com filtros passa-banda

Para ilustrar, serão usadas janelas de Hanning como filtros passa-banda para calcular o espectro de *Hilbert* do chirp linear da Figura 5.11. O código está na página 167. Usando várias janelas de largura de aproximadamente 1Hz, transladadas no domínio da frequência, como filtros para o sinal, e por fim usando a transformada de *Hilbert* para calcular o valor absoluto de cada frequência no tempo, obtemos as imagens da Figura 5.12 e 5.15.

A ideia a partir disto é usar estes vários vetores do valor absoluto dos sinais analíticos gerados e usa-los para a cosntrução de uma representação de tempo-frequência, como mostrada na Figura 5.16. O espectro de *Hilbert* construido conseguiu indicar corretamente o comportamento do sinal.

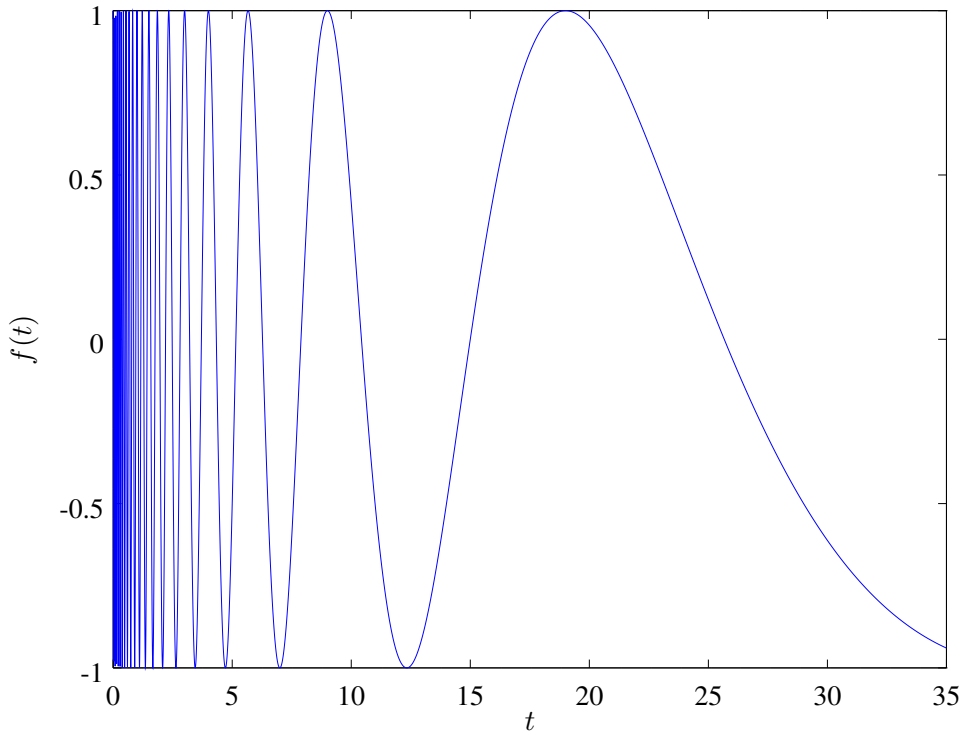


Figura 5.11: Sinal a ser analisado com a transformada de *Hilbert*

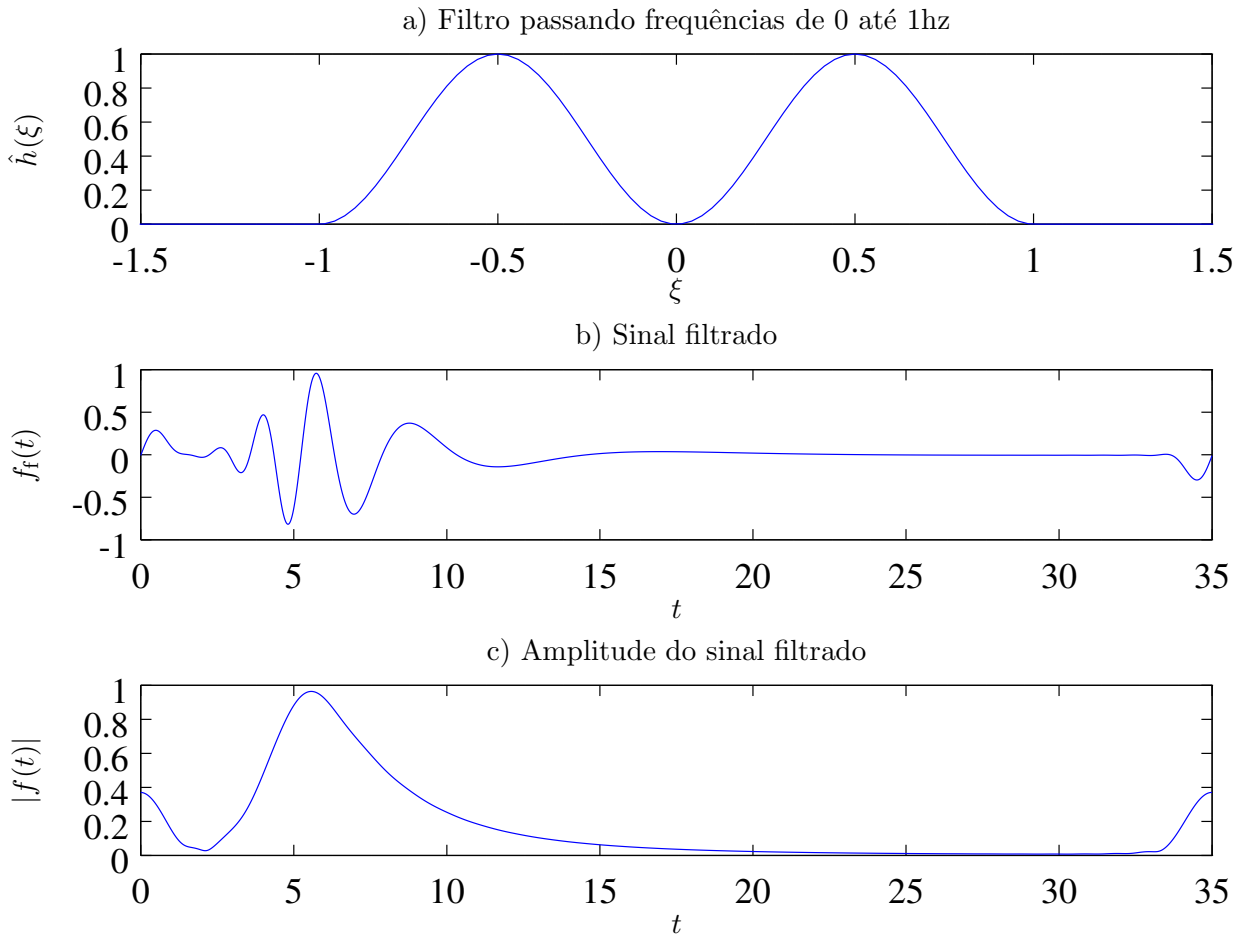


Figura 5.12: Figura a: filtro passa-banda usado. Figura b: sinal filtrado. Figura c: amplitude do sinal.

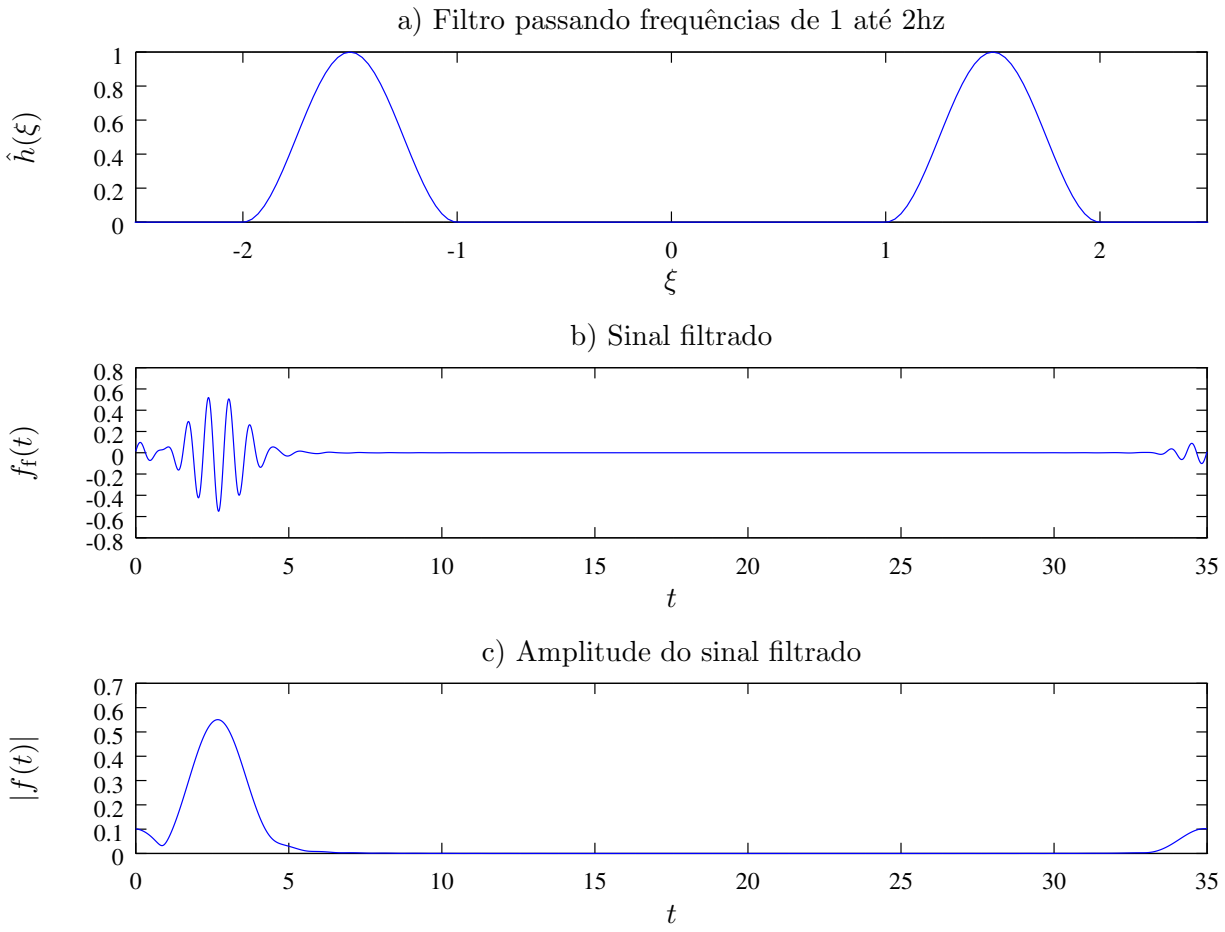


Figura 5.13: Figura a: filtro passa-banda usado. Figura b: sinal filtrado. Figura c: amplitude do sinal.

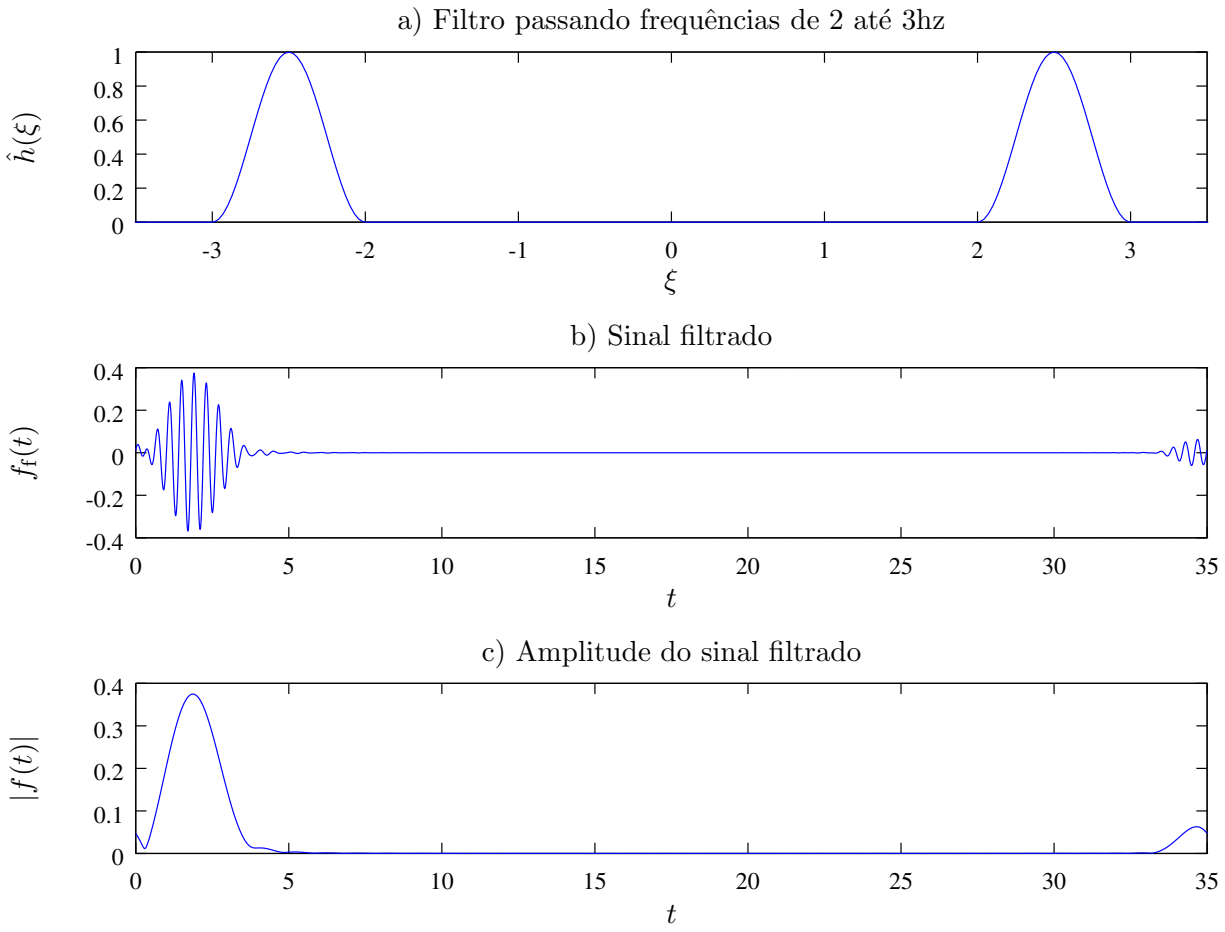


Figura 5.14: Figura a: filtro passa-banda usado. Figura b: sinal filtrado. Figura c: amplitude do sinal.

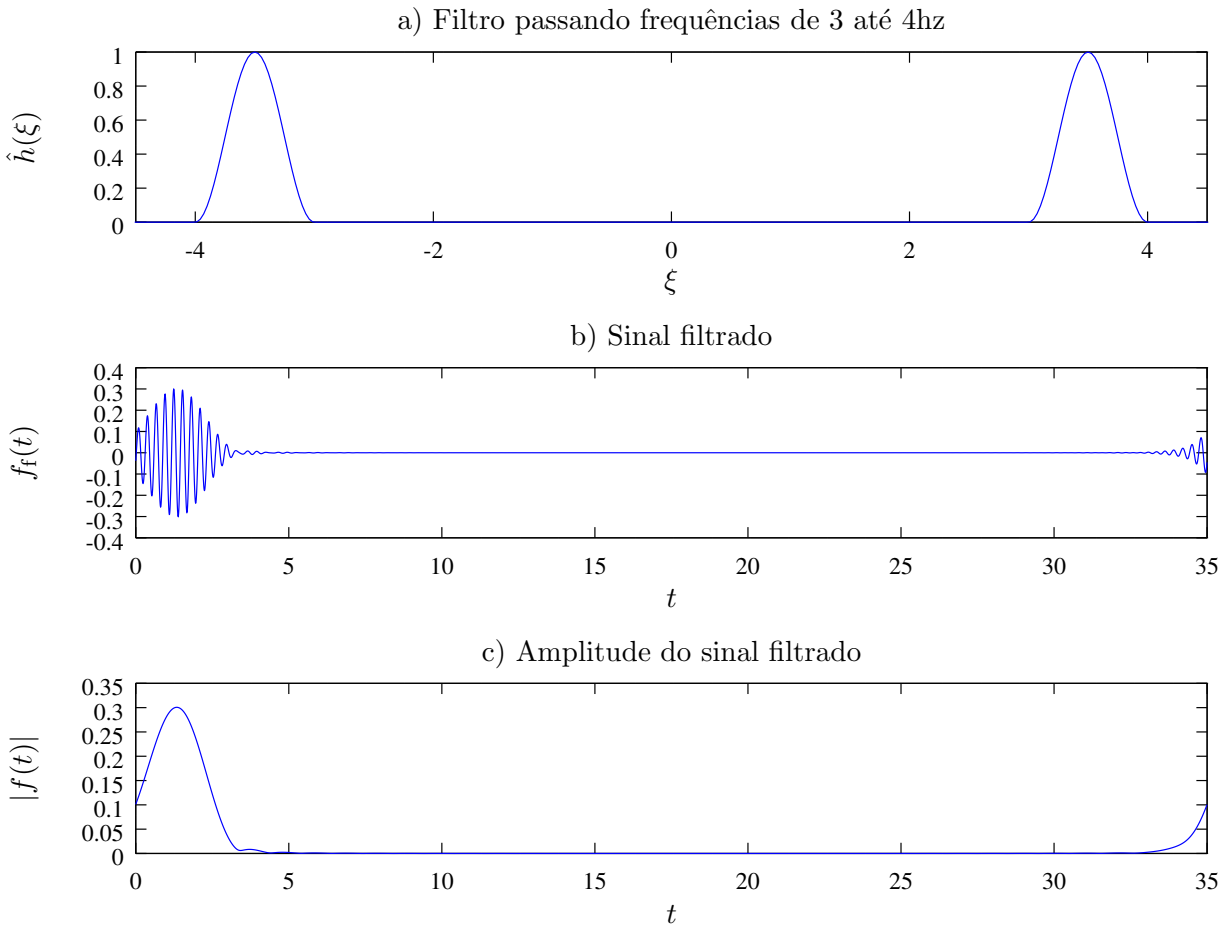


Figura 5.15: Figura a: filtro passa-banda usado. Figura b: sinal filtrado. Figura c: amplitude do sinal.

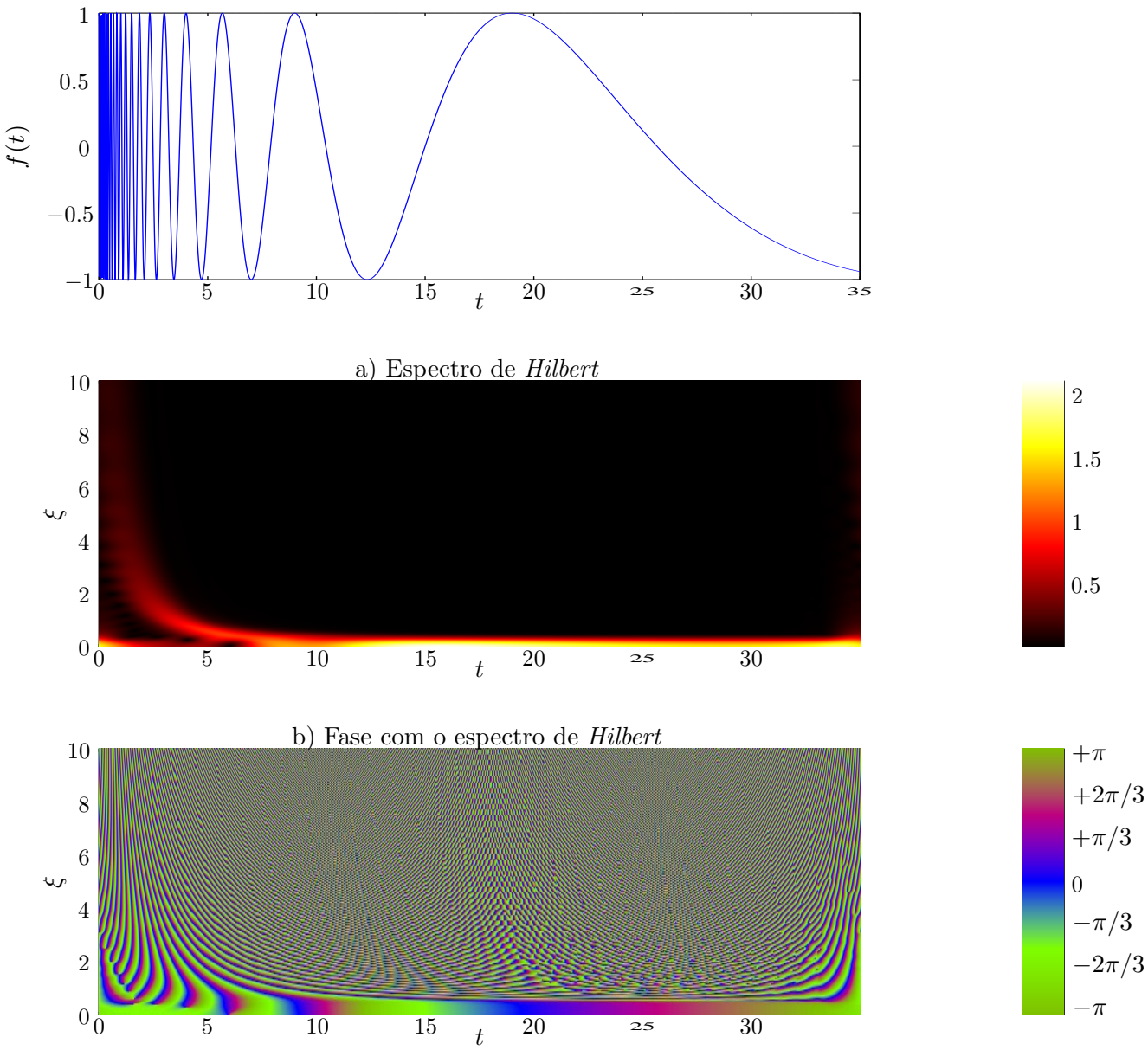


Figura 5.16: Figura a: Espectro de *Hilbert* de um chirp linear. Figura b: Fase com o espectro de *Hilbert*

5.4 Comparação com a transformada *wavelet*

Olhando o espectro de frequências da função de Morlet, observa-se que ela pode ser também usada como um filtro passa-bandas (Figuras 5.17 a 5.19). Nesta seção será construído um espectro de *Hilbert* usando esta função como filtro, e uma escalograma usando a mesma função.

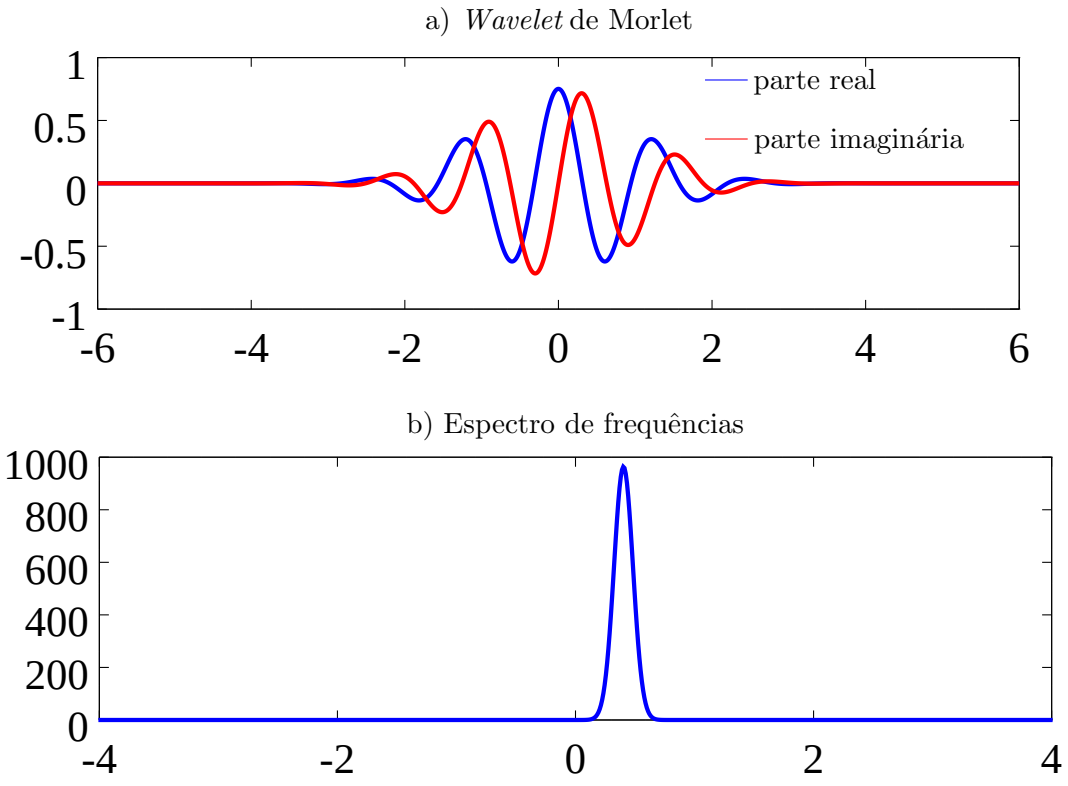


Figura 5.17: *Wavelet* de Morlet frequência central próxima a 0.25Hz

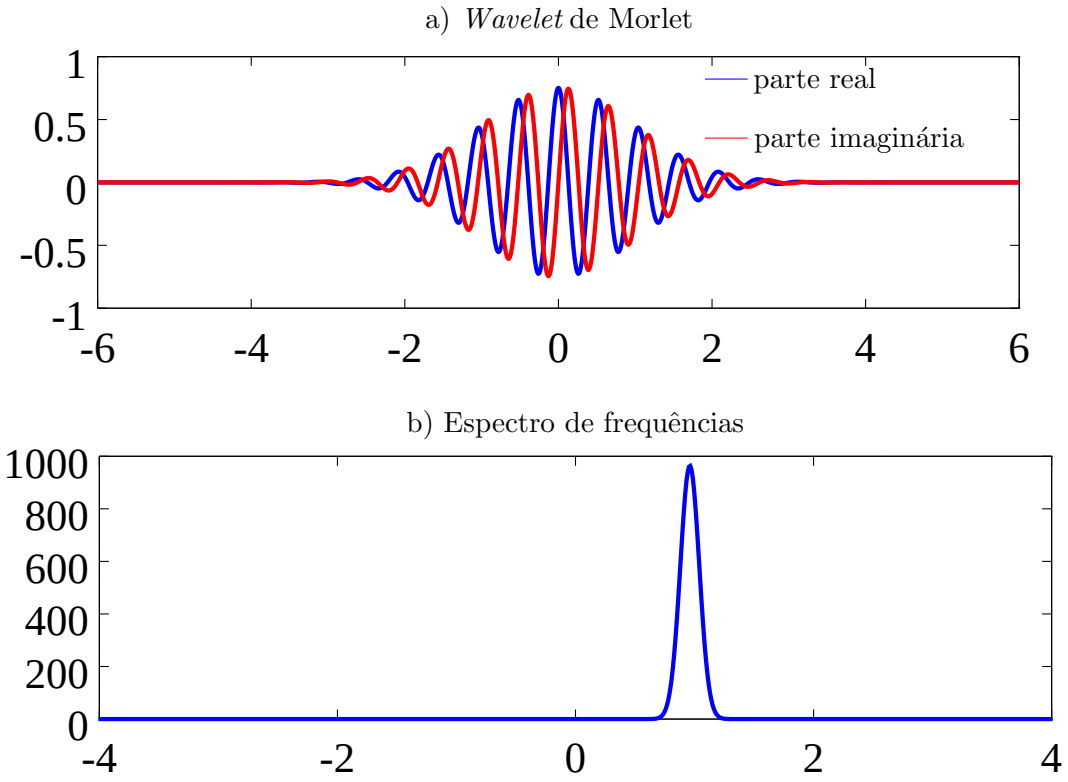


Figura 5.18: *Wavelet* de Morlet frequência central próxima a 1Hz

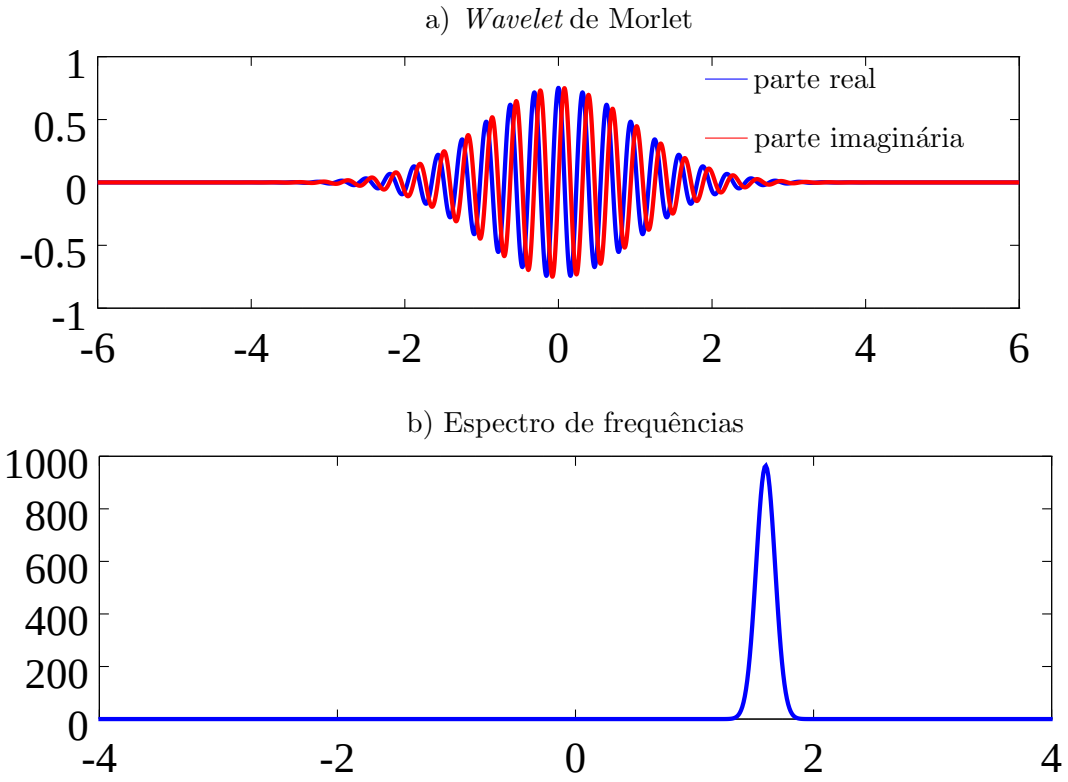


Figura 5.19: *Wavelet* de Morlet frequência central próxima a 1.75Hz

5.4.1 Chirp Linear crescente

Na Figura 5.20 observamos o valor absoluto da transformada *wavelet* de um chirp linear em comparação com o seu espectro de *Hilbert* . Pode ser observado que aqui a transformada de *Hilbert* não só faz um bom trabalho em analisar as frequências, como é capaz de obter a característica linear do sinal.

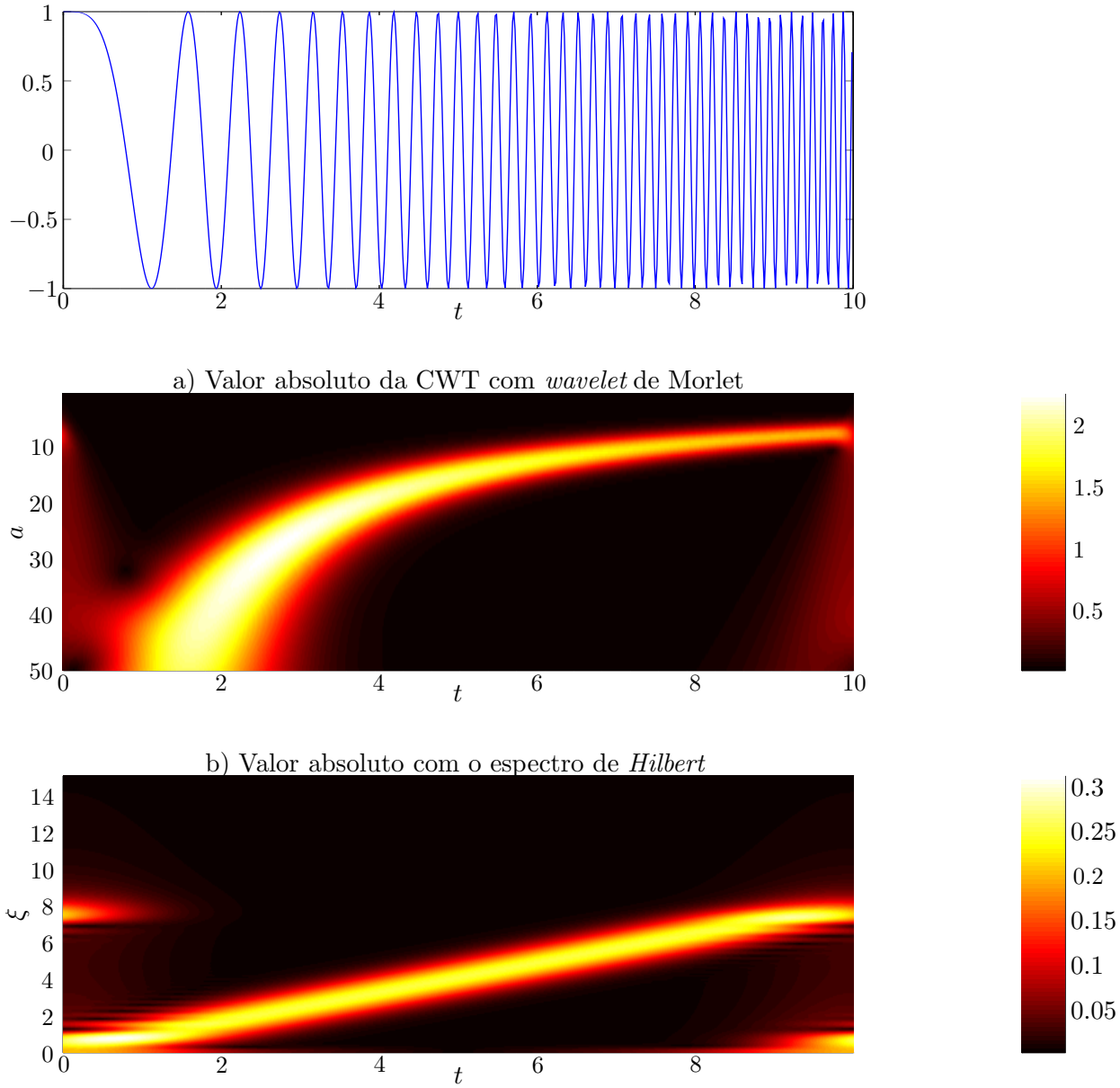


Figura 5.20: Comparação de representações de tempo-frequência do chirp linear crestente com a transformada *Wavelet* (Figura a) e a transformada de *Hilbert* (Figura b).

5.4.2 Soma de chirps hiperbólicos

Na Figura 5.21 observamos a mesma comparação, desta vez para a soma de chirps hiperbólicos. Neste caso, além de sugerir a mudança rápida na frequência, a transformada de *Hilbert* não fez um trabalho muito bom.

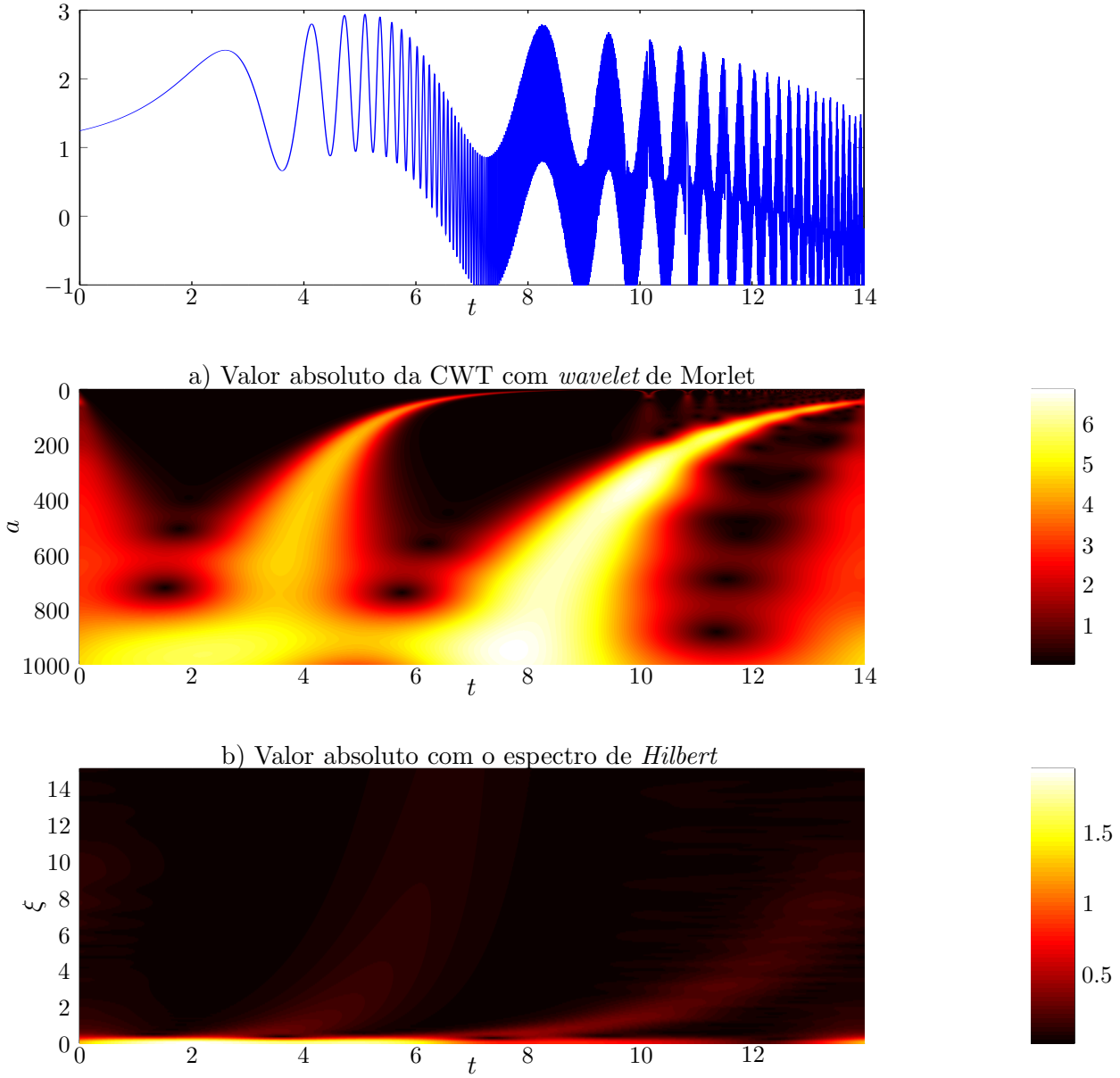


Figura 5.21: Comparação de representações de tempo-frequência de uma soma de chirps hiperbólicos com a transformada *Wavelet* (Figura a) e a transformada de *Hilbert* (Figura b).

5.4.3 Função de Cantor

Na Figura 5.22 observamos comparação novamente com a função de Cantor. Neste caso, o espectro de *Hilbert* demonstrou que a função possui energia em várias frequências, mas não conseguiu detectar as formas fractais presentes nas transformadas *wavelet* da função.

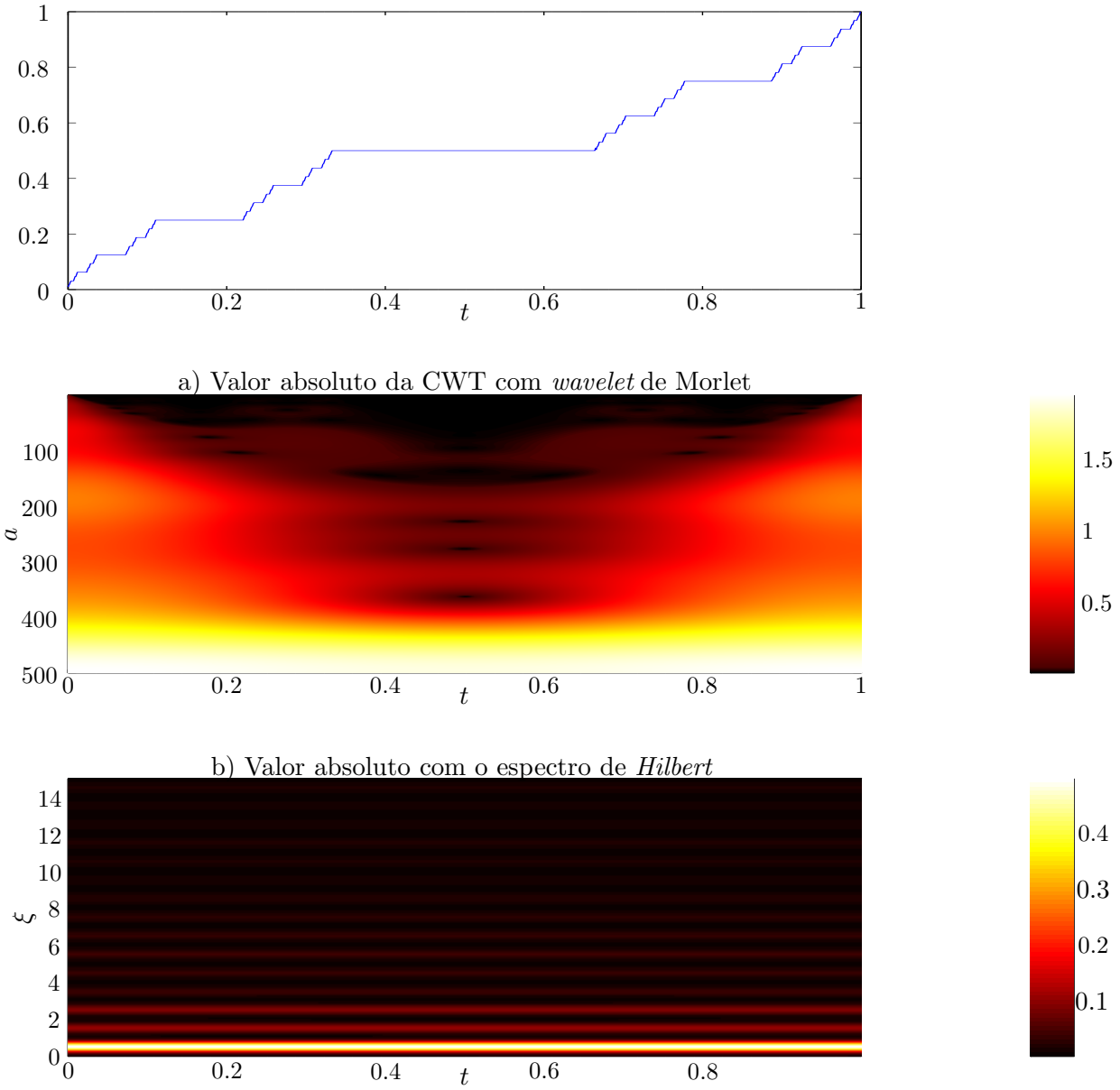


Figura 5.22: Comparação de representações de tempo-frequência da função de Cantor com a transformada *Wavelet* (Figura a) e a transformada de *Hilbert* (Figura b).

Pode ser observado que, assim como as outras transformadas, a transformada de *Hilbert* como método de analise também possui suas limitações. Entretanto, no pŕoximo capítulo será mostrada uma maneira melhor de se usar a esta transformada para ser obtida uma representação tempo-frequência.

Capítulo 6

Transformada de *Hilbert-Huang*

A transformada de *Hilbert-Huang* é uma ferramenta para se obter a decomposição de sinais em bases chamadas de IMF (*Intrinsic Mode Functions*). É um método construído para se trabalhar tanto com sinais não estacionários como sinais não lineares, e diferentemente das outras transformadas apresentadas até agora, ela é um algoritmo, e não uma ferramente teórica aplicada.

6.1 *Intrinsic Mode Functions* (IMF)

Uma IMF é uma função que possui duas condições principais:

1. Em todo o intervalo, o número de zeros e extremos locais deve ser o mesmo (ou diferir por apenas uma únidade).

2. Em qualquer ponto, a média entre os envelopes definido pelos extremos máximos e pelos extremos mínimos deve ser zero.

A IMF representa um modo de oscilação simples, em contrapartida como um função harmônica simples. Isto significa que a transformada de *Hilbert* de uma IMF será uma função bem comportada.

6.2 *Empirical Mode Decomposition* (EMD)

A EMD é um método que atua diretamente no domínio do tempo e supõe que a função é constituída de de vários modos oscilatórios simples, as IMFs. É o processo chave da transformada de *Hilbert-Huang*, onde o sinal original é dividido em várias IMF. Seu processo será exemplificado com a função de chirps hiperbólicos da Figura 6.1.

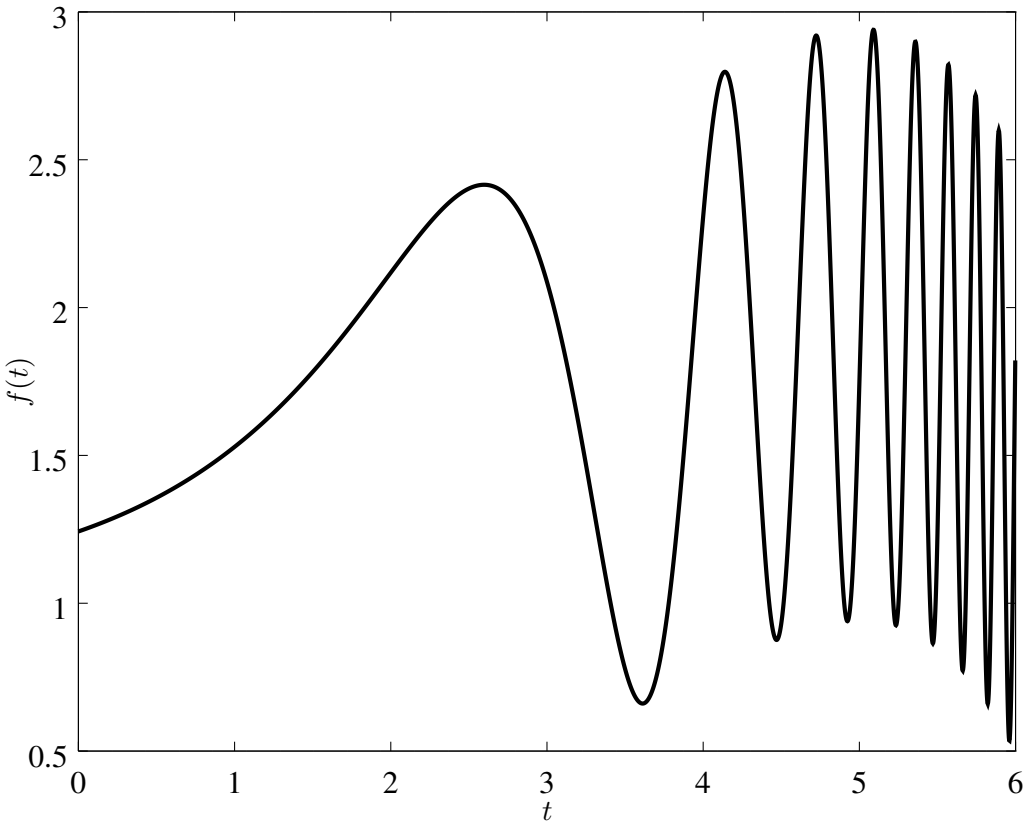


Figura 6.1: Sinal que será analisado com a transformada de *Hilbert-Huang* .

6.2.1 Método

Será exemplificada aqui a computação do método EMD.

Encontrar os extremos

O primeiro passo é encontrar todos os máximos e mínimos da função, como visto na Figura 6.2

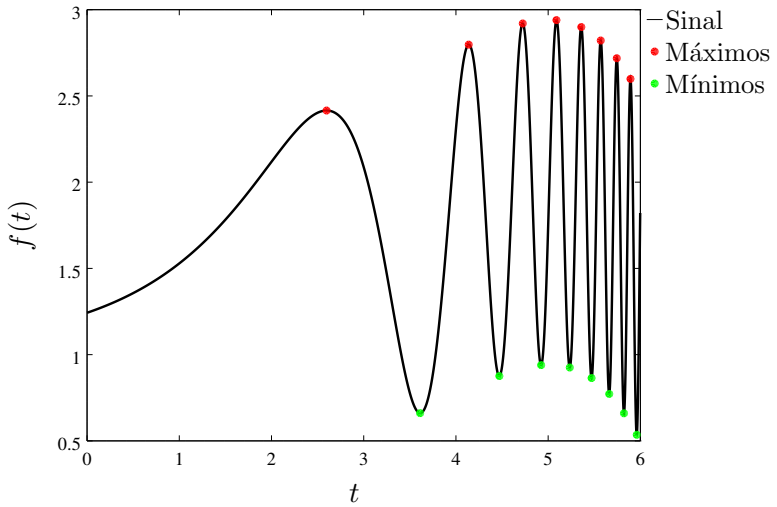


Figura 6.2: Máximos e mínimos do sinal.

Encontrar os envelopes

Em seguida, os máximos devem ser interpolados entre sí com uma função spline (função definida em partes por polinômios) cúbica, conseguindo assim o envelope superior da função, e o mesmo deve ser feito para os pontos de mínimo (Figura 6.3).

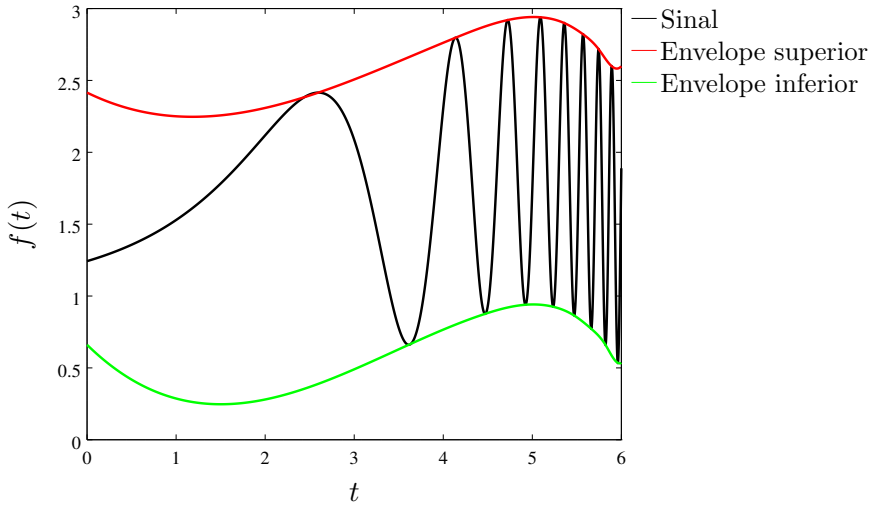


Figura 6.3: Envelopes superior e inferior.

Encontrar a média dos envelopes

O terceiro passo é encontrar a média entre estes envelopes (Figura).

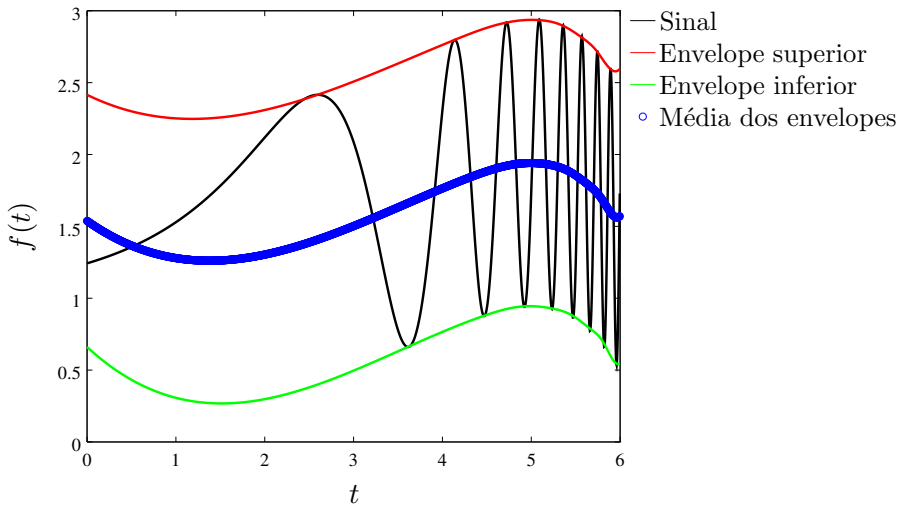


Figura 6.4: Média dos envelopes.

Subtrair a média da função

Em seguida, de-se subtrair a função desta média, gerando o primeiro "proto-modo" (Figura).

$$h_1 = f(t) - m_1(t) \tag{6.1}$$

em que $m_1(t)$ é a média encontrada.

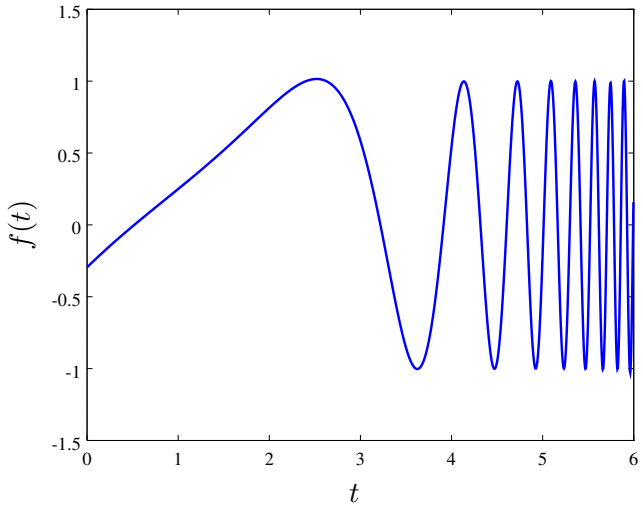


Figura 6.5: Primeiro "proto-modo".

Neste caso, chamamos de "proto-modo" porque a função gerada assim pode não necessariamente satisfazer as condições de uma IMF.

Aprimoração do modo de oscilação

Para que o modo se torne uma IMF, a função conseguida $h_1(t)$ deve ser pegada como função inicial e todo o processo do EMD deve ser aplicado nela novamente:

$$h_{1,k} = h_{1,k-1} - m_{1,k-1} \tag{6.2}$$

Quando a função conseguida, com um certo número k de iterações, for suficientemente boa, então ela é pegada como a o primeiro modo c_1.

$$c_1 = h_{1,k} \tag{6.3}$$

O critério de parada usado foi se melhorar a função IMF até que o desvio padrão

$$SD = \frac{\sum_{n=1}^{N} |h_k - h_{k-1}|^2}{\sum_{n=1}^{N} h_{k-1}^2} \tag{6.4}$$

fosse menor que algum valor (no algoritmo escolhido, este valor foi arbitrariamente escolhido como sendo 10^{-6}.

Encontrar os outros modos

Após ser encontrado este modo c_1, ele deve ser subtraído da função original $f(t)$:

$$r = f(t) - c_1 \tag{6.5}$$

para encontrar o resíduo r. Os próximos modos deverão ser encontrados aplicando-se EMD neste resíduo.

O processo é feito até que o resíduo obtido seja uma função monotônica (totalmente crescente ou totalmente decrescente), pois assim ele não possui mais nenhum modo oscilatório, ou ele seja pequeno demais para ser levado em consideração. Ao se conseguir k IMFs, a função original pode ser escrita como

$$f(t) = \sum_{n=1}^{k} c_n + r \tag{6.6}$$

Onde o resíduo é, em geral, descartado.

As IMFs conseguidas para o chirp estão na Figura 6.7.

6.3 *Hilbert Spectrum Analysis* (HSA)

Após ser obtidas as IMFs, a transforada de *Hilbert* de cada um pode ser calculada para se obter os espectros de *Hilbert*. Em alguns casos, a análise dos espectros com os modos separados através do algoritmo de EMD pode mostrar melhor do que a análise do espectro com a função original. Na Figura 6.7 está a comparação entre os espectros de *Hilbert* do sinal (Figura 6.7 a) e dos espectros das IMFs obtidas (Figuras 6.7 b, c e d).

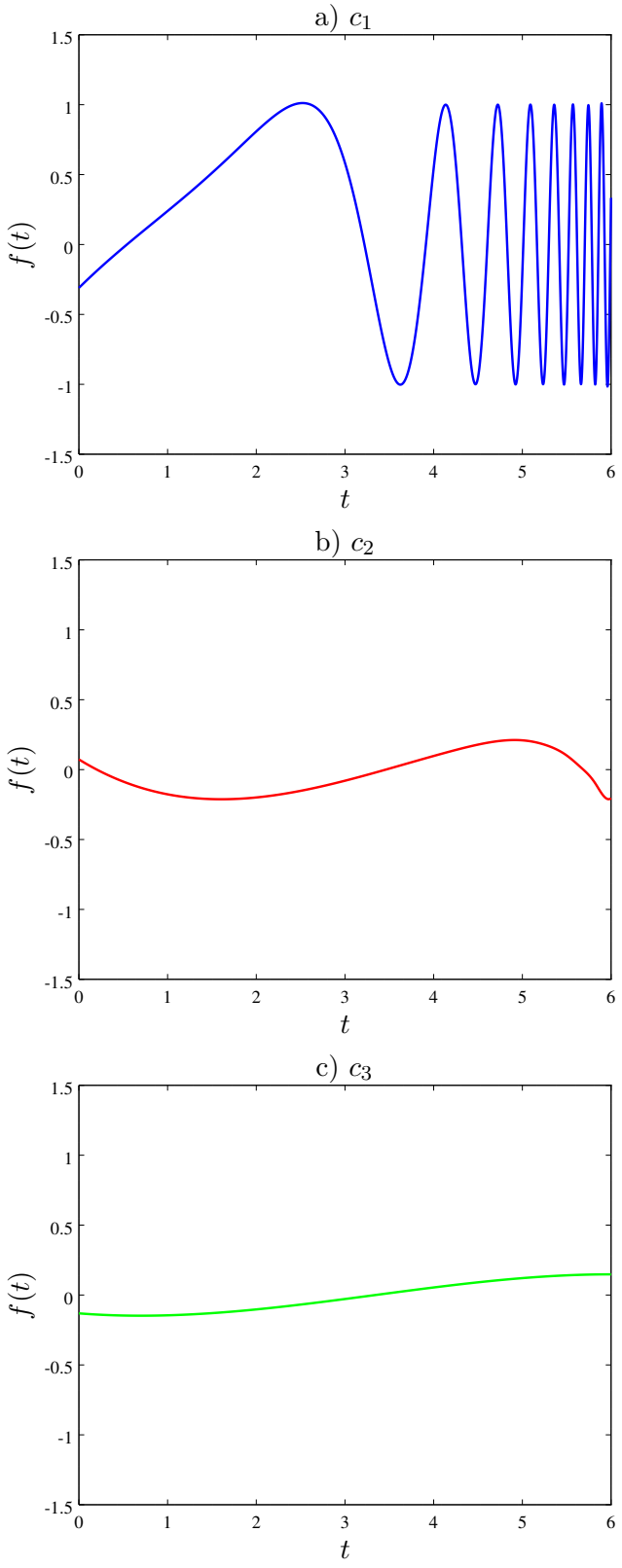


Figura 6.6: Figuras a até b: Primeiro, segundo e terceiro IMFs do chirp hiperbólico.

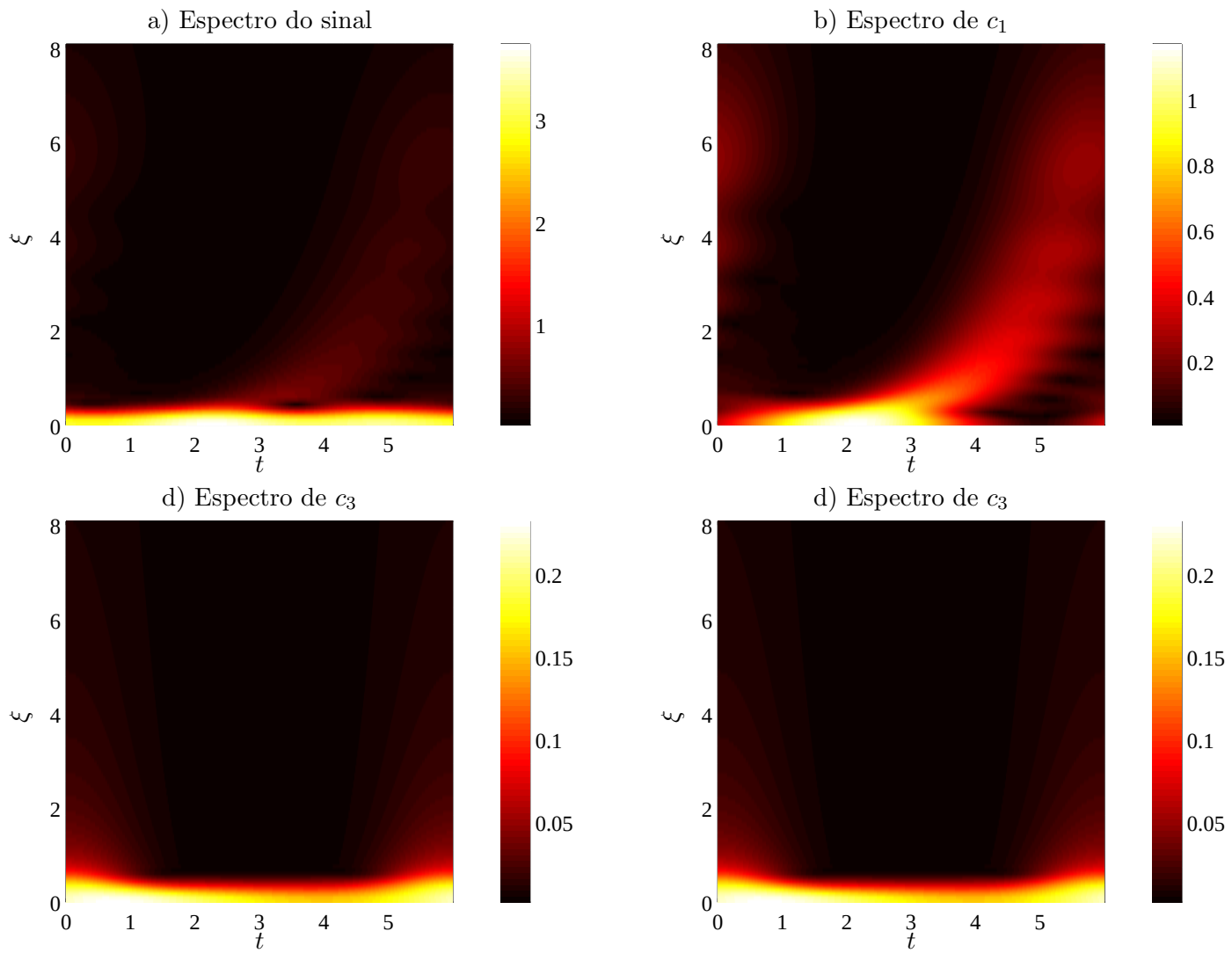


Figura 6.7: Figuras a: espectro de *Hilbert* do sinal. Figuras b até c: espectro de *Hilbert* dos 3 modos obtidos pela EMD.

Capítulo 7

Transformadas de *Fourier* e *Wavelet* contínuas em duas dimensões

As transformadas apresentadas até agora são foram usadas para analisar sinais unidimensionais no tempo, decompondo-os em componentes de frequência temporal. Estas mesmas transformadas podem ser generalizadas para se trabalhar em sinais bidimensionais. Tais sinais serão denotados por:

$$f(\mathbf{x}) = f(x, y) \tag{7.1}$$

em que $\mathbf{x} = (x, y)$ serão suas duas componentes no espaço. O artigo [4] feito para a Conferência Brasileira de Dinâmica, Controle e Aplicações (DINCON) 2015 foi criado baseado neste capítulo.

7.1 Transformada de *Fourier* para duas dimensões

Relembrando-se a definição da transformada de *Fourier* para sinais unidimensionais:

$$\hat{f}(\xi) = \int_{-\infty}^{+\infty} f(t) \cdot e^{-2\pi\imath\xi t} dt$$

a transformada pode ser generalizada para duas dimensões do seguinte modo:

$$\hat{f}(\boldsymbol{\xi}) = \hat{f}(\xi_x, \xi_y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x, y) \cdot e^{-2\pi\imath(\xi_x x + \xi_y y)} dxdy \tag{7.2}$$

em que $\boldsymbol{\xi} = (\xi_x, \xi_y)$ são as duas componentes de frequência espacial, nas dimensões x e y respectivamente.

Assim como as transformadas *wavelet* , a nova transformada de *Fourier* obtida é uma função de duas variáveis que em geral é complexa. Portanto, enquanto o sinal original pode ser representado inteiramente por uma imagem, a transformada 2D de *Fourier* pode ser representada por duas: uma para seu valor absoluto e outra para sua fase. Usando como exemplo o sinal $f(x, y) = \cos\left(\frac{2\pi}{4}(x^2 + y^2)\right)$ na Figura 7.1, tem-se as imagens de sua transformada de *Fourier* na Imagem 7.2.

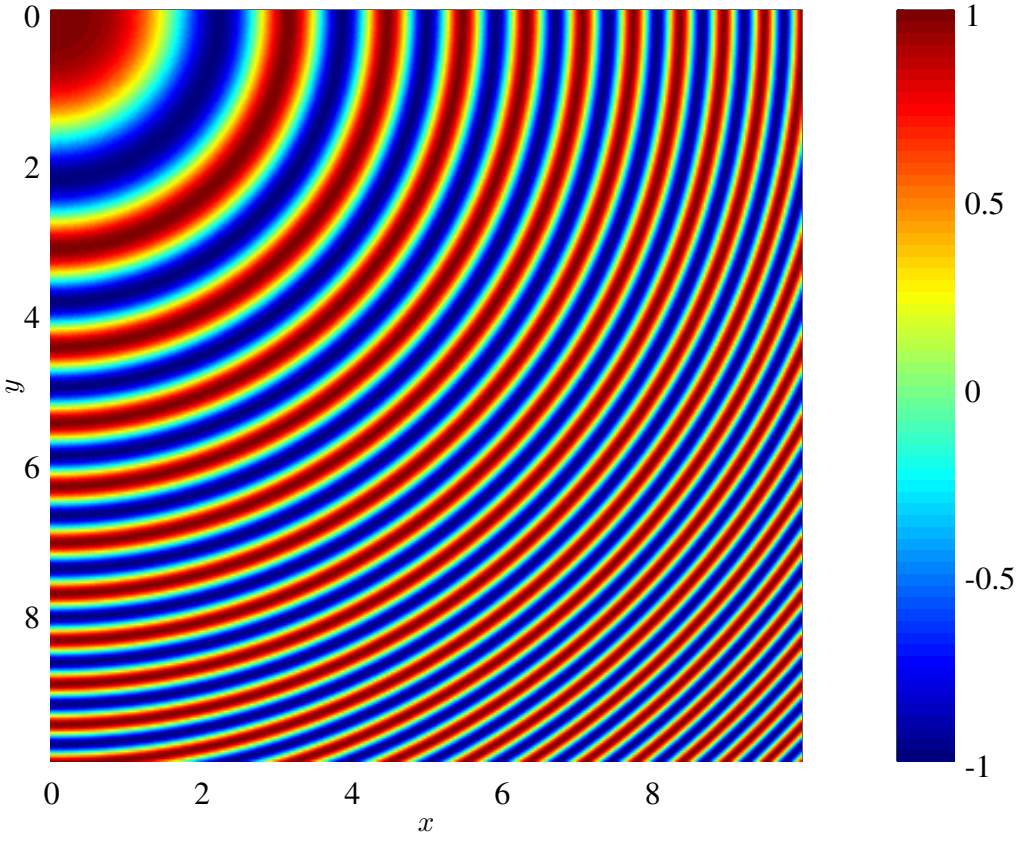


Figura 7.1: $f(x, y) = \cos\left(\frac{2\pi}{4}(x^2 + y^2)\right)$

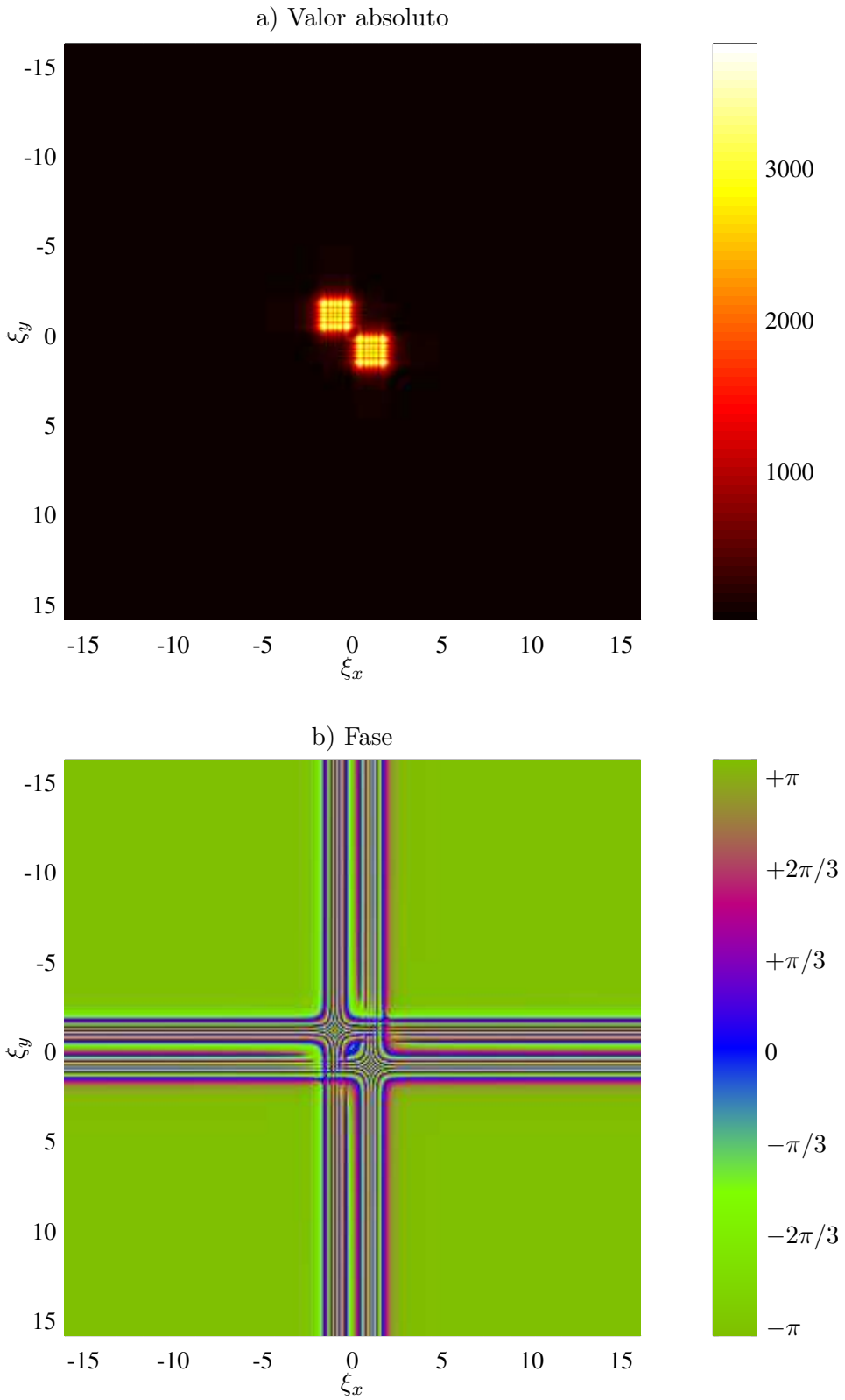


Figura 7.2: Transformada de *Fourier* para suas duas frequências espaciais. Figura a: Valor absoluto. Figura b: Fase.

Neste sentido de imagens e unidades espaciais, a frequência está relacionada com a quantidade de detalhes na função. Se uma imagem for passada por um filtro passa-baixas, obtêm-se uma versão suavizada da imagem (com menos detalhes). O código que gerou esta transformada está na página 169. Se a mesma imagem for passada por um filtro passa-altas, obtêm-se os detalhes da imagem (bordas e linhas). Este efeito será exemplificado a seguir.

7.1.1 Filtragem no domínio de *Fourier*

Pegando uma imagem como a da Figura 7.3 e calculando sua transformada de *Fourier* bidimensional, terá-se as imagens da transformada na Figura 7.4. Olhando o gráfico do valor absoluto da imagem, observa-se uma concentração no centro. As componentes no centro da figura indicam as componentes de menor variação (que no domínio das *wavelets* , indicam as escalas maiores). As componentes mais distante, entretanto, indicam as frequências maiores e guardam informações sobre os detalhes da imagem original.

Aplicando-se um filtro passa baixas como o da Figura 7.5, em que branco indica valor nulo e preto indica valor = 1, consegue-se a imagem da Figura 7.6 a), e aplicando um filtro inverso (0 no centro e 1 nas extremidades), consegue a imagem da Figura 7.6 b). Observa-se que, enquanto a imagem que teve suas componentes de frequência maior retiradas é uma versão suavizada da imagem original, a outra imagem guarda apenas os detalhes mais fortes (perdidos na primeira). O script usado para gerar esta imagem está na página 171.

Figura 7.3: Imagem a ser filtrada.

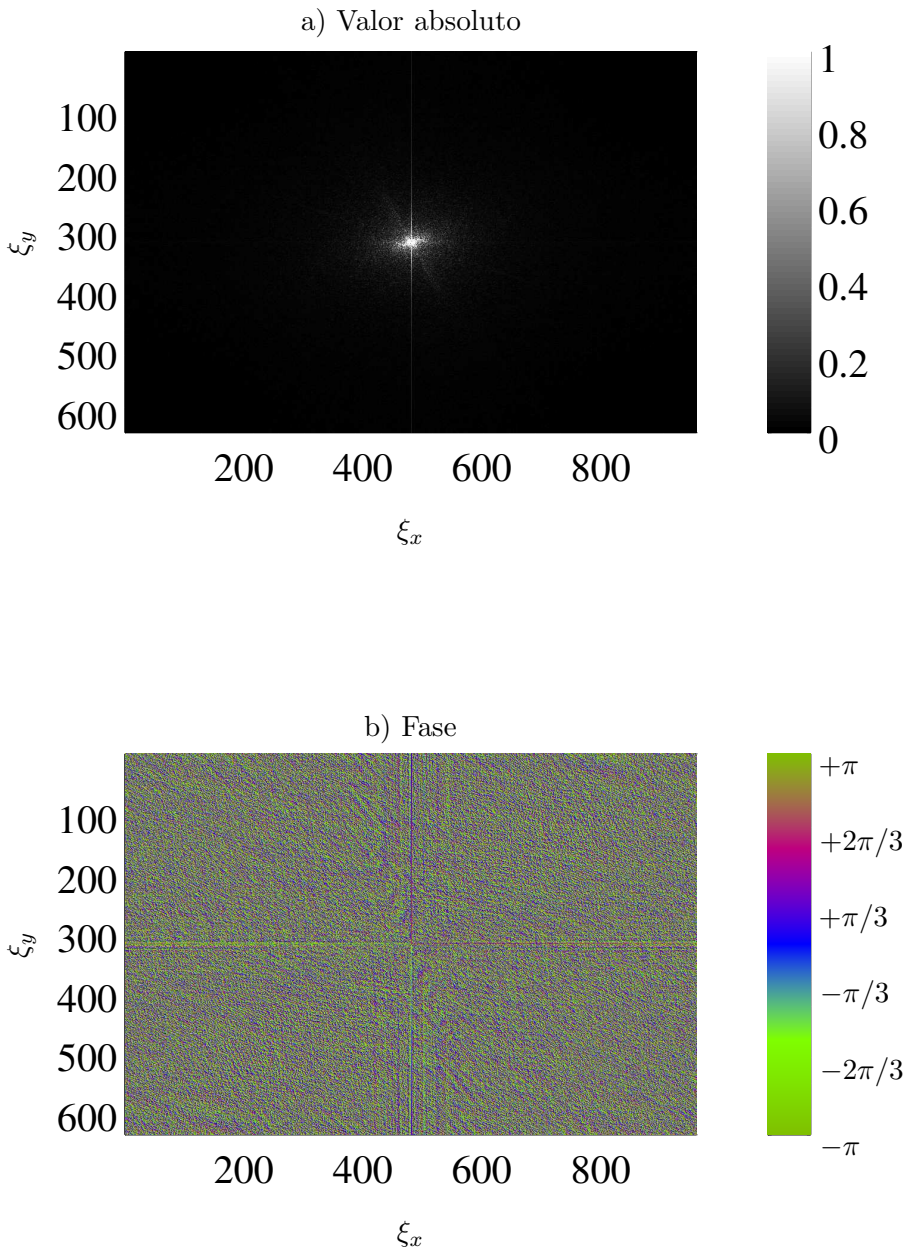


Figura 7.4: Figura a: Valor absoluto da FFT da imagem. Figura b: Fase da FFT da imagem.

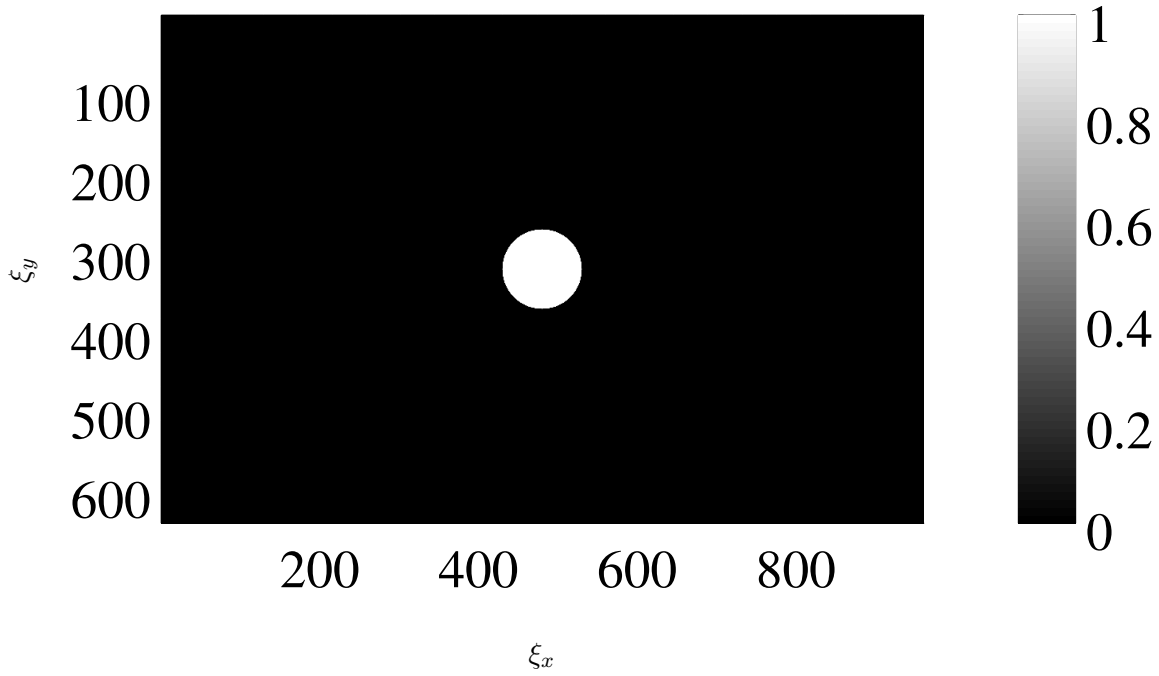


Figura 7.5: Filtro utilizado.

a) Imagem com filtro passa-baixas

b) Imagem com filtro passa-altas

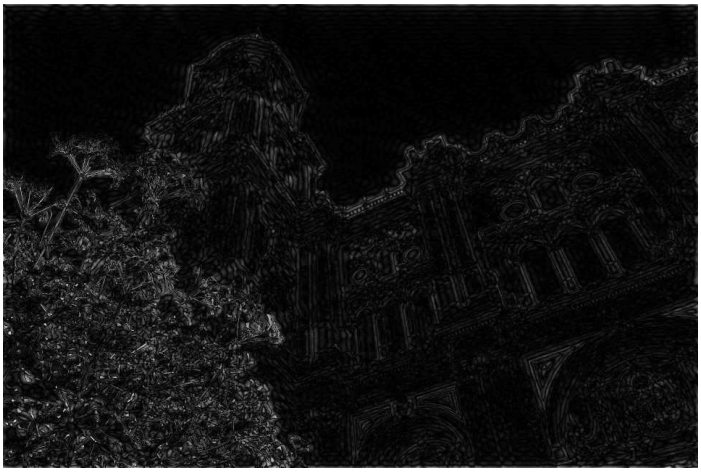


Figura 7.6: Figura a: Valor absoluto da FFT da imagem. Figura b: Fase da FFT da imagem.

7.2 Transformadas *wavelet* contínuas bidimensionais

A transformada *wavelet* contínua para duas dimensões é definida para sinais $f(\mathbf{x}) = f(x, y)$ que satisfazem

$$\int |f(\mathbf{x})|^2 \mathbf{d}^2\mathbf{x} = \int\int |f(x, y)|^2 dxdy < \infty \tag{7.3}$$

Na prática, os sinais serão imagens em preto e branco de valores não negativos limitados por um valor M. Uma imagem colorida equivale à 3 imagens monocromáticas, cada uma representando o valor de cada canal de cor (RGB, do ingles, *red, blue* e itgreen ou seja vermelho, azul e verde respectivamente). Em sinais unidimensionais, as operações aplicadas na *wavelet* analisadora são as operações de dilatação e translação, e calcula-se o produto interno da função com sinais modificados $\psi\left(\frac{t-b}{a}\right)$. No caso de sinais bidimensionais, as operações possíveis são as operações de dilatação, translação e rotação, gerando funções

$$\psi_{a,\theta,\mathbf{b}}(\mathbf{x}) = \frac{1}{a}\psi\left(\frac{\mathbf{r}_{-\theta}\left(\mathbf{x}-\mathbf{b}\right)}{a}\right) \tag{7.4}$$

em que a é a escala, $\mathbf{b} \in \mathbb{R}^2$ é a posição (translação), e $\mathbf{r}_\theta$ representa a rotação, normalmente dada por $\mathbf{r}_\theta(x, y) = (x \cdot \cos\theta - y \cdot \text{sen}\theta, x \cdot \text{sen}\theta + y \cdot \cos\theta)$. Se a função modificada for invariante na rotação, sua transformação será então:

$$\psi_{a,\mathbf{b}}(\mathbf{x}) = \frac{1}{a}\psi\left(\frac{\mathbf{x}-\mathbf{b}}{a}\right) \tag{7.5}$$

As funções *wavelet* em 2D são funções complexas que, assim como no caso 1D, devem satisfazer uma condição de admissibilidade. A condição neste caso é dada por:

$$c_\psi = (2\pi)^2 \int \frac{\hat{\psi}(\boldsymbol{\xi})}{|\boldsymbol{\xi}|^2}\mathbf{d}^2\boldsymbol{\xi} < \infty \tag{7.6}$$

em que $\hat{\psi}$ é a transformada de *Fourier* da função e $\boldsymbol{\xi} = (\xi_x, \xi_y)$ são as frequências espaciais no espaço de *Fourier* . A transformada *wavelet* será dada como o produto interno entre o sinal f e a *wavelet* modificada $\psi_{a,\theta,\mathbf{b}}$:

$$\mathfrak{W}_f^\psi(a, \theta, \mathbf{b}) = \int f(\mathbf{x})\psi_{a,\theta,\mathbf{b}}(\mathbf{x})\mathbf{d}^2\mathbf{x} \tag{7.7}$$

Observa-se que a dimensionalidade da transformada gera problemas de visualização. Tem-se com ela 4 variáveis independentes (escala, rotação e translações nos dois sentidos) para representar duas grandezas diferentes (módulo e fase dos coeficientes). O modo mais simples de visualizar a transformada neste caso em um plano 2D é manter alguns dos parâmetros fixos. Tem-se duas representações básica [3].

1. Representação de posição, em que a e θ são mantidas constantes e a CWT é uma função apenas dos parâmetros de deslocamento $\mathbf{b}$.

2. Representação de escala-ângulo, em que as translações são mantidas constantes e a CWT é uma função apenas da escala e do ângulo.

A representação de posição é a mais comum, e possui muita aplicação em processamento de imagens: detecção de posição, forma e contornos, ou detecção de padrões, etc... .

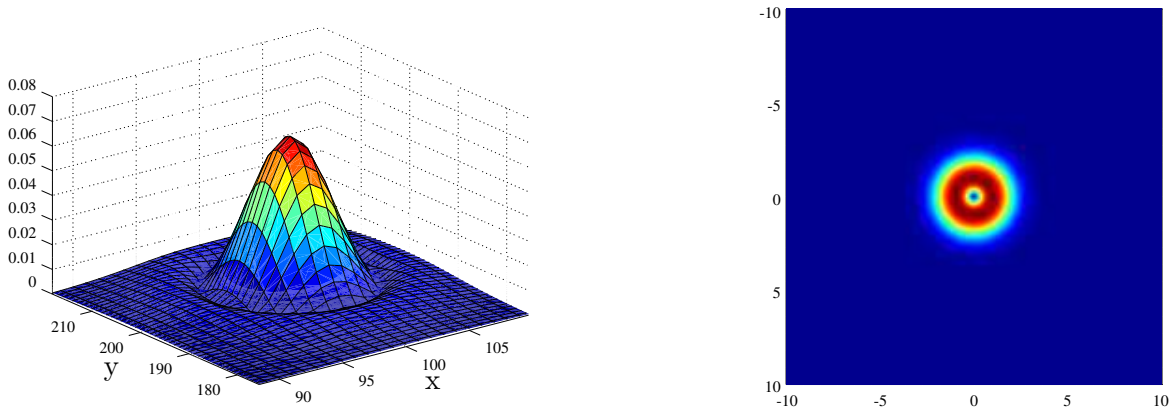
7.2.1 Exemplo com *wavelet* invariante na rotação.

Para exemplificar, na Figura 7.8, tem-se uma imagem que será analisada com a transformada *wavelet* usando novamente a toolbox YAWTb. O código está na página 173. A toolbox define a *wavelet* do chapéu mexicano no domínio de *Fourier* como:

$$\hat{\psi}(\boldsymbol{\xi}) = -2\pi|\boldsymbol{\xi}|^\beta \exp\left(-\frac{(\sigma_x\,\xi_x)^2 + (\sigma_y\,\xi_y)^2}{2}\right) \tag{7.8}$$

em que β é chamado de ordem e foi usado o valor padrão da toolbox de $\beta = 2$ e, para os valores σ_x e σ_y, também foram usados os valores padrões de $\sigma_x = \sigma_y = 1$ que quando iguais tornam a transformada invariante na rotação. A *wavelet* usada, no domínio do espaço, está na Figura 7.7.

Uma coisa que deve ser notada é que como a *wavelet* é invariante na rotação, o valor usado para o ângulo é irrelevante. Para ilustrar, tem-se a representação de escala-ângulo na Figura 7.9. Nesta mesma representação, percebe-se a qual a força de cada escala na transformada bidimensional.



a) b)

Figura 7.7: Figura a: *Wavelet* do chapéu mexicano em duas dimensões. Figura b: Suporte da *wavelet* do chapéu mexicano no domínio de *Fourier* .

Em seguida, na Figura 7.10, observa-se a transformada em sua representação de posição, com valores de escala de 2 a 10. Observa-se a facilidade que a transformada *wavelet* possui em identificar mudanças bruscas na imagem (os contornos, como a mudança entre claro e escuro no meio da imagem, produzem máximos de energia no escalograma bidimensional na transformada).

Figura 7.8: Imagem que será analisada

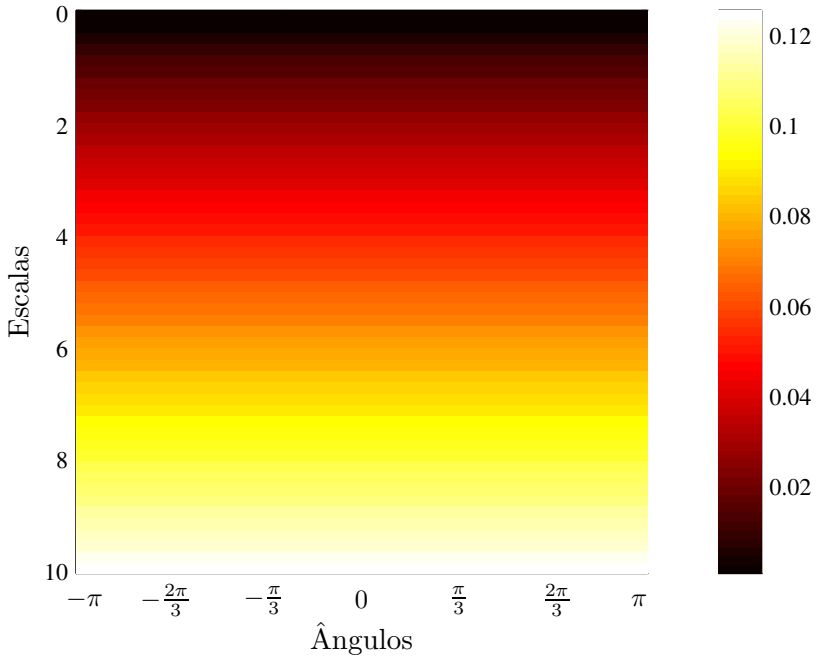


Figura 7.9: Representação em escala-ângulo da CWT da imagem com *wavelet* isotrópica do chapéu mexicano.

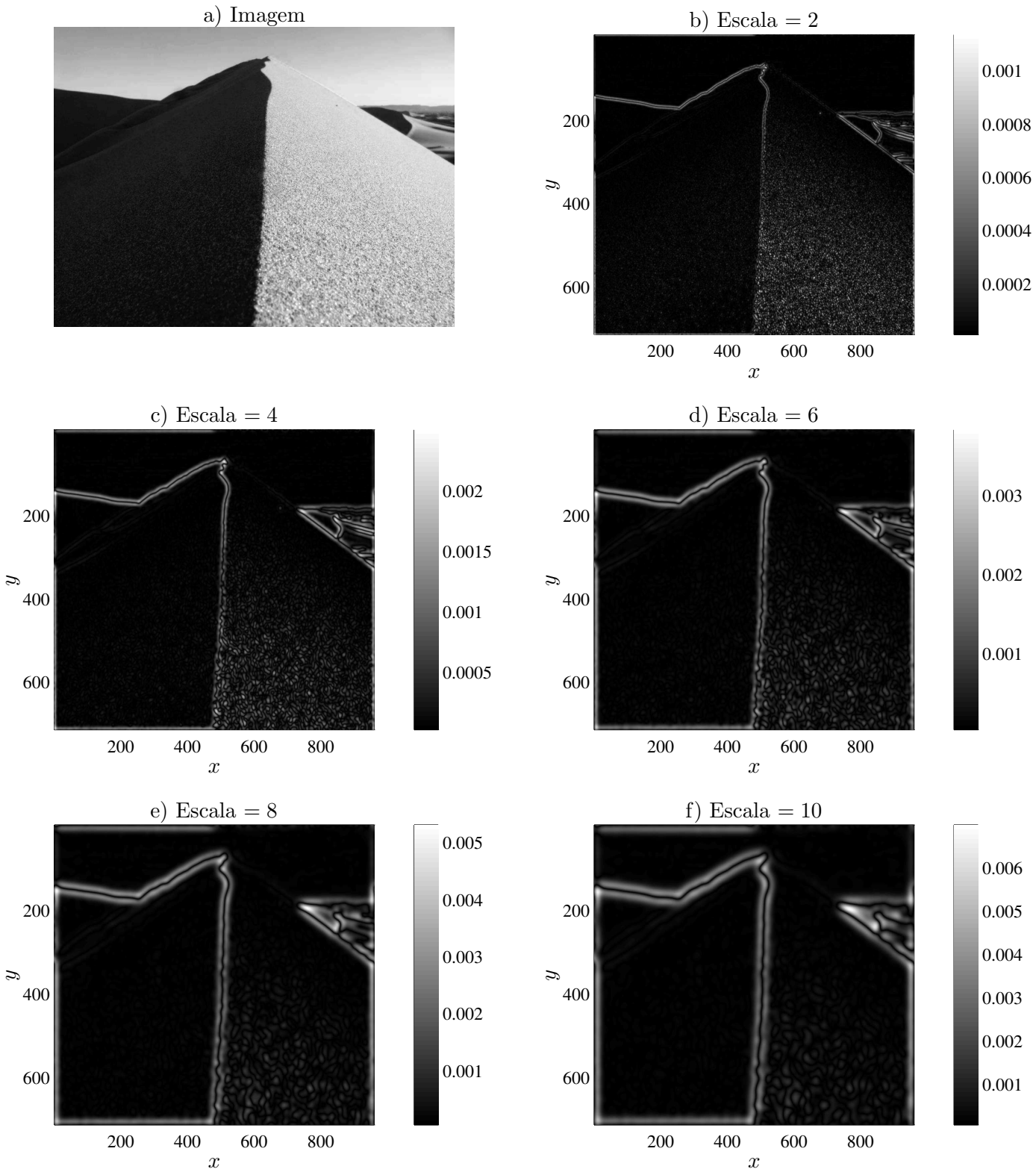


Figura 7.10: Figura a: Imagem analisada. Figuras b até f: Representações de posição da CWT da imagem com chapéu mexicano para escalas de 2 até 10.

Este tipo de *wavelet* do chapéu mexicano é dito estar em sua versão isotrópica. É possível também de se usar uma versão anisotrópica (varia com a rotação), mas ela possui pouco uso na prática. Em geral, *wavelets* direcionais são usadas para detectar direção, mas a *wavelet* do chapéu mexicano não a consegue detectar.

7.2.2 *Wavelets* direcionais

Um problema comum em processamento de imagens é a detecção de bordas. Se é necessária a detecção de propriedades orientadas em uma imagem, uma *wavelet* que possua seletividade direcional é necessária. Uma função *wavelet* é dita direcional se o seu suporte efetivo (ou compacto) no domínio de frequências espaciais $\boldsymbol{\xi}$ está contida em um cone com ápice na origem do plano (x, y).

A *wavelet* de Morlet bidimensional pode ser definida no espaço de *Fourier* como:

$$\hat{\psi}(\boldsymbol{\xi}) = e^{-\frac{\sigma^2}{2}\left((\xi_x-\xi_0)^2+(\epsilon\xi_y)^2\right)} \tag{7.9}$$

em que $\xi_0 \cdot \sigma > 5.5$ é uma condição razoável de admissibilidade. Analisando o suporte da *wavelet* do chapéu mexicano na Figura 7.7 no domínio de *Fourier* , observa-se que ela não satisfaz o critério de direcionalidade (não é possível desenhar um cone com centro na origem que contenha todo o suporta da *wavelet*). Por outro lado, na Figura 7.11, observa-se a *wavelet* de Morlet no domínio de *Fourier* com $\xi_0 = 2$, $\sigma = 5.5$ e dois valores para ϵ. Pode ser observado que seu suporte é centrado em ξ_0, é totalmente contido em tal cone de centro na origem, e ainda, quanto menor o ϵ, mais "afiado" é o suporte, o que a torna ainda mais seletiva na direção.

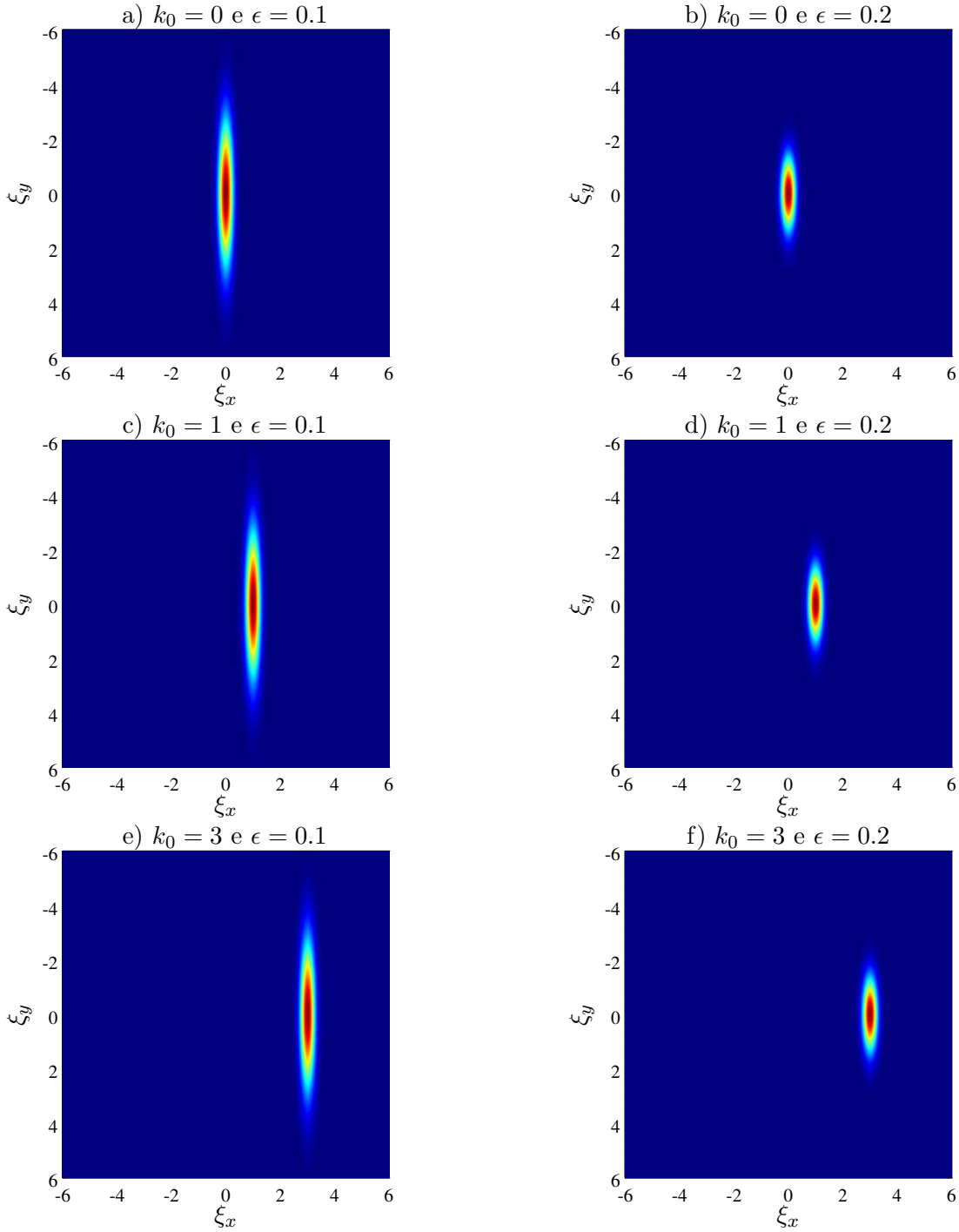


Figura 7.11: Suporte da *wavelet* de Morlet no domínio de *Fourier* para vários valores de ϵ e k_0. Note que, para $k_0 = 0$, a *wavelet* não só não é direcional (como pode ser observado na imagem, pois está localizada na origem) como também não satisfaz a condição de admissibilidade.

A *wavelet* de Morlet então pode ser usada para detectar direcionalidade. Deve ser notado também que algumas outras *wavelets* que não satisfazem este critério podem também apresentar seletividade na direção. Este critério é um critério suficiente, mas não necessário. Usando esta *wavelet* para analisar a mesma imagem, este efeito pode ser percebido. Na Figura 7.12, tem-se a representação em escala-ângulo da transformada com a *wavelet* de Morlet da Figura 7.8.

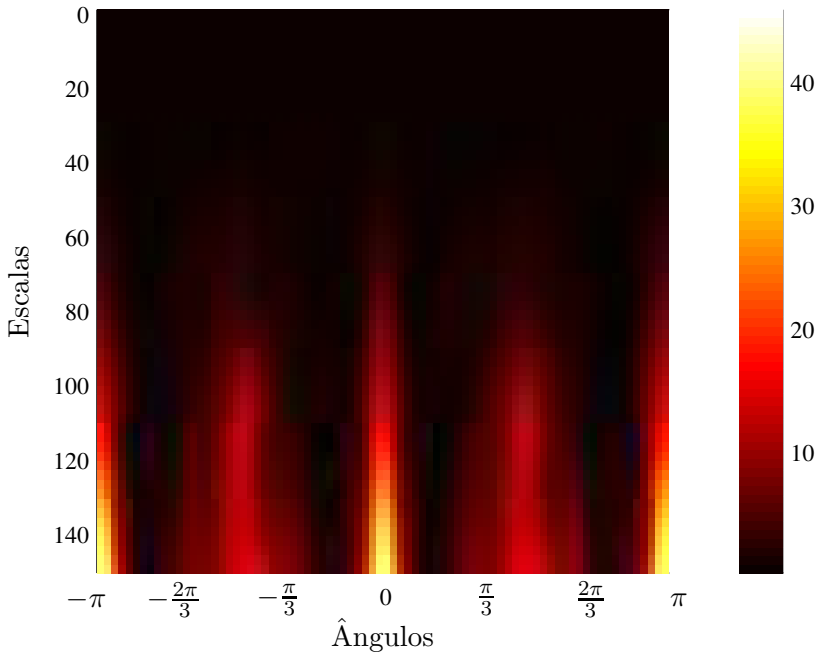


Figura 7.12: Representação de escala-ângulo da CWT com *wavelet* de Morlet.

Observa-se uma concentração de energia para os ângulos aproximadamente iguais a 0, $\pi/3$ e $-\pi/3$. Analisando-se a representação de posição para estes 3 ângulos diferentes na Figura, observa-se o efeito da direcionalidade. Tem-se a seguir a representação de posição com escalas variando de 10 até 50 para 3 ângulos diferentes, $\theta = -\frac{\pi}{3}$ na Figura 7.13, $\theta = 0$ na Figura 7.14, e $\theta = +\frac{\pi}{3}$ na Figura 7.15. Além disso, observa-se que o aumento da escala faz a transformada ignorar os pequenos formatos da areia. Na Figura 7.16 vemos também a plotagem das fases de algumas destas representações de posição das transformadas contínuas com *wavelet* de Morlet. Observa-se que a fase acompanha o ângulo da *wavelet* .

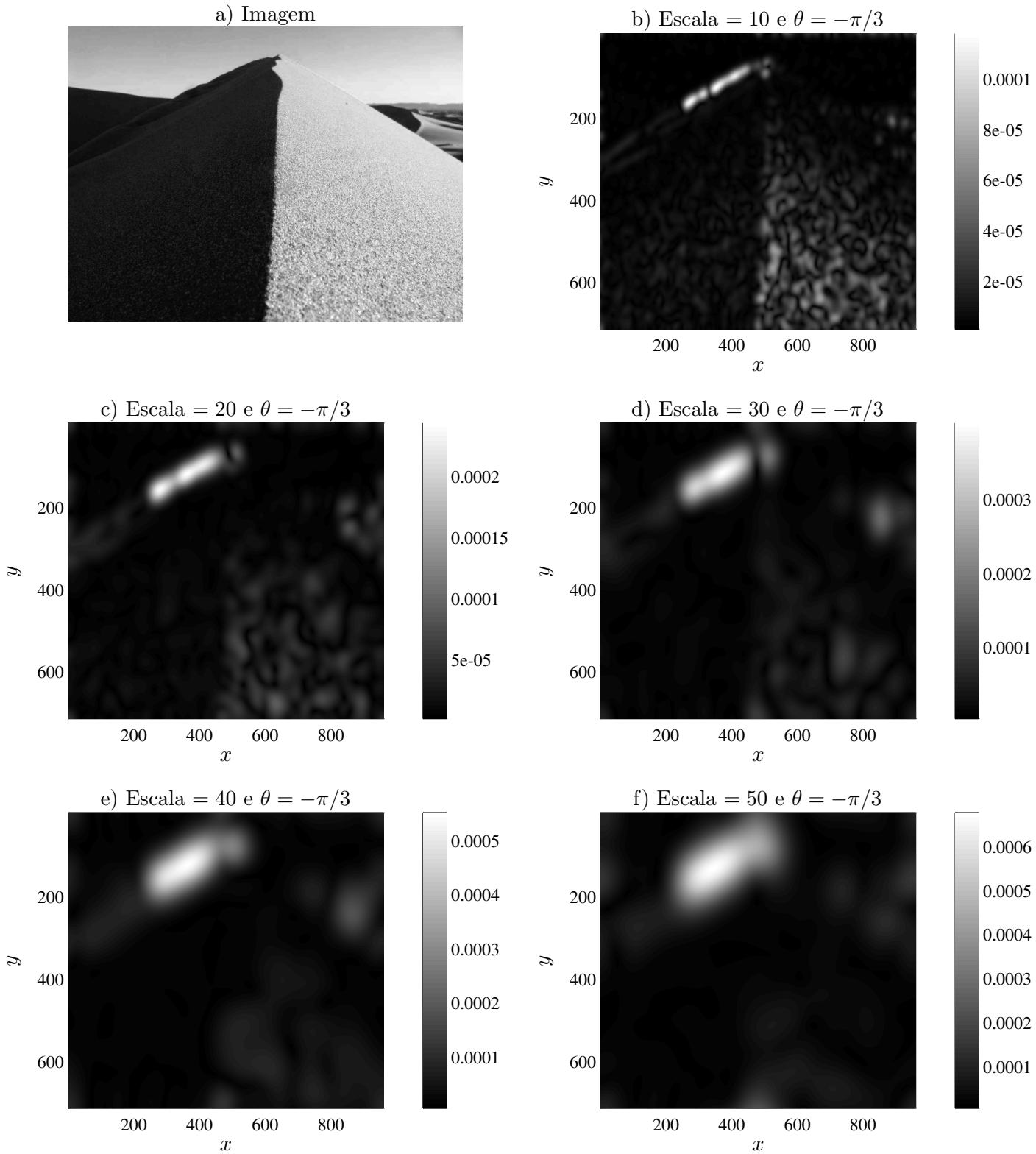


Figura 7.13: Comparação da representação de posição da CWT bidimensional com *wavelet* de Morlet para $\theta = -\pi/3$. Figura a: Imagem analisada. Figuras b até f: escalas variando de 10 até 50.

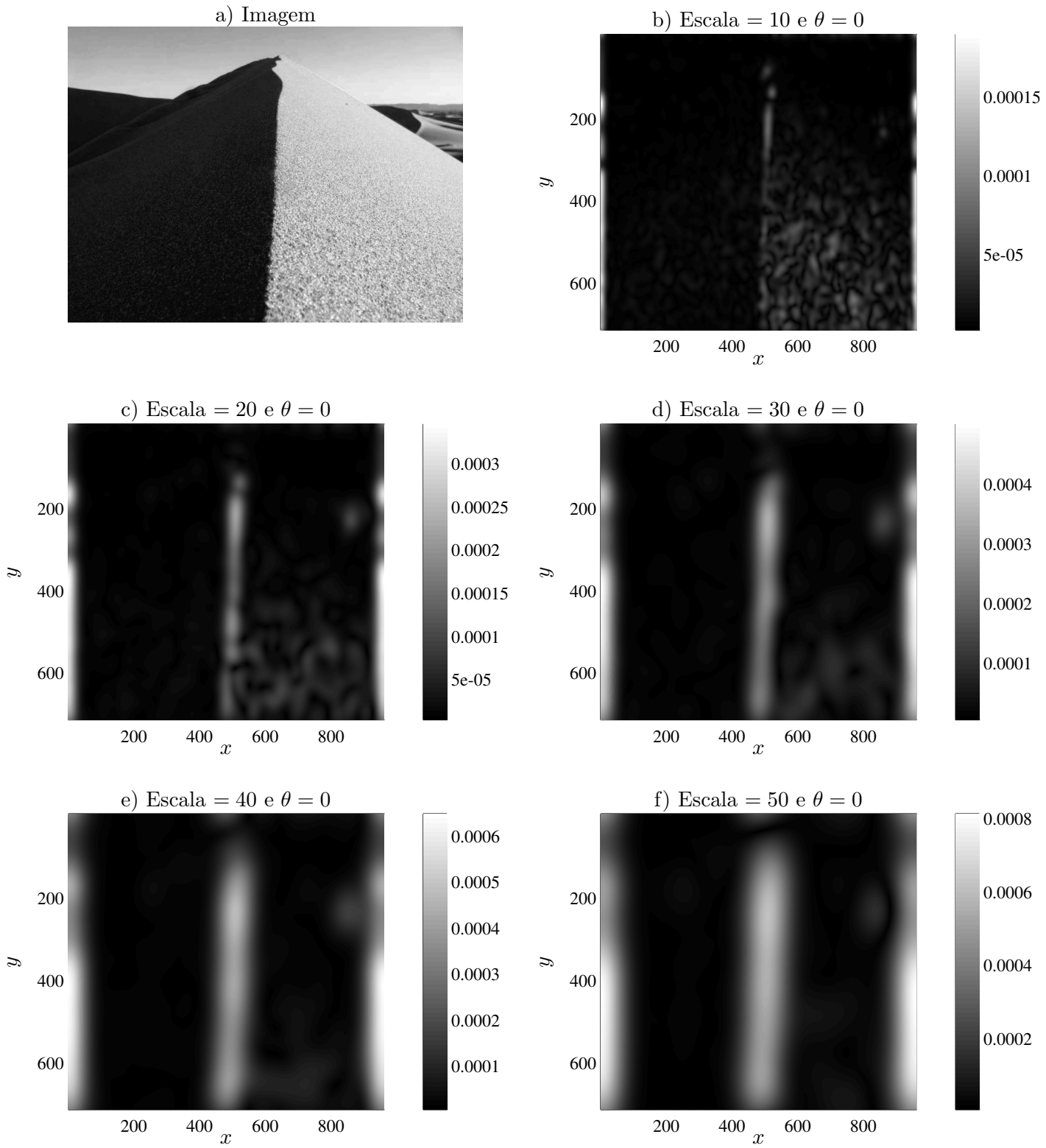


Figura 7.14: Comparação da representação de posição da CWT bidimensional com *wavelet* de Morlet para $\theta = 0$. Figura a: Imagem analisada. Figuras b até f: escalas variando de 10 até 50.

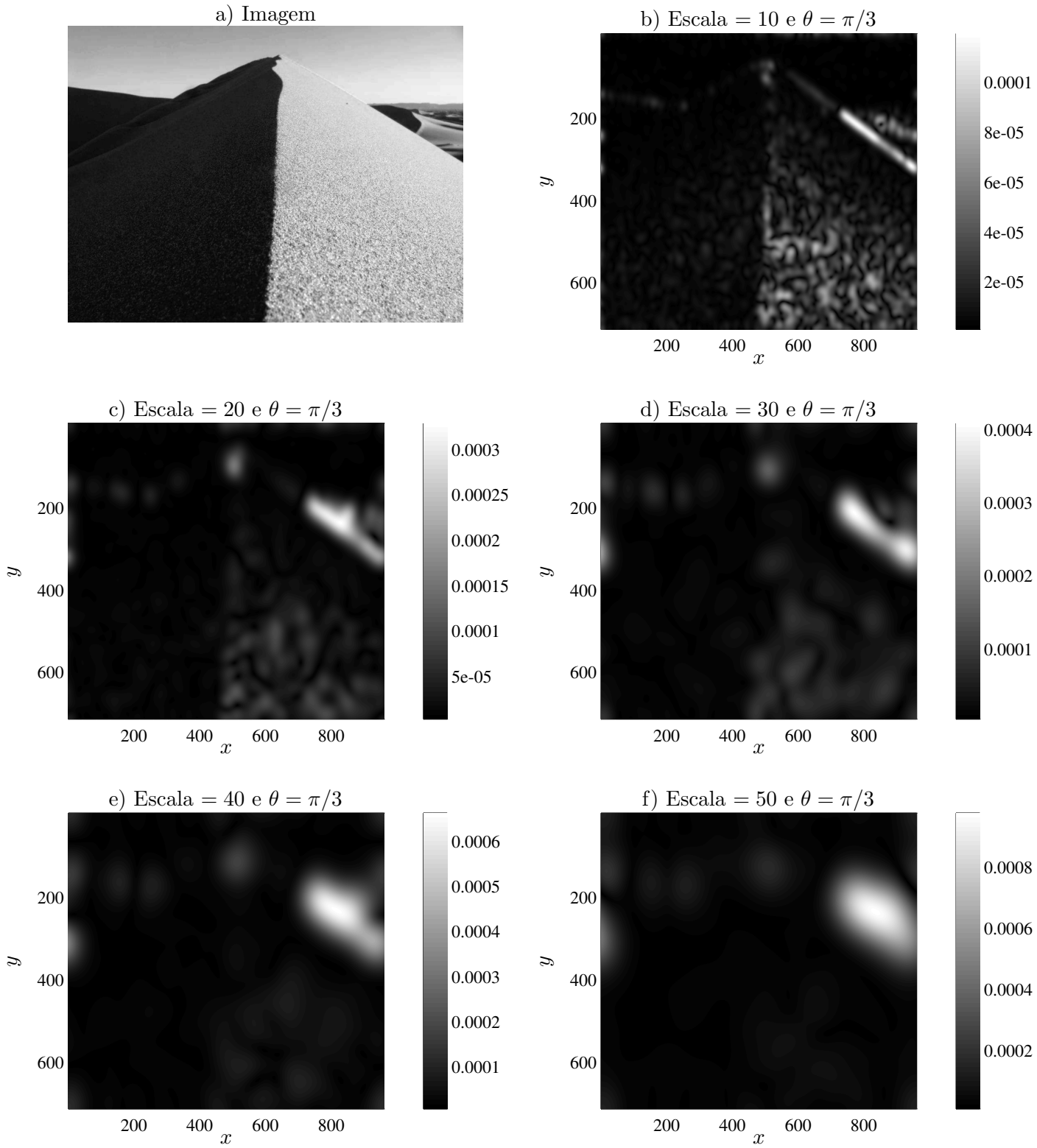


Figura 7.15: Comparação da representação de posição da CWT bidimensional com *wavelet* de Morlet para $\theta = \pi/3$. Figura a: Imagem analisada. Figuras b até f: escalas variando de 10 até 50.

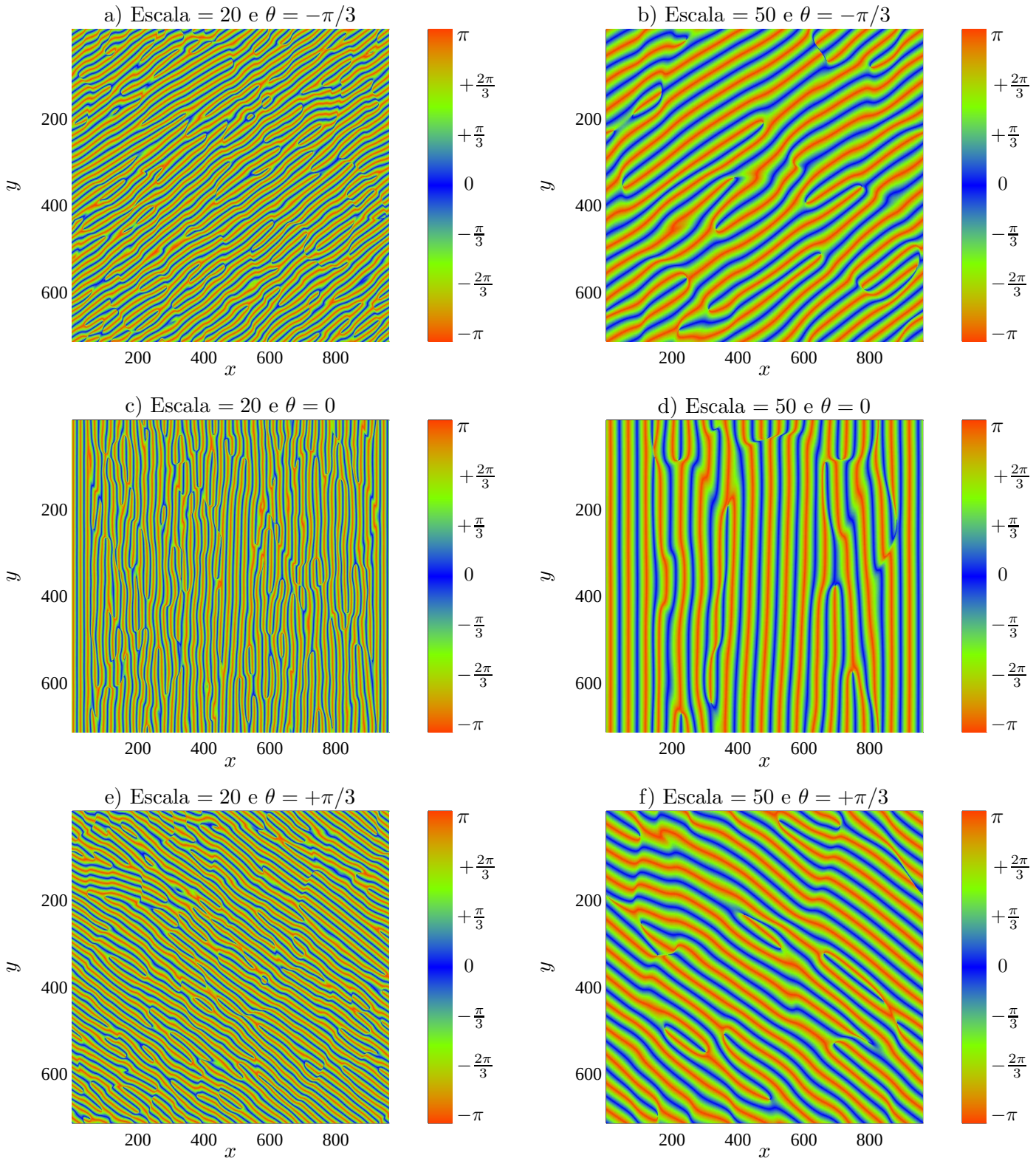


Figura 7.16: Comparação da fase da representação de posição da CWT bidimensional com *wavelet* de Morlet.

Para dar outro exemplo, serão calculadas as transformadas com a *wavelet* de Morlet de uma imagem que possui várias linhas, algumas que passam pela figura inteira e outras que param na metade.

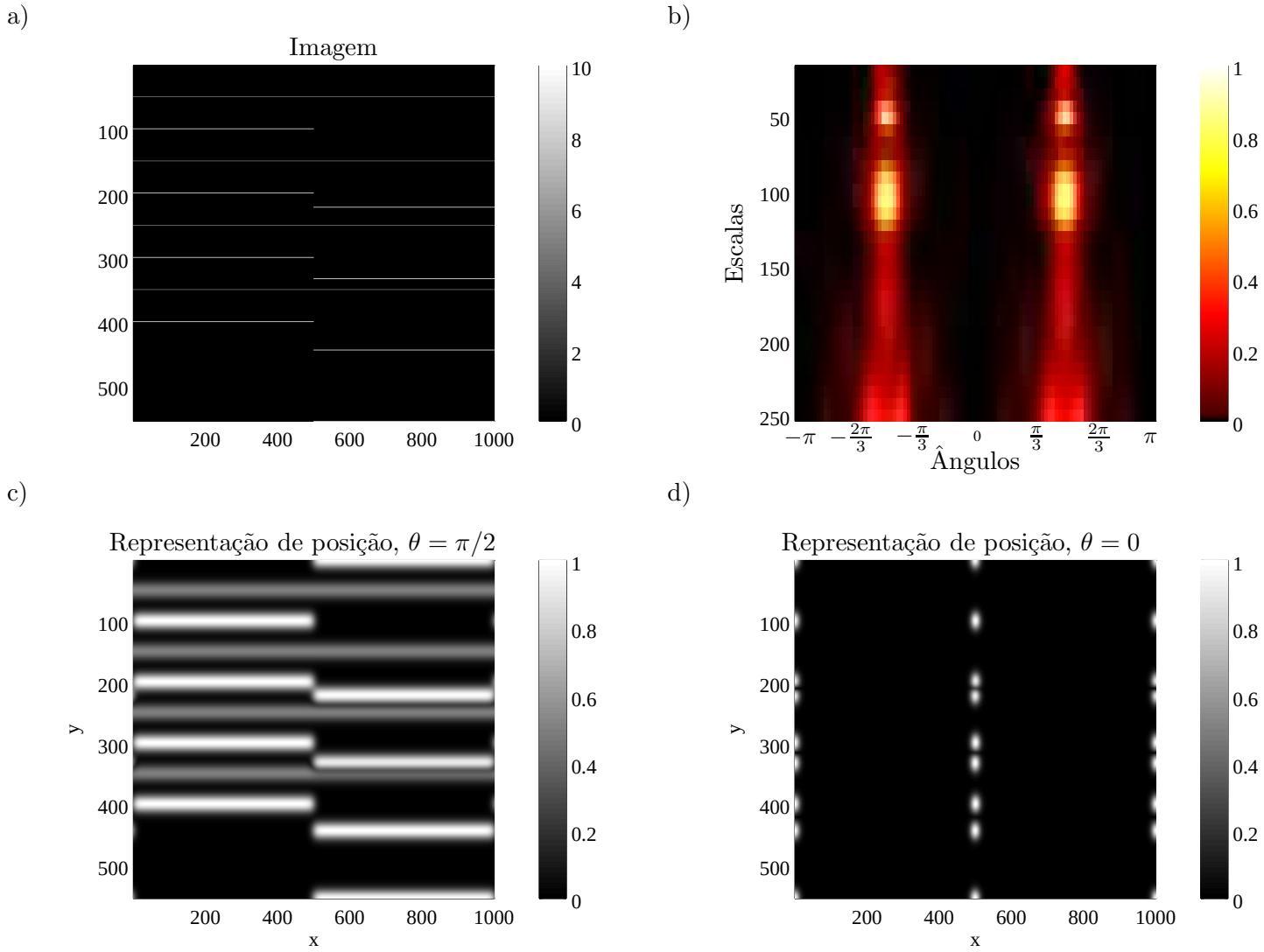


Figura 7.17: Figura a: imagem com várias linhas. Figura b: média da representação de escala-ângulo da CWT com *wavelet* de Morlet. Figuras c e d: CWT em representação de posição, com $\theta = \pi/2$ e $\theta = 0$ respectivamente.

A imagem da Figura 7.17 a) possui linhas com apenas um pixel de largura. Usando a transformada *wavelet* em sua média da representação de escala-ângulo, observa-se na Figura 7.17 b) que a energia está presente principalmente no ângulo de 90° (ou $\pi/2$ rad). Nas Figuras 7.17 c) e d) observa-se as transformadas em representação de posição para os ângulos de $\theta = \pi/2$ e 0 respectivamente. Para $\theta = \pi/2$ observa-se o destaque que a transformada conseguiu dar às linhas. Para uma imagem muito maior, isto tornaria possível a visualização de linhas finas que anteriormente não seria possível. Para $\theta = 0$ ele também detectou algumas formas, efeito que se dá no momento da descontinuidade de algumas linhas no meio da imagem, e efeito de borda no fim e início das linhas (efeito que acontece por que o algoritmo considera a imagem como sendo periódica, propriedade que ela não possui de fato).

7.3 Testes com as imagens do experimento LASCO

Para demonstrar a capacidade da transformada *wavelet* de destacar contornos de estruturas presentes em uma imagem, é apresentada a análise da imagem presente na Figura 7.18(a). Esta imagem foi obtida no experimento *The Large Angle Spectrometric COronagraph* (LASCO), um dos 11 instrumentos do satélite *Solar and Heliospheric Observatory* (SOHO), que usa anteparos (coronógrafos) que bloqueiam a luz intensa do centro do sol para estudar a coroa solar e objetos com menos intensidade [1]. Conforme discutido em [1], este instrumento é desenvolvido para medir a coroa solar de 1.1 a 32 raios solares, e as questões essenciais de física solar estudadas pelo LASCO são [1]: Como a coroa é aquecida? Quando e como o vento solar é acelerado? O que causam fenômenos coronais transientes, e qual o papel deles no desenvolvimento evolucionário de padrões coronais em larga escala? Na Figura 7.18 a), observa-se uma imagem do sol que demonstra emissão de massa coronal, e na Figura 7.18 b), está plotado o módulo da representação de escala-ângulo da imagem. Como a imagem possui uma grande círculo negro no centro (onde fica o anteparo), além de possuir efeito de borda na transformada, esta representação de escala-angulo é feita de um modo diferente. Ao invés de se usar simplesmente a função do pacote YAWTB, usa-se um algoritmo que calculou várias representações de posição, zerando o centro e as bordas de cada uma destas representações, e então a média de toda esta representação é usada para gerar o valor de um dos pontos da representação de escala-ângulo. No resultado final, observa-se que a figura possui emissão em uma grande gama de direções, como é visto pelo variação quase nula que a representação de escala-ângulo demonstrou no ângulo.

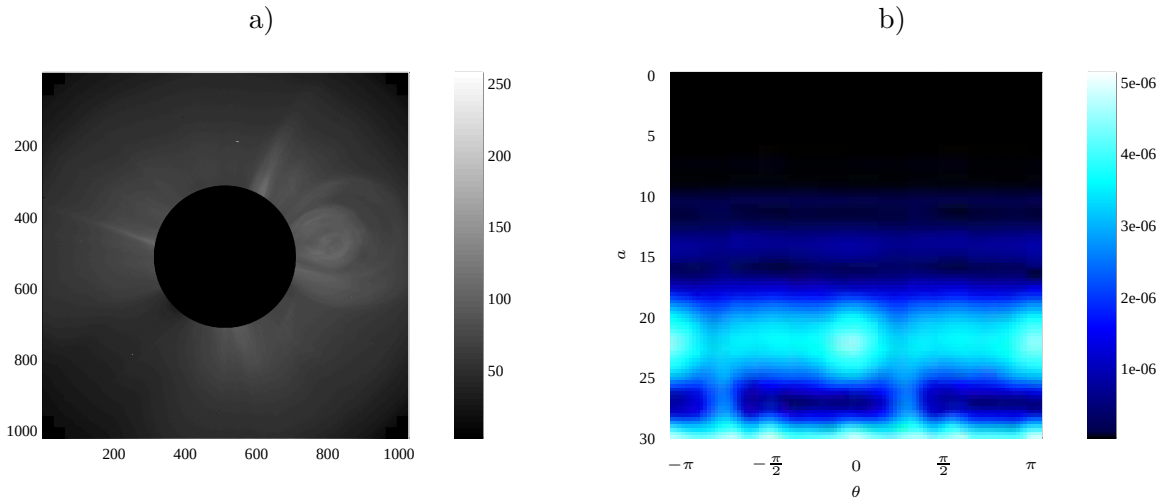


Figura 7.18: Imagem do satélite LASCO dia 06/10/2001, contendo um anteparo para exclusão do disco solar (a) e CWT com *wavelet* de Morlet para escalas variando entre 1 e 30, em uma representação ângulo-escala.

Na Figura 7.19 estão plotadas a representação de posição da CWT da imagem para escalas de 1, 3, 5 e 7. Estas imagens possuem muitas informações irrelevantes, como pontos fortes que se destacam sem mostrar informação alguma e estruturas durante toda a imagem. Para gerar uma melhor visualização, estes elementos foram retirados na imagem de escala 7 gerando a representação da Figura 7.20, onde observa-se com maior facilidade as estruturas de interesse em destaque.

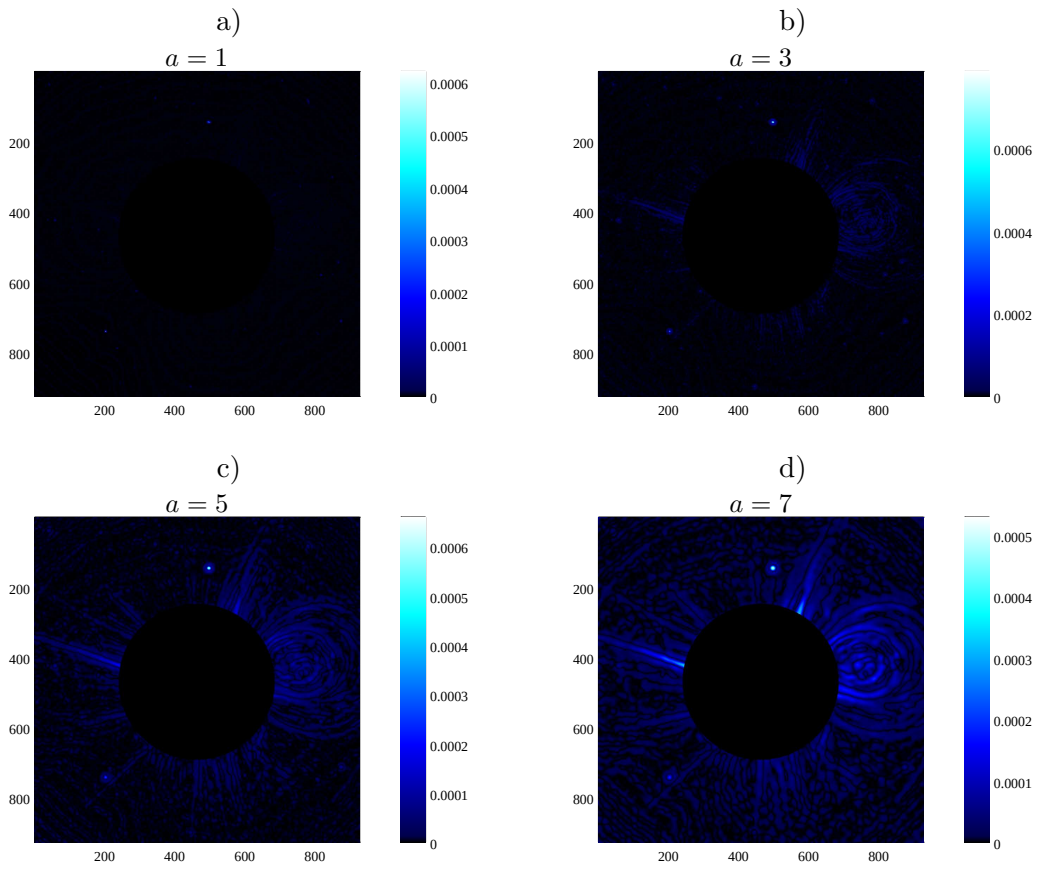


Figura 7.19: Representação de posição da imagem para escalas variando de 1 até 7.

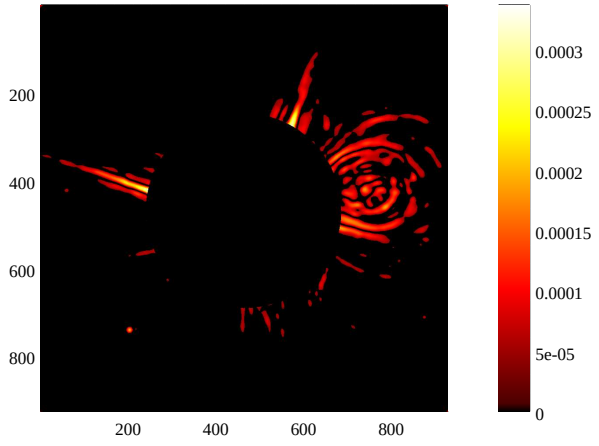


Figura 7.20: Representação de posição para escala 7 destacando os elementos de interesse.

Capítulo 8

Análise de contornos com CWT

Um importante problema em processamento de imagens é a caracterização de formas, que encontra aplicações em diversas áreas da ciência e computação. Em geral, existem dois modos de se analisar a forma de uma imagem:

- Por região, uma método intrinsecamente 2D e que cuida de formas primitivas em 2D;
- Por contorno, que lida com imagens bidimensionais usando operações e representações unidimensionais.

O segundo modo é chamado de *caracterização por contorno* e neste capítulo será apresentado um método de análise do contorno usando a transformada *wavelet* contínua, descrito em [2].

8.1 Extração da imagem

Neste capítulo, será demonstrado o método usado neste trabalho para a extração das formas em cada imagem, e subsequentemente, seu contorno. A imagem da Figura 8.1 será usada para exemplificar os passos.

8.2 Segmentação

O processo de segmentação da imagem é a divisão da mesma em partes "parecidas". Cada região da imagem é substituída por uma versão suavizada da imagem nesta mesma região, de modo que o computador consegue mais facilmente identificar cada parte da figura.

No GNU/Octave, a função `im2bw` do pacote "image" é um função que separa a imagem em regiões, resultando em uma imagem binária (isto é, o sinal bidimensional de saída só possui dois valores, "alto" ou "baixo"). Esta função funciona separando partes da imagem cuja iluminação média é acima de um certo valor limite (threshold) dado, que varia de 0 até 1 (0% até 100%). Na Figura 8.2 observa-se a imagem passada por esta função para 3 valores diferentes do limite.

A Figura 8.2 d) foi escolhida para ser usada no exemplo.

8.3 Extração de um formato

Para extrair o formato de machado na Figura 8.2, será usada a função `bwlabel`, também do pacote de imagens. Lembrando-se que a imagem segmentada é totalmente binária, este novo comando implementa um algoritmo que faz com que todas os pixels com valor alto em uma dada região receba um novo valor distinto das outras regiões. Logo, os pixels do machado terão um certo valor, enquanto os pixels do martelo terão outro valor, etc... . Na Figura 8.3 é demonstrada esta nova figura, chamada de "imagem

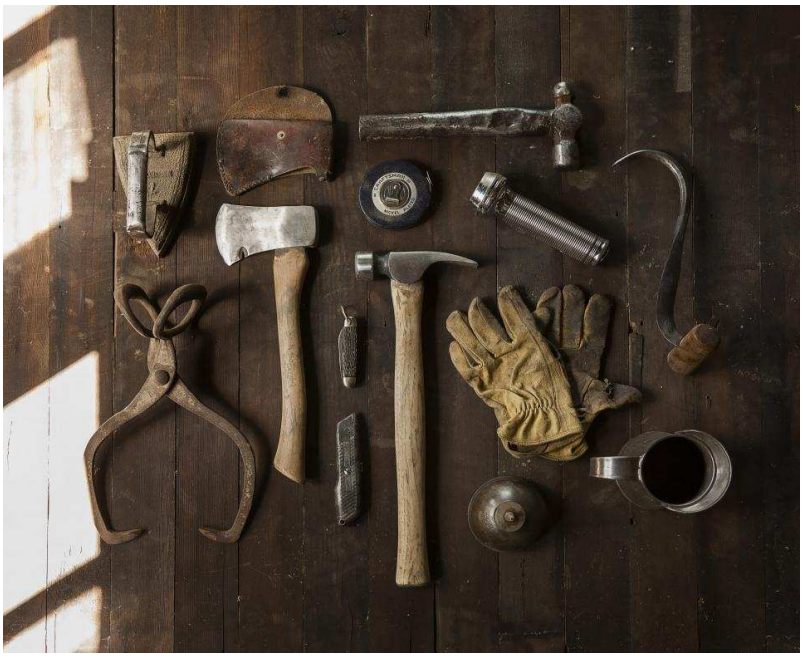


Figura 8.1: Imagem que será usada nos exemplos. Fonte: `http://stocksnap.io`

rotulada" em comparação com a figura binária. A partir deste estágio, para extrair o formato desejado, é só criar uma nova imagem binária com valor alto apenas para os pixels que possuem o mesmo valor que um dado ponto arbitrário pertencente a imagem rotulada.

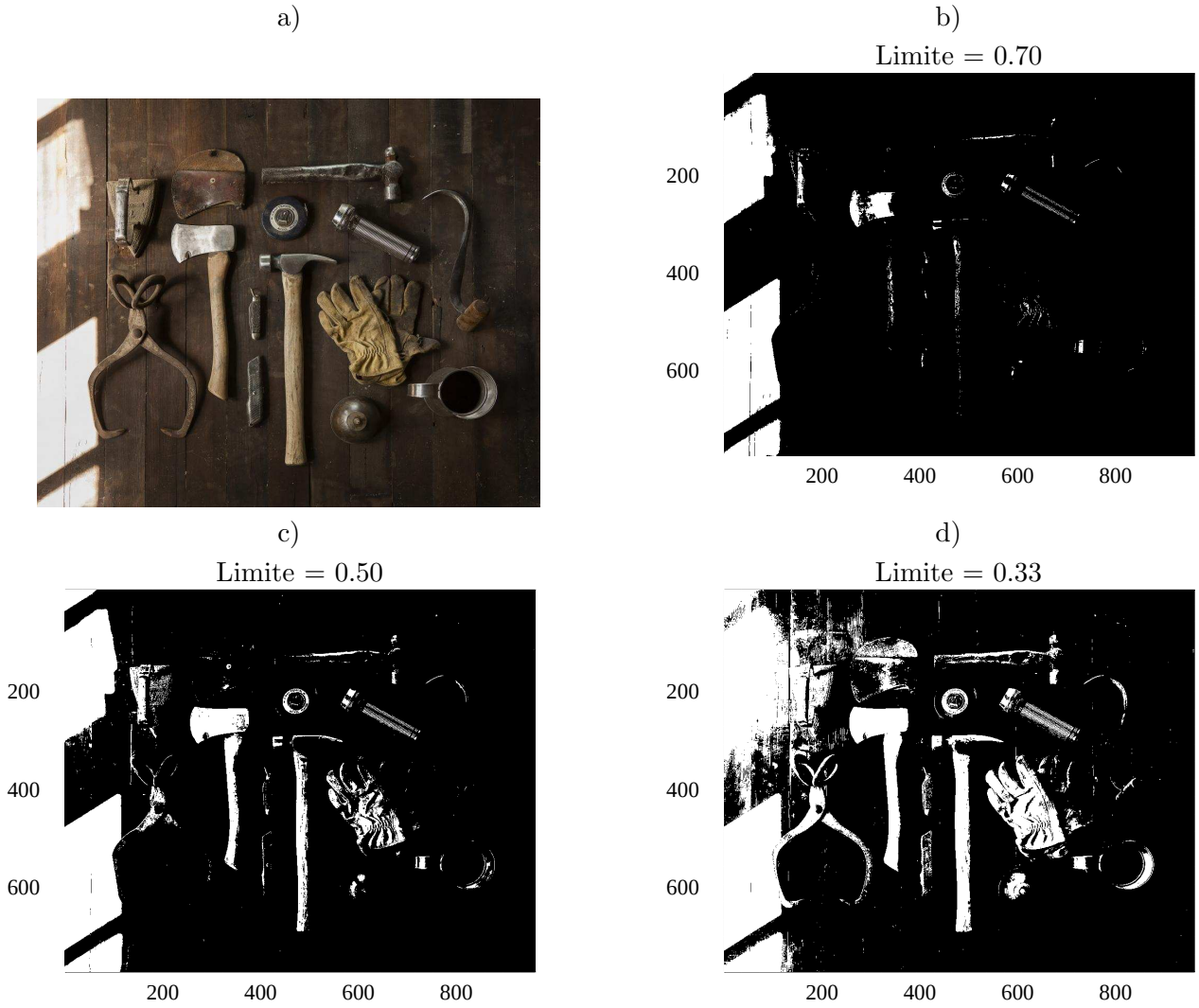


Figura 8.2: Figura a: Imagem original. Figuras b, c e d: Imagem passada pelo comando `im2bw` com para limites de 0.70, 0.50 e 0.33, respectivamente.

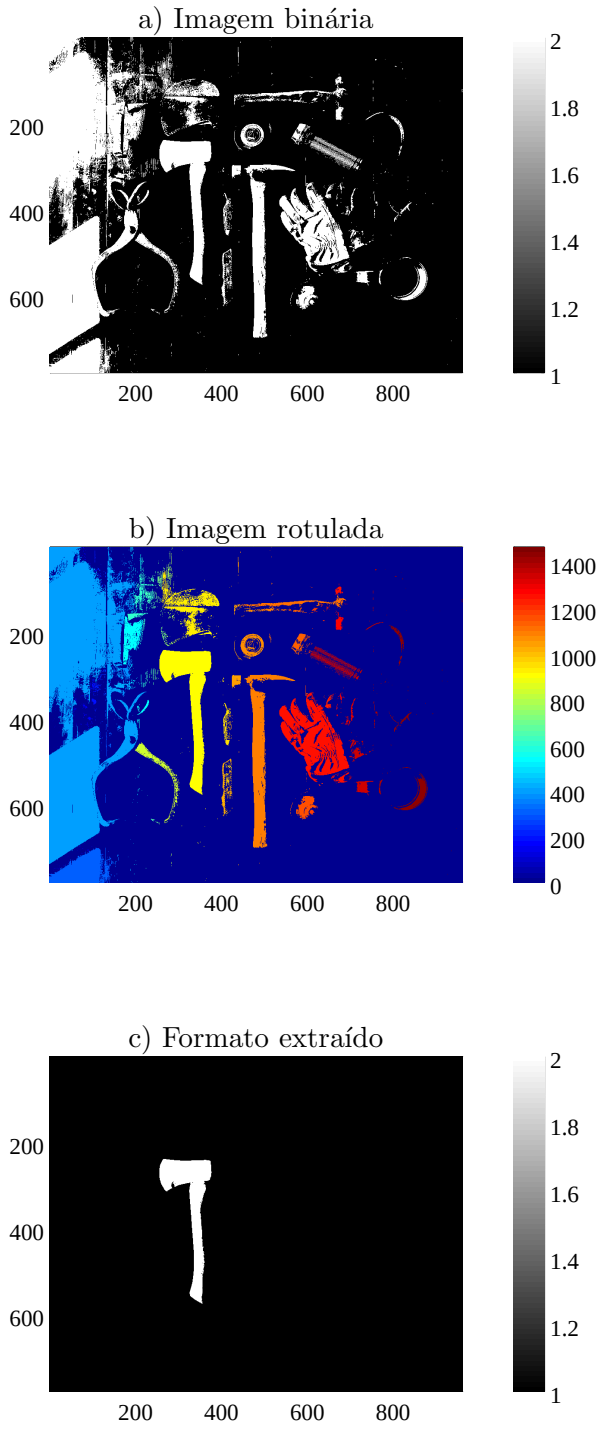


Figura 8.3: Figura a: Imagem binária. Figuras b: Imagem rotulada. Figura c: Formato extraído.

8.4 Extração do contorno

Existem vários modos de se extrair o contorno da imagem após se obter a forma desejada, e neste será feita a extração com a função `bwboundaries` do pacote de imagem. A função pega uma imagem binária e retorna contornos da figura desejada, contendo ou não buracos dentro delas.

O modo como o contorno, chamado aqui de λ, é analisado no artigo [2], é criando uma função complexa $\lambda[n] = x[n] + \imath y[n]$, em que x e y carregam os valores em ambos os eixos do n-ésimo ponto do contorno. A Figura 8.4 mostra a extração do contorno, e a Figura 8.5 mostra o gráfico de $x[n]$ e $y[n]$.

Figura 8.4: Extração do contorno com a função `bwboundaries`.

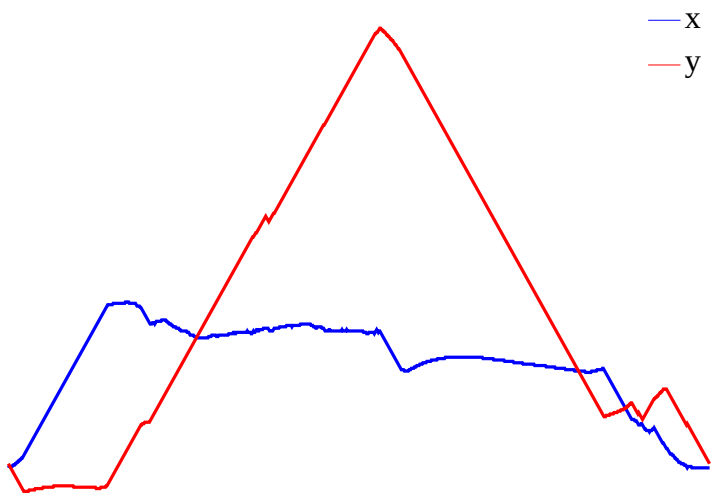
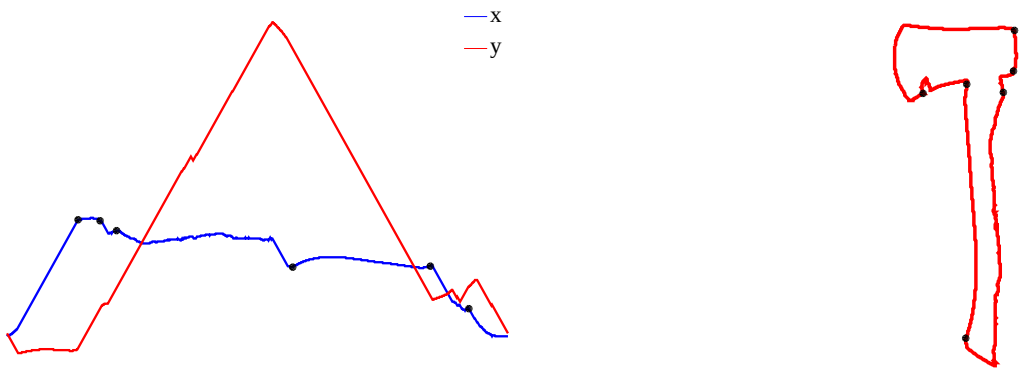


Figura 8.5: Em azul, a componente x, e em vermelho, a componente y.

Na Figura 8.6, são destacados alguns pontos de mudança da derivada das funções x e y, a) e b) respectivamente. Pode ser observado que estes pontos em particular guardam informação sobre os pontos de mudança brusca da figura.

a)



b)

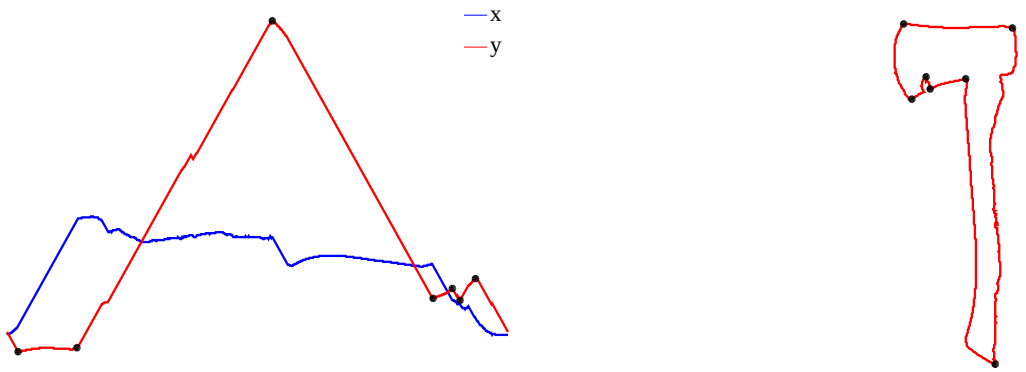


Figura 8.6: Figuras a e b: pontos de destaque das funções x e y, respectivamente.

8.5 CWT da função λ

Após ter sido determinada a função complexa que representa o contorno da imagem, a CWT é calculada. Esta transformada é chamada de "Representação-W". Na Figura 8.7 são mostradas as transformadas com as *wavelets* de Morlet e do chapéu mexicano. Observe que a fase da transformada com o chapéu mexicano

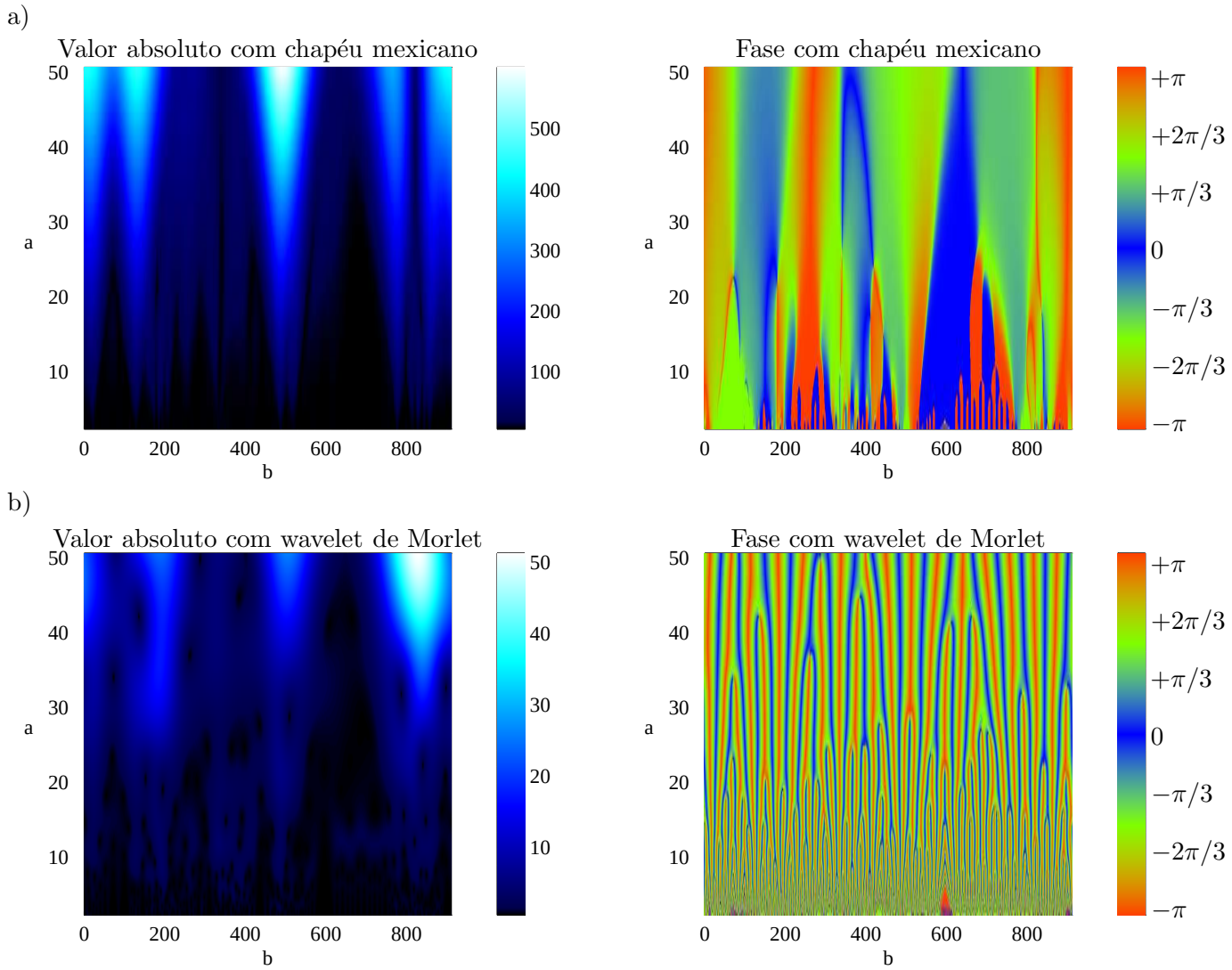


Figura 8.7: Transformadas *wavelet* contínuas com *wavelet* do chapéu mexicano (Figura a) e *wavelet* de Morlet (Figura b).

é mostrada pois, apesar de a *wavelet* ser real, desta vez o sinal que teve a sua CWT calculada não é. Pode ser notado também que a *wavelet* com chapéu mexicano não possui a capacidade da *wavelet* de Morlet de identificar a fase, porém o seu módulo possui estruturas cônicas que se destacam mais facilmente.

Singularidades na derivada de um sinal, em geral, formam estruturas similares às estas estruturas cônicas, como discutido em [2]. A *wavelet* do chapéu mexicano será então escolhida para dar continuidade a este exemplo, visto que singularidades na derivada do sinal podem justamente indicar um mudança na direção do contorno, que denotam a presença de cantos na estrutura da imagem estudada. Na Figura 8.8, vemos a plotagem do valor absoluto da CWT para 4 valores de escala. Analisar alguma escala arbitrária pode ser um método usado para se encontrar pontos interessantes na imagem, porém não muito eficiente. Nas Figura 8.9 a) e b) vemos a análise dos picos do módulo na escala $a = 13$, e observa-se bastante ruído ainda presente de escalas menores. Na Figura 8.9 c) e d) é feita uma filtragem antes da análise dos picos,

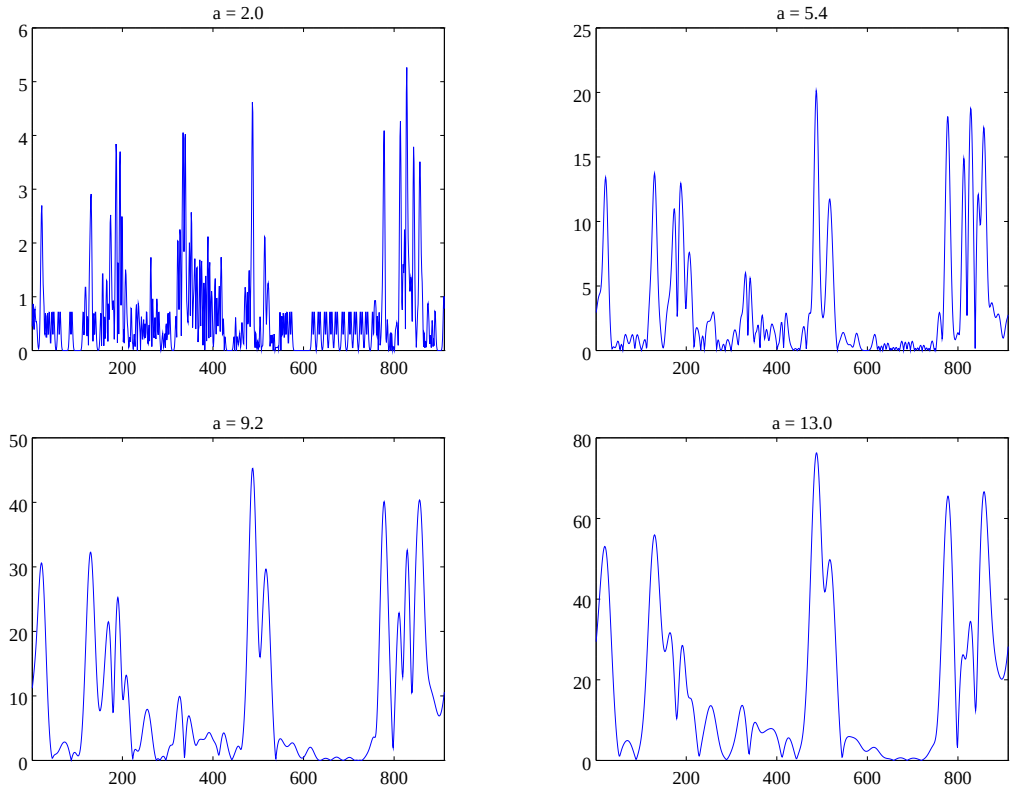


Figura 8.8: Módulo da representação-W para algumas escalas.

detectando assim só pontos realmente interessantes.

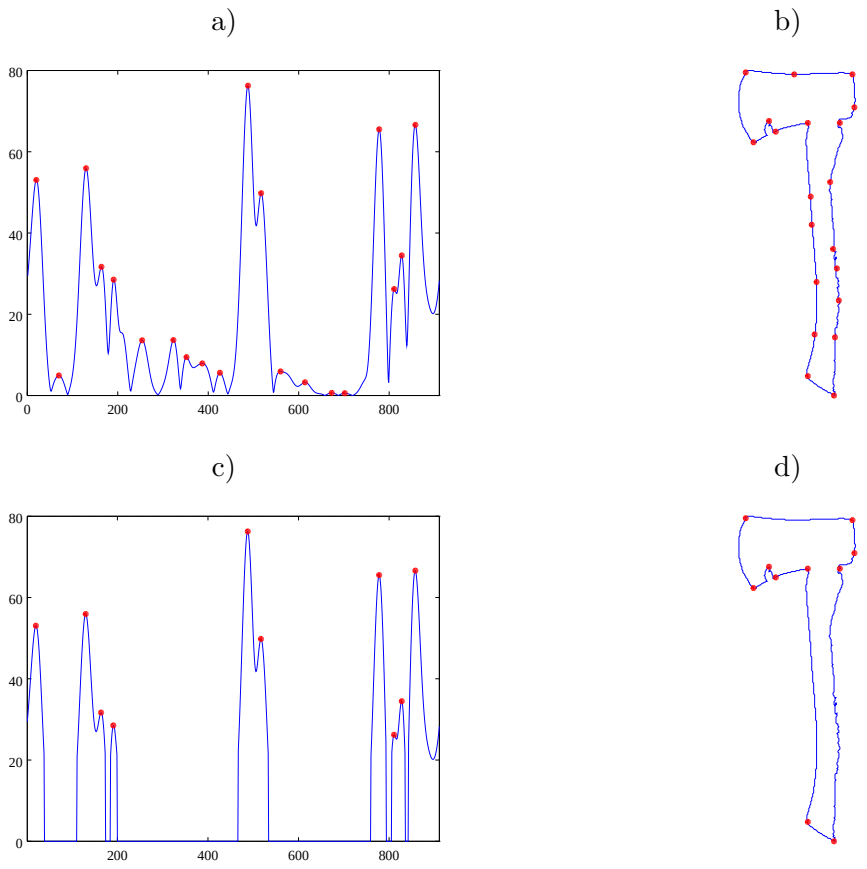


Figura 8.9: Figura a) e b): picos do módulo da escala $a = 13$ e pontos correspondentes no contorno. Figuras c) e d): picos para o módulo filtrado da escala, e pontos correspondentes.

Um modo mais eficiente e com resultados melhores é encontrar as chamadas linhas de máxima da representação-W. Estes picos nas funções para cada escala geram linhas que se estendem desde $a = 0$ até as maiores escalas. Estas linhas podem ser filtradas utilizando-se de algum critério, e a plotagem delas é chamado de "esqueleto" da representação-W, ilustrado na Figura 8.10. Utilizando-se deste esqueleto para encontrar os pontos de interesse é uma maneira mais interessante, porque ao fazer isto, pontos são encontrados que possuem máxima em todas as escalas.

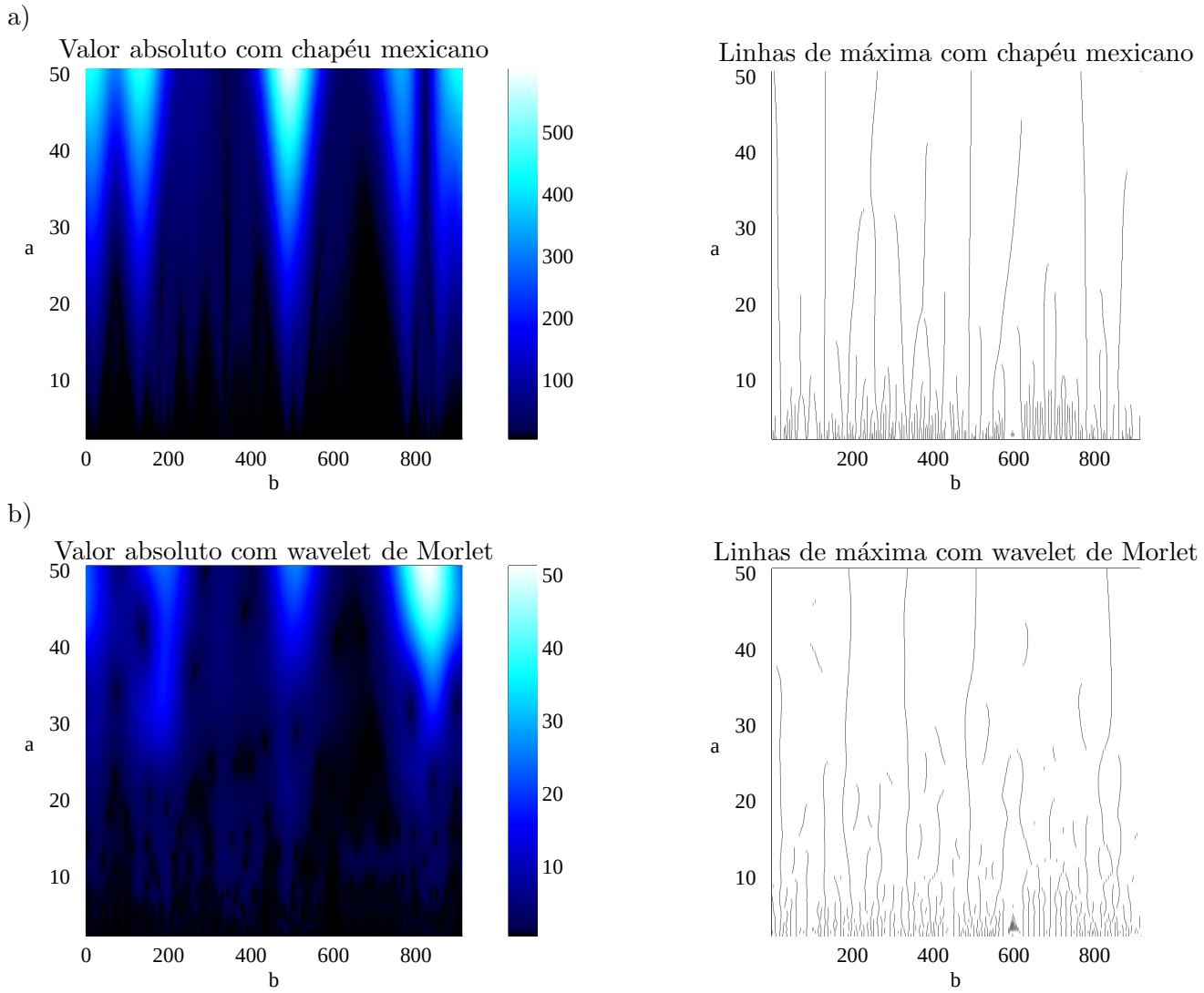


Figura 8.10: Transformadas *wavelet* contínuas e seus esqueletos de linhas verticais, com *wavelet* do chapéu mexicano (Figura a) e *wavelet* de Morlet (Figura b).

Pode-se notar também que mesmo a *wavelet* de Morlet, que na comparação inicial não demonstrou facilmente muitas estruturas, mostra claramente as estruturas mais importantes em suas linhas. Vários critérios para a seleção das linhas podem ser usados, como o comprimento de cada linha, ou a integral da transformada ao decorrer desta linha. Entretanto, para qualquer que seja o critério, algum método para a detecção de cada linha separadamente deve ser usado.

8.6 Exemplo

A imagem do Sol da Figura 8.11 será usada como exemplo de aplicação. Na Figura 8.12, vemos o a

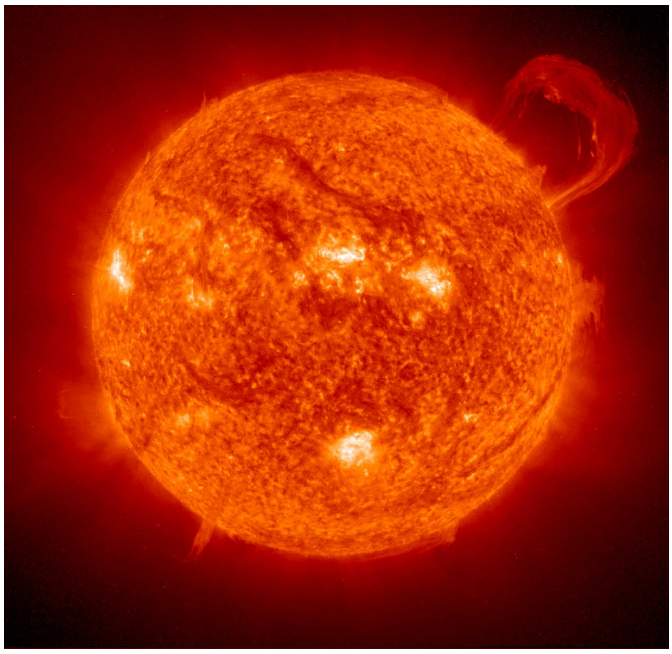


Figura 8.11: Sol e emissão solar. Fonte: satélite SOHO.

formato extraído com o limite = 0.13 (valor escolhido arbitrariamente) do lado do seu contorno. Deste contorno foi feita a transformada *wavelet* contínua para se obter a representação-W com as *wavelets* de Morlet, chapéu mexicano e Haar, com seus valores absolutos todos representados, respectivamente, nas Figuras 8.13 a, b e c.

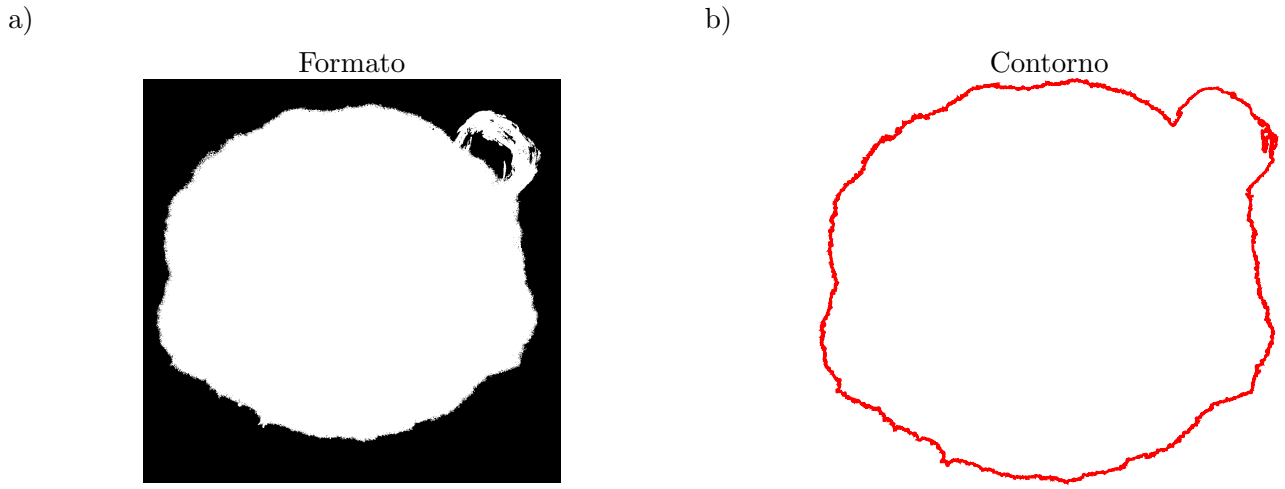


Figura 8.12: Contorno extraído da fotografia do Sol.

Com as 3 *wavelets* diferentes, pode ser observado em torno do ponto 7000 do contorno, algo de diferente ocorre na imagem. Na *wavelet* de Haar, ocorre um vale bem perceptível próximo a esta região, enquanto com as *wavelets* de Morlet e chapéu mexicano, estruturas com altos valores ocorrem. Na Figura 8.14 dois

pontos são destacados, mostrando que a região realmente corresponde à emissão presente na imagem do Sol no momento da fotografia.

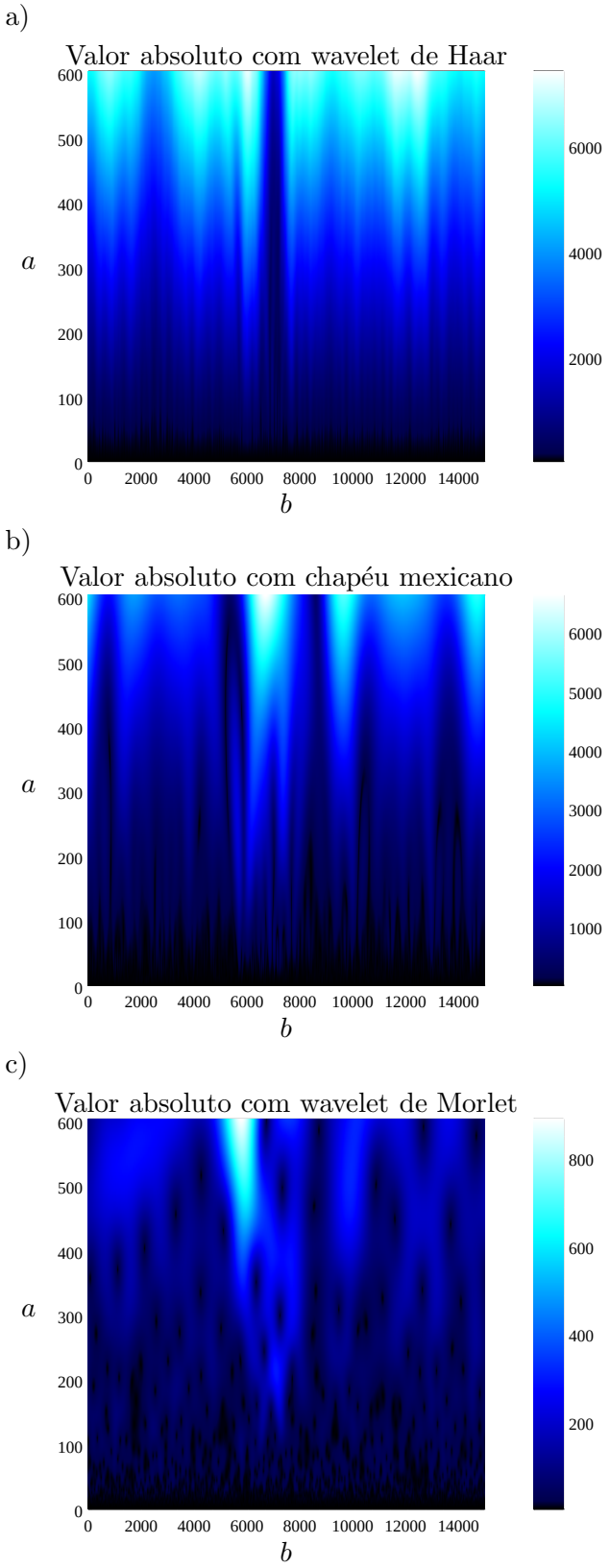


Figura 8.13: Representação-W do contorno do Sol com as *wavelets* de Haar (Figura a), chapéu mexicano (Figura b), e Morlet (Figura c).

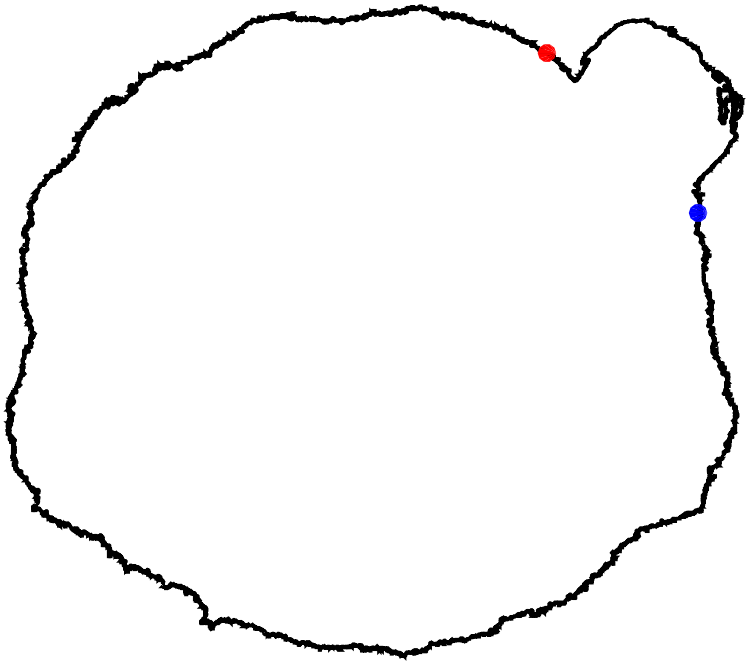


Figura 8.14: Pontos 5500 (em vermelho) e 8000 (em azul), que coincidem aproximadamente com o intervalo da emissão presente nas amostras do contorno.

Capítulo 9

Referências Bibliográficas

[1] Lasco experiment. `http://lasco-www.nrl.navy.mil`. Acessado em 11/05/2015.

[2] J.-P. Antoine, D. Barache, R. M. Cesar, Jr., and L. da Fontoura Costa. Shape characterization with the wavelet transform. *Signal Process.*, 62(3):265–290, November 1997.

[3] J.-P. Antoine and R. Murenzi. Two-dimensional directional wavelets and the scale-angle representation. *Signal Process.*, 52(3):259–281, August 1996.

[4] Evandro Bernardes, Margarete Oliveira Domingues, and Odim Mendes. Transformada wavelet contínua bidimensional: variabilidade direcional e em escala. *DINCON*, 2015.

[5] J. Castilho, M. Domingues, A. Pagamisse, and O. Mendes. *Introdução ao Mundo das Wavelets*, volume 62 of *Notas em Matemática Aplicada*. SBMAC, São Carlos, SP, 2012.

[6] M. Farge. Imagerie scientifique: Choix des palettes de couleurs pour la visualisation des champs scalaires bidimensionnels. *L'Aéronautique et l'Astronautique*, 1990.

[7] Matthias Geissbuehler and Theo Lasser. How to display data by color schemes compatible with red-green color perception deficiencies. *Optics Express*, 21(8):9862–9874, 2013.

[8] A. Grossmann and J. Morlet. Decomposition of hardy functions into square integrable wavelets of constant shape. *SIAM Journal on Mathematical Analysis*, 15(4):723–736, 1984.

[9] Barbara Burke Hubbard. *The World According to Wavelets: The Story of a Mathematical Technique in the Making*. A. K. Peters, Ltd., Natick, MA, USA, second edition edition, 1996.

Apêndice A

Derivação transformada contínua de *Fourier*

A.1 Série de *Fourier*

A.1.1 Ortogonalidade das funções trigonométricas

Duas funções $f(x)$ e $g(x)$ são chamadas ortogonais em um intervalo $a < x < b$ se, para uma dada função-peso $w(x)$, elas satisfazem a seguinte propriedade:

$$\langle f, g \rangle_w = \int_a^b f(x) \cdot g(t) \cdot w(x) dx = 0$$

sempre que $f(x) \neq g(x)$, em que $\langle f, g \rangle_w$ é chamado de produto escalar das funções $f(x)$ e $g(x)$ no espaço w, e $w(x) = 1$ para casos simples.

As funções trigonométricas $\text{sen}(t)$ e $\cos(t)$ possuem esta propriedade, que será demonstrada nesta seção.

Caso 1: $\cos \cdot \cos$ **e** $\text{sen} \cdot \text{sen}$**,** $n \neq m$

Supondo duas funções $\cos\left(\frac{2\pi n}{T}\right)$ e $\cos\left(\frac{2\pi m}{T}\right)$ em que $n \neq m$ são inteiros, calculando seu produto interno no intervalo T:

$$\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi m}{T}x\right) \right\rangle = \int_0^T \cos\left(\frac{2\pi n}{T}x\right) \cdot \cos\left(\frac{2\pi m}{T}x\right) dx$$

usando-se a propriedade

$$\cos(a)\cos(b) = \frac{\cos(a+b) + \cos(a-b)}{2}$$

tem-se:

$$\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi m}{T}x\right) \right\rangle = \frac{1}{2}\int_0^T \cos\left(\frac{2\pi(n+m)}{T}x\right) + \cos\left(\frac{2\pi(n-m)}{T}x\right) dx$$

$$= \frac{T}{4\pi(n+m)}\text{sen}\left(\frac{2\pi(n+m)}{T}x\right)\Bigg|_0^T + \frac{T}{4\pi(n-m)}\text{sen}\left(\frac{2\pi(n-m)}{T}x\right)\Bigg|_0^T$$

como tanto n quanto m são inteiros, ambos os senos são nulos.

$$\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi n}{T}x\right) \right\rangle = 0 \tag{A.1}$$

Isto prova que o produto escalar de cossenos diferentes com mesmo período é nulo, e o mesmo pode ser mostrado facilmente para dois senos.

Caso 2: $\cos \cdot \cos$ **e** $\text{sen} \cdot \text{sen}$**,** $n = m$

Supondo agora que $n = m$, não pode ser usada a mesma integral deduzida acima porque teria-se uma divisão por zero. Será usada desta vez a propriedade:

$$\cos^2(a) = \frac{1 + \cos(2a)}{2}$$

da seguinte forma:

$$\begin{aligned}\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi n}{T}x\right)\right\rangle &= \int_0^T \cos^2\left(\frac{2\pi n}{T}x\right) dx \\ &= \frac{1}{2}\int_0^T 1 + \cos\left(\frac{4\pi n}{T}x\right) dx \\ &= \frac{x}{2}\bigg|_0^T + \frac{T}{8\pi n}\text{sen}\left(\frac{2\pi n}{T}x\right)\bigg|_0^T\end{aligned}$$

Novamente, o seno se anula, e tem-se que:

$$\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi n}{T}x\right)\right\rangle = \frac{T}{2} \tag{A.2}$$

Da mesma forma, pode ser mostrado que:

$$\left\langle \text{sen}\left(\frac{2\pi n}{T}x\right), \text{sen}\left(\frac{2\pi n}{T}x\right)\right\rangle = \frac{T}{2}$$

Caso 3: $\cos \cdot \text{sen}$

$$\text{sen}(a)\cos(b) = \frac{\text{sen}(a-b) - \text{sen}(a+b)}{2}$$

Resultando novamente em duas integrais que se anulam. Logo, o produto internos destas funções trigonométricas vale $\frac{T}{2}$ se elas são iguais, e zero se são diferentes.

A.2 Coeficientes da série de *Fourier*

Para se conseguir deduzir a fórmula para as constantes a_n e b_n, utiliza-se duas propriedades das funções trigonométricas: o fato de elas possuírem média nula, e o fato de elas serem ortogonais entre sí.

Da primeira propriedade, tem-se que:

$$\int_0^T \cos\left(\frac{2\pi n}{T}x\right) dx = \int_0^T \text{sen}\left(\frac{2\pi n}{T}x\right) dx = 0 \tag{A.3}$$

Logo, integrando então a equação 1.1 em um período completo, obtem-se:

$$\int_0^T f(x)dx = \int_0^T \left\{ \frac{a_0}{2} + \sum_{n=1}^{+\infty} \left[a_n \cdot \cos\left(\frac{2\pi n}{T}x\right) + b_n \cdot \operatorname{sen}\left(\frac{2\pi n}{T}x\right) \right] \right\} dx$$

$$\int_0^T f(x)dx = \frac{a_0}{2} \int_0^T 1dx + \sum_{n=1}^{+\infty} \left[a_n \cdot \int_0^T \cos\left(\frac{2\pi n}{T}x\right) dx + b_n \cdot \int_0^T \operatorname{sen}\left(\frac{2\pi n}{T}x\right) dx \right]$$

$$\int_0^T f(x)dx = \frac{a_0}{2} \cdot T + \sum_{n=1}^{+\infty} [a_n \cdot 0 + b_n \cdot 0]$$

logo,

$$a_0 = \frac{2}{T} \int_0^T f(x)dx \tag{A.4}$$

Em seguida, é usado o fato de que elas são ortogonais. A propriedade da ortogonalidade das funções trigonométricas garante que:

$$\left\langle \cos\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi m}{T}x\right) \right\rangle = 0$$

e:

$$\left\langle \operatorname{sen}\left(\frac{2\pi n}{T}x\right), \operatorname{sen}\left(\frac{2\pi m}{T}x\right) \right\rangle = 0$$

Caso $n \neq m$, e :

$$\left\langle \operatorname{sen}\left(\frac{2\pi n}{T}x\right), \cos\left(\frac{2\pi m}{T}x\right) \right\rangle = 0$$

Independente da igualdade ou não de n e m. Calculando então o produto interno dos dois lados da equação 1.1 com uma função seno ou cosseno, todos os componentes da somatória (menos um correspondente a $n = m$) são cancelados, o que possibilita então encontrar uma fórmula explícita para os termos a_n e b_n. Utilizando-se dessa técnica, a série de Fourier então pode ser descrita inteiramente como:

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{+\infty} \left[a_n \cdot \cos\left(\frac{2\pi n}{T}x\right) + b_n \cdot \operatorname{sen}\left(\frac{2\pi n}{T}x\right) \right],$$

$$a_0 = \frac{2}{T} \int_0^T f(x)dx$$

$$a_n = \frac{2}{T} \int_0^T f(x) \cos\left(\frac{2\pi n}{T}x\right) dx$$

$$b_n = \frac{2}{T} \int_0^T f(x) \operatorname{sen}\left(\frac{2\pi n}{T}x\right) dx$$

A.3 Transformada de Fourier

Nesta seção, será derivada uma forma alternativa (porém equivalente) da série de Fourier, e ela será usada para a derivação subsequente da transformada logo em seguida.

A.3.1 Forma Exponencial

Pode-se observar claramente das definições dos coeficientes a_n e b_n que:

$$a_{-n} = a_n \quad e \quad b_{-n} = -b_n \tag{A.5}$$

Propriedade esta que será usada em breve.

Para transformar a função de uma soma de senóides para uma soma de exponenciais complexas, usa-se o fato de que:

$$\cos(t) = \frac{e^{\imath t} + e^{-\imath t}}{2}$$

e

$$\operatorname{sen}(t) = \frac{e^{\imath t} - e^{-\imath t}}{2\imath}$$

Jogando estas fórmulas na definição da série de *Fourier* , obtem-se:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{+\infty} \left[a_n \cdot \cos\left(\imath \frac{2\pi n}{T} t\right) + b_n \cdot \operatorname{sen}\left(\imath \frac{2\pi n}{T} t\right) \right]$$

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{+\infty} \left[a_n \cdot \frac{e^{\imath \frac{2\pi n}{T} t} + e^{-\frac{\imath 2\pi n}{T} t}}{2} + b_n \cdot \frac{e^{\imath \frac{2\pi n}{T} t} - e^{-\imath \frac{2\pi n}{T} t}}{2\imath} \right]$$

$$f(t) = \frac{a_0}{2} + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n (e^{\imath \frac{2\pi n}{T} t} + e^{-\imath \frac{2\pi n}{T} t}) - \imath b_n (e^{\imath \frac{2\pi n}{T} t} - e^{-\imath \frac{2\pi n}{T} t}) \right]$$

$$f(t) = \frac{a_0}{2} + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t} + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n + \imath b_n \right] e^{-\imath \frac{2\pi n}{T} t}$$

Fazendo em seguida a substituição da variável de somatória n na segunda somatória por $-n$, obtem-se:

$$f(t) = \frac{a_0}{2} + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t} + \frac{1}{2} \sum_{n=-1}^{-\infty} \left[a_{-n} + \imath b_{-n} \right] e^{\imath \frac{2\pi n}{T} t}$$

Usando a propriedade em A.5, tem-se:

$$f(t) = \frac{a_0}{2} + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t} + \frac{1}{2} \sum_{n=-1}^{-\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t}$$

E utilitando-se do fato de que $b_0 = 0$, soma-se o fator $-\imath \frac{b_0}{2}$:

$$f(t) = \frac{1}{2} \left[a_0 - \imath b_0 \right] + \frac{1}{2} \sum_{n=1}^{+\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t} + \frac{1}{2} \sum_{n=-1}^{-\infty} \left[a_n - \imath b_n \right] e^{\imath \frac{2\pi n}{T} t}$$

Deste modo, pode-se juntar tudo em uma única somatória:

$$f(t) = \sum_{n=-\infty}^{+\infty} \frac{\left[a_n - \imath b_n \right]}{2} e^{\imath \frac{2\pi n}{T} t}$$

$$f(t) = \sum_{n=-\infty}^{+\infty} c_n \cdot e^{\imath \frac{2\pi n}{T} t} \tag{A.6}$$

em que

$$c_n = \frac{a_n - \imath b_n}{2} = \frac{1}{T} \int_{-T/2}^{T/2} f(t) \left[\cos\left(\frac{2\pi n}{T} \cdot t\right) - \imath \text{sen}\left(\frac{2\pi n}{T} \cdot t\right) \right] dt$$

$$c_n = \frac{1}{T} \int_{-T/2}^{T/2} f(t) \cdot e^{-\frac{2\pi n}{T} t} dt$$

A.3.2 Definição final da transformada de Fourier

Como a série de *Fourier* só funciona para sinais periódicos, para generalizá-la para sinais gerais, é só supor que a função possui um período infinito. Mas primeiramente, escreve-se a função assim:

$$f(t) = \sum_{n=-\infty}^{+\infty} c_n \cdot e^{\imath \frac{2\pi n}{T} t}$$

$$f(t) = \sum_{n=-\infty}^{+\infty} \left[\frac{1}{T} \int_{-T/2}^{T/2} f(t) \cdot e^{-\frac{2\pi n}{T} t} dt \right] \cdot e^{\imath \frac{2\pi n}{T} t}$$

Chamando a frequência angular das exponenciais de $\imath \frac{2\pi n}{T} = \omega_n$, obtem-se:

$$f(t) = \sum_{n=-\infty}^{+\infty} \left[\frac{1}{T} \int_{-T/2}^{T/2} f(t) \cdot e^{-\imath \omega_n t} dt \right] \cdot e^{\imath \omega_n t}$$

E usando o seguinte passo matemático:

$$\Delta\omega = \omega_{n+1} - \omega_n = \frac{2\pi(n+1)}{T} - \frac{2\pi(n)}{T} = \frac{2\pi}{T}$$

Obtem-se:

$$\frac{2\pi}{T} = \Delta\omega$$

$$\frac{1}{T} = \frac{1}{2\pi} \Delta\omega$$

Sendo assim, pode-se escrever:

$$f(t) = \frac{1}{2\pi} \sum_{n=-\infty}^{+\infty} \left[\int_{-T/2}^{T/2} f(t) \cdot e^{-\imath \omega_n t} dt \right] \cdot e^{\imath \omega_n t} \cdot \Delta\omega$$

Em seguida, tudo o que resta fazer é tender T para infinito:

$$\frac{2\pi}{T} \longrightarrow 0$$

$$\Delta\omega \longrightarrow d\omega$$

$$\sum \longrightarrow \int$$

E a equação se torna:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \left[\int_{-\infty}^{+\infty} f(t) \cdot e^{-\imath\omega t} dt \right] \cdot e^{\imath\omega t} \cdot d\omega$$

Definindo a transformada de *Fourier* da função $f(t)$ como sendo:

$$\hat{f}(\omega) = \int_{-\infty}^{+\infty} f(t) \cdot e^{-\imath\omega t} dt$$

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \hat{f}(\omega) \cdot e^{\imath\omega t} d\omega$$

A forma da transformada usada neste trabalho será, entretanto, a seguinte forma (utilizando-se de uma frequência dada em Hz):

$$\hat{f}(\xi) = \int_{-\infty}^{+\infty} f(t) \cdot e^{-2\pi\imath\xi t} dt \tag{A.7}$$

Que possui inversa definida como:

$$f(t) = \int_{-\infty}^{+\infty} \hat{f}(\xi) \cdot e^{2\pi\imath\xi t} d\xi \tag{A.8}$$

Apêndice B

Derivação da constante de admissibilidade e da CWT inversa

As seguintes propriedades da transformada de *Fourier* serão usadas para a dedução da fórmula:

- Propriedade 1, translação no tempo:

$$\mathcal{F}_t^{\xi}\{f(t-b)\} = e^{+\imath 2\pi\xi b}\hat{f}(\xi) \tag{B.1}$$

- Propriedade 2, dilatação no tempo:

$$\mathcal{F}_t^{\xi}\{f(t/a)\} = |a|\hat{f}(a\xi) \tag{B.2}$$

- Propriedade 3, transformada da função espelhada no tempo:

$$\mathcal{F}_t^{\xi}\{f(-t)\} = \hat{f}(-\xi) \tag{B.3}$$

- Propriedade 4, transformada da função conjugada:

$$\mathcal{F}_t^{\xi}\{\overline{f}(t)\} = \overline{\hat{f}}(-\xi) \tag{B.4}$$

- Propriedade 5, transformada inversa da multiplicação:

$$\mathcal{F}^{-1}\{\hat{f}(\xi)\hat{g}(\xi)\} = f(t) * g(t) \tag{B.5}$$

em que $\mathcal{F}_t^{\xi}$ indica a transformada aplicada na função com variável de tempo t gerando uma transformada com variável de frequência ξ. Definindo a CWT da função $f(t)$ como:

$$\mathfrak{W}_f^{\psi}(a,b) = \frac{1}{\sqrt{|a|}}\int f(t)\,\overline{\psi}\left(\frac{t-b}{a}\right)\,dt \qquad a>0,$$

Será demonstrado passo a passo um modo de se obter a fórmula da transformada inversa.

B.1 Dedução

B.1.1 Aplicar transformada de *Fourier*

Aplicando a transformada de *Fourier* na transformada *wavelet* (transformando a variável b), obtêm-se:

$$\mathcal{F}_b^{\xi}\{\mathfrak{W}_f^{\psi}(a,b)\} = \mathcal{F}_b^{\xi}\left\{\frac{1}{\sqrt{|a|}}\int f(t)\,\overline{\psi}\left(\frac{t-b}{a}\right)\,dt\right\}$$

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) = \frac{1}{\sqrt{|a|}}\int f(t)\quad \mathcal{F}_b^{\xi}\left\{\overline{\psi}\left(\frac{t-b}{a}\right)\right\}\quad dt$$

e simplificando a transformada de *Fourier* dentro da integral usando as propriedades B.1, B.2, B.3 e B.1, tem-se:

$$\begin{aligned}
\mathcal{F}_b^{\xi}\left\{\overline{\psi}\left(\frac{t-b}{a}\right)\right\} &= \overline{\mathcal{F}_b^{-\xi}\left\{\psi\left(\frac{t-b}{a}\right)\right\}} \\
&= \overline{\mathcal{F}_b^{-\xi}\left\{\psi\left(\frac{-(b-t)}{a}\right)\right\}} \\
&= \overline{\mathcal{F}_b^{\xi}\left\{\psi\left(\frac{b-t}{a}\right)\right\}} \\
&= \overline{e^{+\imath 2\pi\xi t}\mathcal{F}_b^{\xi}\left\{\psi\left(\frac{b}{a}\right)\right\}} \\
&= e^{-\imath 2\pi\xi t}\overline{\mathcal{F}_b^{\xi}\left\{\psi\left(\frac{b}{a}\right)\right\}} \\
&= |a|e^{-\imath 2\pi\xi t}\overline{\mathcal{F}_b^{a\xi}\left\{\psi\left(b\right)\right\}} \\
&= |a|e^{-\imath 2\pi\xi t}\overline{\hat{\psi}}(a\xi)
\end{aligned}$$

logo:

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) = \frac{1}{\sqrt{|a|}} \int f(t) \quad \mathcal{F}_b^{\xi}\left\{\overline{\psi}\left(\frac{t-b}{a}\right)\right\} \, dt$$

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) = \frac{1}{\sqrt{|a|}} \int f(t) \, |a| e^{-\imath 2\pi\xi t} \overline{\hat{\psi}}(a\xi) \, dt$$

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) = \sqrt{|a|} \, \overline{\hat{\psi}}(a\xi) \int f(t) \, e^{-\imath 2\pi\xi t} \, dt$$

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) = \sqrt{|a|} \, \overline{\hat{\psi}}(a\xi) \, \hat{f}(\xi)$$

B.1.2 Integrar na escala

O segundo passo será o de dividir a equação por $|a|\sqrt{|a|}$, multiplicar por $\psi(a\xi)$ e então integrá-la na escala:

$$\hat{\mathfrak{W}}_f^{\psi}(a,\xi) \, \hat{\psi}(a\xi) = \sqrt{|a|} \, \overline{\hat{\psi}}(a\xi)\hat{\psi}(a\xi) \, \hat{f}(\xi)$$

$$\frac{1}{|a|\sqrt{|a|}}\hat{\mathfrak{W}}_f^{\psi}(a,\xi) \, \hat{\psi}(a\xi) = \frac{|\hat{\psi}(a\xi)|^2}{|a|} \, \hat{f}(\xi)$$

$$\int \frac{1}{|a|\sqrt{|a|}}\hat{\mathfrak{W}}_f^{\psi}(a,\xi) \, \hat{\psi}(a\xi) \, da = \int \frac{|\hat{\psi}(a\xi)|^2}{|a|} \, \hat{f}(\xi) \, da$$

$$\int \frac{1}{|a|\sqrt{|a|}}\hat{\mathfrak{W}}_f^{\psi}(a,\xi) \, \hat{\psi}(a\xi) \, da = \hat{f}(\xi) \int \frac{|\hat{\psi}(a\xi)|^2}{|a|} \, da$$

Desta equação, pode ser observada a integral que define a admissibilidade da *wavelet*. Para que a inversa possa ser encontrada, c_ψ, definida como:

$$c_\psi = \int \frac{|\hat{\psi}(a\xi)|^2}{|a|} \, da = \int \frac{|\hat{\psi}(a)|^2}{|a|} \, da$$

deve ser um número finito maior que zero. Desta forma, a equação pode ser reescrita como:

$$\hat{f}(\xi) = \frac{1}{c_\psi} \int \frac{1}{|a|\sqrt{|a|}}\hat{\mathfrak{W}}_f^{\psi}(a,\xi)\hat{\psi}(a\xi)$$

B.1.3 Aplicar a transformada inversa

O último passo será aplicar a transformada inversa de *Fourier*:

$$\hat{f}(\xi) = \frac{1}{c_\psi} \int \frac{1}{|a|\sqrt{|a|}} \hat{\mathfrak{W}}_f^\psi(a,\xi)\hat{\psi}(a\xi)$$

$$f(t) = \mathcal{F}^{-1}\left\{\frac{1}{c_\psi} \int \frac{1}{|a|\sqrt{|a|}} \hat{\mathfrak{W}}_f^\psi(a,\xi)\hat{\psi}(a\xi)\ da\right\}$$

$$f(t) = \frac{1}{c_\psi} \int \frac{1}{|a|\sqrt{|a|}} \mathcal{F}^{-1}\left\{\hat{\mathfrak{W}}_f^\psi(a,\xi)\hat{\psi}(a\xi)\right\}\ da$$

e usando a propriedade B.5:

$$f(t) = \frac{1}{c_\psi} \int \frac{1}{|a|\sqrt{|a|}} \mathcal{F}^{-1}\left\{\hat{\mathfrak{W}}_f^\psi(a,\xi)\right\} * \mathcal{F}^{-1}\left\{\hat{\psi}(a\xi)\right\}\ da$$

$$f(t) = \frac{1}{c_\psi} \int \frac{1}{|a|^2\sqrt{|a|}} \mathfrak{W}_f^\psi(a,t) * \psi\left(\frac{-t}{a}\right)\ da$$

$$f(t) = \frac{1}{c_\psi} \int \frac{1}{|a|^2\sqrt{|a|}} \mathfrak{W}_f^\psi(a,b)\ \psi\left(\frac{t-b}{a}\right)\ dtda$$

o que completa a demonstração.

Apêndice C

Scripts Usados nos capítulos do Trabalho

C.1 Série e transformada de *Fourier*

C.1.1 Transformada de *Fourier* de um sinal ruidoso

```
%*********************************************************************
%
%       Using the FFT function to extract frequencies in a noisy signal
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   3
%       last edited in: 01/22/2015
%
%*********************************************************************

Fs = 64;          % Sampling frequency
T = 1/Fs;         % Sampling period
L = 256;          % Number of samples
t = (0:L-1)*T;    % Time vector

% Signal with two waves of frequencies of 10 and 30Hz
f = sin(2 * pi * 10 .* t) + sin(2 * pi * 30 .* t) + randn(size(t));

% Plotting of the signal in the time domain
figure(1)
plot(t,f);
xlabel('t');
ylabel('f(t)');
```

```
% Number of samples at the frequency domain
N = L;

% Computing of the DFT (using the FFT pre-built function)
F = fft(f,N)/L;

% Conversion to Hz
xi = (Fs/2)*linspace(0,1,N/2 + 1);

% Plotting of the signal at the frequency domain
figure(2);
plot(xi,abs(F(1:N/2 +1)));
axis([0 32]);
xlabel('xi(hz)');
ylabel('F(xi)');
```

C.1.2 Plotagem de algumas funções janela

```
%*********************************************************************
%       Comparison of window functions
%       Before runnning the code, initialize the chosen window with
%       win = 'rect', 'hann' or 'gauss'
%
%       Octave version: 3.8.1
%       Extra packages: special functions package
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 01/28/2015
%
%*********************************************************************

pkg load specfun;

W = 10;                         % Interval size
fs = 128;                       % Sampling frequency
T = 1/fs;                       % Sampling period
L = W*fs;                       % Number of samples
t = linspace(-1/2,1/2,L)*W;     % Time vector

% Rectangular window definition
if      strcmp(win,'rect')
        f = heaviside(t+1) - heaviside(t-1);
        label = 'Rectangular window';

% Hanning window definition
elseif  strcmp(win,'hann')
        f = -0.5*(1 - cos( (2*pi*t)/W ) );
        label = 'Hanning Window';

% Gaussian Window definition
elseif  strcmp(win,'gauss')
        sigma = 1/4;
        f = exp(  -(1/2) * ( (t) / (sigma*W/2) ).^2 );
        label = sprintf('Gaussian Window with sigma = %.2f', sigma);
end

% Plotting of the signal in the time domain
subplot(3,1,1)
```

```
plot(t,f);
axis([-W/2 W/2]);
xlabel('t');
ylabel('f(t)');
title(label);

% Calculating the frequency spectrum
F = fftshift ( fft(f,L)/L );
xi = linspace(-1,1,L)*fs/2;

% Plotting the frequency spectrum
subplot(3,1,2)
plot(xi,abs(F));
axis([-5 5]);
xlabel('xi');
ylabel('|F(xi)|');
title(sprintf('Frequency spectrum of the %s', label));

% Plotting the log 10 of the frequency spectrum
subplot(3,1,3)
plot(xi,log(abs(F))/log(10));
axis([-5 5]);
xlabel('xi');
ylabel('log|F(xi)|');
title(sprintf('Log 10 of the Frequency spectrum of the %s', label));
```

C.1.3 Demonstração das limitações da transformada de *Fourier*

```
%*********************************************************************
%       Using the FFT function to show the limitation of the
%       Fourier transform
%
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 01/22/2015
%
%*********************************************************************

Fs = 128;         % Sampling frequency
T = 1/Fs;         % Sampling period
L = 256;          % Number of samples

% Time vector
t1 = (0:L/2)*T;
t2 = (L/2 + 1:L - 1)*T;
t = [t1 t2];

% Signals with two waves of frequencies of 10 and 30Hz
f1 = sin(2 * pi * 10 .* t) + sin(2 * pi * 30 .* t) + randn(size(t));
f2 = [ sin(2 * pi * 10 .* t1) , sin(2 * pi * 30 .* t2) ];

% Plotting of the signals in the time domain
figure(1)

subplot(2,1,1)
plot(t,f1,'b');
xlabel('t');
ylabel('f(t)');

subplot(2,1,2)
plot(t,f2,'r');
xlabel('t');
ylabel('f(t)');

% Number of samples at the frequency domain
```

```
N = L;

% Computing of the DFT (using the FFT pre-built function)
F1 = fft(f1,N)/L;
F2 = fft(f2,N)/L;

% Conversion to Hz
xi = (Fs/2)*linspace(0,1,N/2 + 1);

% Plotting of the first signal at the frequency domain
figure(2);

subplot(2,1,1)
plot(xi,abs(F1(1:N/2 +1)));
axis([5 32 0 8/10]);
xlabel('xi(hz)');
ylabel('F(xi)');

% Plotting of the second signal at the frequency domain

subplot(2,1,2)
plot(xi,abs(F2(1:N/2 +1)),'r');
axis([5 32 0 4/10]);
xlabel('xi(hz)');
ylabel('F(xi)');
```

C.1.4 Plotagem do espectrograma de um sinal

```
%*********************************************************************
%       Using the stft function to plot the spectrogram of a signal
%       Fourier transform
%
%       Usage:
%       [F,xi] = myspectrogram(f,t,WIN,WIN_SIZE,overlap,FIG);
%       inputs =        f - signal vector
%                       t - time vector
%                       WIN - window function
%                       WIN_SIZE - window size (in time dimension)
%                       overlap - fraction of overlap
%                       FIG - figure number to plot
%
%       outputs =       F - STFT matrix
%                       xi - frequency dimension of STFT
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   6
%       last edited in: 02/06/2015
%
%*********************************************************************

function [F,xi] = myspectrogram(f,t,WIN,WIN_SIZE,overlap,FIG);

L = length(t);
fs = floor(L/(t(end) - t(1)));

% Size of the window (500ms), and step with overlap
label = sprintf('%s window, size = %.3fs, overlap = %.2f', WIN, WIN_SIZE, overlap);

% Window size from seconds to samples
A = 1;
[B,B] = min(abs(t - (t(A) + WIN_SIZE)) );
SIZE = B - A + 1;

% Time step
INC = 1 + floor(abs(SIZE*(1 - overlap)));
```

```
% Computing of the Short Time Fourier Transform
F = stft (f, SIZE, INC, L, WIN);
F = F';

% Plotting of the spectrogram
SIZE = size(F);
xi = linspace(0,1,size(F)(1))*fs/2;
figure(FIG);
imagesc(t,xi,abs(F)', [min(min(abs(F))) max(max(abs(F)))]);
set(gca,'YDir','normal')
title(label);
xlabel('t'); ylabel('xi');
colormap(hot);
colorbar();
axis('normal')

end
```

C.2 Transformada *Wavelet*

C.2.1 Teste das funções de Morlet

```
%*********************************************************************
%
%       Testing the properties of the complex Morlet wavelet
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 01/25/2015
%
%*********************************************************************

W = 8;                                  % Signal duration (seconds)
fs = 32;                                % Sampling frequency
T = 1/fs;                               % Sampling period
L = fs*W;                               % Number of samples
t = linspace(-W/2,W/2,L);               % Time vector

% Complex Morlet Wavelet
%y = pi^(-1/4).*exp((-( (t-b)/a ).^2)/2) .* (exp(i*beta*( (t-b)/a )) - exp(-beta^2/2)
y = exp(-(t.^2) / 2).*exp(i * 2 * pi * beta * t);

%% Calculating the coefficient
c = trapz(t,( (abs(y) ).^2)./abs(t))
I = trapz(t,y)
energia = trapz(t,  ( abs(y) ).^2 )
plottitle = sprintf('a = %.1f e b = %d', a,b);

% Plotting the figure
figure(1);
%plot(t,real(y),'b','LineWidth',2,t,imag(y),'r','LineWidth',2);
plot(t,real(y),'b','LineWidth',2);
axis([-W/2 W/2]);
title(plottitle);

%print(figure(1),'-deps','-color',sprintf('beta%d.eps', beta))
```

C.2.2 Implementação da CWT

```
%*********************************************************************
%
%       Implementation of the continuous wavelet transform to analize
%       the scalegram of a signal with the morlet and mexican hat
%       wavelets
%
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   6
%       last edited in: 01/27/2015
%
%*********************************************************************


% Initializing the sinal
%---------------------------------------------------------------------
W = 10;                         % Signal duration (seconds)
fs = 32;                    % Sampling frequency
T = 1/fs;                   % Sampling period
L = fs*W;                   % Number of samples
t = (0:L-1)*T;              % Time vector

% Construction of the signal by parts
t1 = (0:floor(L/3))*T;
t2 = (floor(L/3) + 1:floor(2*L/3))*T;
t3 = (floor(2*L/3) + 1:L-1)*T;
t = [t1 t2 t3];
x1 = 1*sin(2 * pi * (1/2) .* t1);
x2 = 1*sin(2 * pi * 1 .* t2);
x3 = 1*sin(2 * pi * 3 .* t3);
f = [x1 x2 x3];

t = linspace(0,5,128);
f = margfun(t);

% Plotting of the signal in the time domain
figure(1);
subplot(4,1,1)
```

```
plot(t,f);
xlabel('t');
ylabel('f(t)');

% Calculating the fourier spectrum
F = fft(f,L)/L;
xi = (fs/2)*linspace(0,1,L/2 + 1);

% Plotting of the signal at the frequency domain
figure(2);
plot(xi,abs(F(1:L/2 +1)));
axis([0 fs/2]);
ylabel('F(xi)');

% MEXICAN HAT WAVELET SCALEGRAM
%-----------------------------------------------------------------------------

% Construction of a scale vector using log-spaced values between s1 and
% s2, with sn values
s1 = 1/2;
s2 = 40;
sn = 200;
s = logspace( log(s1)/log(10), log(s2)/log(10) , sn);

% Definition of the mexican hat wavelet
omega = 1/8;
function y = mhat(t,a,b,omega)
    y = -((2*pi*omega^2)*(-1/2)).*(1 - 2*pi.*(( (t-b)/a ) / omega).^2);
    y = y.*exp(-pi.*(( (t-b)/a )/omega).^2);
endfunction

Fmhat = zeros(sn,L);

% Calculating the wavelet transform with the mexican hat
for n = 1:sn
  for m = 1:L
    a = s(n);
    b = m*T;
    y = mhat(t,a,b,omega);
    Fmhat(n,m) = trapz(t,f.*y/sqrt(a));
  end
end

% Plotting of the scalegram
figure(1);
subplot(4,1,2)
imagesc(t,s,abs(Fmhat));
title('Magnitude of Maar CWT');
%set(gca,'YDir','normal')
axis('normal'); ylabel('scales');
```

```
colormap(hot);
colorbar();

% COMPLEX MORLET WAVELET SCALEGRAM
%-----------------------------------------------------------------------------

% Construction of a scale vector using  log-spaced values between s1 and
% s2, with sn values
s1 = 1/5;
s2 = 4;
sn = 200;
s = logspace( log(s1)/log(10), log(s2)/log(10) , sn);

% Definition of the complex morlet wavelet
beta = 5;
function y = morlet(t,a,b,beta)
    y = pi^(-1/4).*exp((-( (t-b)/a ).^2)/2) .* (exp(i*beta*( (t-b)/a ));
    y = y - exp(-beta^2/2));
endfunction

Fmorlet = zeros(sn,L);

% Calculating the wavelet transform with the complex morlet
for n = 1:sn
  for m = 1:L
    a = s(n);
    b = m*T;
    y = morlet(t,a,b,beta);
    Fmorlet(n,m) = trapz(t,f.*y/sqrt(a));
  end
end

% Plotting of the scalegram
subplot(4,1,3)
imagesc(t,s,abs(Fmorlet));
title('Magnitude of Morlet CWT with beta = 5');
%set(gca,'YDir','normal')
axis('normal'); ylabel('scales');
colormap(hot);
colorbar();

% Plotting of the scalegram
subplot(4,1,4)
imagesc(t,s,angle(Fmorlet));
title('Phase of Morlet CWT with beta = 5');
%set(gca,'YDir','normal')
axis('normal'); ylabel('scales');
colormap(hot);
colorbar();
```

C.2.3 Função de Cantor

```
%*********************************************************************
%
%       Definition of the Cantor function
%
%       Usage = out = cantor(t,N)
%
%       inputs =        t - time vector (must have a power of 2 samples)
%                       N - number of iterations
%
%       outputs =       out - cantor function
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 01/31/2015
%
%*********************************************************************

% Function declaration
function out = cantor(t,N)

        L = length(t);                  % Vector size (samples)
        W = t(end);                     % Time interval
        fs = 2^nextpow2(L/W);           % Sampling frequency
        out = t;                        % 0th iteration

        out(floor(L/3):floor(2*L/3)) = 0.5*W;


        % Iterative processes
        for n = [0:N-1]
            m=1;

            while (m < L/3)
                out(m) = 0.5 * out( 3*m );
                m = m+1;
            end

            m = floor(2*L/3);
```

```
            while m <= L
                out(m) = 0.5 * (W + out( abs(floor((3*m - 2*fs*W))) ) );
                m = m+1;
            end
        end
end
```

C.3 Palhetas de Cores

C.3.1 Criação da matriz de Farge

```
%*****************************************************************************
%
%       Creating a array of colors with the Farge algorithm
%
%       Usage:
%       v = farge(J,I,v11)
%
%       inputs  =       I - number of crominances
%                       J - number of colormaps wanted
%                       v11 - first value
%
%       outputs =       v - farge array
%
%       Octave version: 3.8.1
%
%       Created by Evandro Bernardes for the science initiation project "Integral Tra
%       and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics at the National
%       of Space Research
%
%       Code version:   2
%       last edited in: 06/11/2015
%
%*****************************************************************************
function v = farge(J,I,v11)
        for i=1:I
                for j=1:J
                        v(i,j) = mod((v11 + 6*(i-1)/I + 6*(j-1)/(I*J) ) , 6);
                end
        end
end
```

C.3.2 Conversão da matriz de Farge em vetores RGB

```
%*****************************************************************************
%
%       Convert Farge's array into RGB vectors
%
%       Usage:
%       [rm,gm,bm] = fargetorgb(v)
%
%       inputs  =       v - farge array
%
%       outputs =       rm - red vector
%                       gm - green vector
%                       bm - blue vector
%
%       Octave version: 3.8.1
%
%       Created by Evandro Bernardes for the science initiation project "Integral Tra
%       and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics at the National
%       of Space Research
%
%       Code version:   2
%       last edited in: 06/11/2015
%
%*****************************************************************************
function [rm,gm,bm] = fargetorgb(v)
        rm = [];
        gm = [];
        bm = [];
        I = size(v)(1);
        J = size(v)(2);
        t = linspace(0,1,I);
        for i=1:I
                for j=1:J

                        if      (v(i,j) == 1)     % Red
                                rgb = [255,     0,      0]/255;

                        elseif  ((v(i,j) > 1) && (v(i,j) < 2))
                                rgb = [255,     128/2,  0]/255;

                        elseif  (v(i,j) == 2)     % Orange
                                rgb = [255,     128,    0]/255;

                        elseif  ((v(i,j) > 2) && (v(i,j) < 3))
                                rgb = [255/2,   383/2,  0]/255;
```

```
                    elseif (v(i,j) == 3)     % Yellow
                            rgb = [255,      255,     0]/255;

                    elseif ((v(i,j) > 3) && (v(i,j) < 4))
                            rgb = [255/2,    255,     0]/255;

                    elseif (v(i,j) == 4)     % Green
                            rgb = [0,                 255,     0]/255;

                    elseif ((v(i,j) > 4) && (v(i,j) < 5))
                            rgb = [0,                 255/2,   255/2]/255;

                    elseif (v(i,j) == 5)     % Blue
                            rgb = [0,                 0,       255]/255;

                    elseif ((v(i,j) > 5) && (v(i,j) < 6))
                            rgb = [127/2,    0,       255]/255;

                    elseif (v(i,j) == 6)     % Violet
                            rgb = [127,      0,       255]/255;

                    else
                            rgb = [382/2,    0,       255/2]/255;

                    end

            rgbcell = num2cell(rgb);
            [rm(i,j+1),gm(i,j+1),bm(i,j+1)] = rgbcell{:};

            time = [t(i),t(i),t(i)];
            timecell = num2cell(time);
            [rm(i,1),gm(i,1),bm(i,1)] = timecell{:};
            end
        end
end
```

C.3.3 Criação de uma palheta de cores com os vetores RGB

```
%*****************************************************************************
%
%       Create maps from RGB vectors
%
%       Usage:
%       map = createmap(N, rm, gm, bm,j,type)
%
%       inputs  =       N - number of points;
%                       rm, gm and bm - RGB vectors, may or may not be converted from
%                       j - number of map from array
%                       type - 'linear' (default) or 'log'
%
%       outputs =       map - result colormap
%
%       Octave version: 3.8.1
%
%       Created by Evandro Bernardes for the science initiation project "Integral Tra
%       and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics at the National
%       of Space Research
%
%       Code version:   1
%       last edited in: 01/30/2015
%
%*****************************************************************************
function map = createmap(N, rm, gm, bm,j,type)

        % checking if log method is used
        if (strcmp(type,'log'))
                rm(:,j+1) = exp(rm(:,j+1));
                gm(:,j+1) = exp(gm(:,j+1));
                bm(:,j+1) = exp(bm(:,j+1));
        end

        x = linspace(0,1, N);                   % Linear vector with N values
        rv = interp1( rm(:,1), rm(:,j+1), x);   % Interpolation of red matrix
        gv = interp1( gm(:,1), gm(:,j+1), x);   % Interpolation of green matrix
        bv = interp1( bm(:,1), bm(:,j+1), x);   % Interpolation of blue matrix

        % if log, return to log values
        if (strcmp(type,'log'))
                rv = log(rv);
                gv = log(gv);
                bv = log(bv);
        end
```

```
        map = [ rv', gv', bv'];                    % Creation of map

        % Deleting of invalid values
        map( isnan(map) ) = 0;
        map( (map>1) ) = 1;
        map( (map<0) ) = 0;
end
```

C.3.4 Plotagem de uma palheta de cores

```
%*****************************************************************************
%
%       Plot colormap's RGB spectra and luminosity medium
%
%       Usage:
%       showmap(Map,FIG)
%
%       inputs  =       Map - desired colormap
%                       FIG - figure to be ploted
%
%       Octave version: 3.8.1
%
%       Created by Evandro Bernardes for the science initiation project "Integral Tra
%       and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics at the National
%       of Space Research
%
%       Code version:   2
%       last edited in: 06/11/2015
%
%*****************************************************************************
function showmap(Map,FIG)
        x = linspace(0,1, size(Map)(1));
        figure(FIG)
        lw = 4;
        plot(   x, Map(:,1),'color',[1,0,0],'linewidth',lw,
                x, Map(:,2),'color',[0,1,0],'linewidth',lw,
                x, Map(:,3),'color',[0,0,1],'linewidth',lw,
                x, mean(Map,2),'color',[0.7,0.7,0.7],'o')
        xlabel 'fraction'
        ylabel 'intensity'
end
```

C.3.5 Exemplo de palheta criada com as funções deste trabalho

```
%*****************************************************************************
%
%       Demonstration of colormap algorithms
%
%       Octave version: 3.8.1
%
%       Created by Evandro Bernardes for the science initiation project "Integral Tra
%       and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics at the National
%       of Space Research
%
%       Code version:   2
%       last edited in: 06/11/2015
%
%*****************************************************************************

% Initializing values

J = 2;          % Number of colormaps
I = 5;          % Number of chrominances in each map
v11 = 2;        % Number first color in first map, according to Farge's color algebra

% Creating farge array
v = farge(J,I,v11);
v(:,3) = v(:,2);
v(1,3)=v(I,3);
v(I-1,3) = v(2,3);
%v(I-1,3) = 6;
% Conversion of array to RGB matrices
[rm,gm,bm] = fargetorgb(v);

% Creation of maps
N = 1000;               % Number of points in interpolation
type = 'log';   % Interpolation type

% Loop for creating J maps
for j = 1:J+1
        map{j} = createmap( N, rm, gm, bm,j,type);
end
```

C.4 Transformada de *Hilbert*

C.4.1 Cálculo da fase instantânea

```
%*********************************************************************
%
%       Calculating the instantaneous phase of a signal with the
%       hilbert transform
%
%       Usage:
%       phase = myinstphase(f);
%       inputs =        f - signal vector
%
%       outputs =       phase - instantaneous phase
%
%       Octave version: 3.8.1
%       Extra packages: Signal package
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 02/02/2015
%
%*********************************************************************
function phase = myinstphase(f)

    % Loading the hilbert function
    pkg load signal;

    % Calculating the analytic signal
    fa = hilbert(f);

    % Phase of the analytic signal
    phase = angle(fa);
end
```

C.4.2 Cálculo da frequência instantânea

```
%*********************************************************************
%
%       Calculating the instantaneous frequency of a signal with the
%       hilbert transform
%
%       Usage:
%       freq = myinstfreq(t,f);
%       inputs =        f - signal vector
%                       t - time vector
%
%       outputs =       freq - instantaneous frequency
%
%       Octave version: 3.8.1
%       Extra packages: Signal package
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   2
%       last edited in: 02/05/2015
%
%*********************************************************************
function freq = myinstfreq(f,t)


    % Loading the hilbert function
    pkg load signal;

    % Calculating the analytic signal
    fa = hilbert(f);

    % Phase of the analytic signal
    phase = unwrap(angle(fa));

    % Calculating the frequency
    freq = (diff(phase)./diff(t))/(2*pi);

    freq = [freq freq(end)];
end
```

C.4.3 Cálculo do espectro de *Hilbert*

```
%*********************************************************************
%
%       Calculating the Hilbert spectrum of a signal
%
%       Usage:
%       H =  myhspectrum(f,t,xis,FIGabs,FIGphase);
%       inputs =        f - signal vector
%                       t - time vector
%                       xis - frequency vector
%                       FIGabs - figure to plot the hilbert spectrum
%                       FIGphase - figure to plot the hilbert phase
%
%       outputs =       H - return Hilbert spectrum matrix, if wanted
%
%       Octave version: 3.8.1
%       Extra packages: Signal package
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   3
%       last edited in: 06/06/2015
%
%*********************************************************************

function H = myhspectrum(f,t,xis,FIGabs,FIGphase)

L = length(f);

fs = 1 + floor(L / (t(end) - t(1)));
pkg load signal;

% Declaration of the hannning window with width = a
function filter = hann(xi,a)
  pkg load specfun; % Loading heaviside function
  filter = (heaviside(xi + a/2) - heaviside(xi - a/2));
  filter = filter*0.5.*(1 + cos( (2*pi*(xi))/a));
end

% Using the hanning window as a pass-band filter at center frequency = b;
function filter = hannpass(xi,b,a)
  filter = hann(xi-b,a) + hann(xi+b,a);
end
```

```
% Frequency vector
xi = linspace(-1,1,L)*fs/2;

% Signal's fft
F = fftshift( fft(f,L) );

% Declaration of central frequencies used
%xis = linspace(0,10,200);
a = 1;

HS = [];
HP = [];
% Calculating the Hilbert Spectrum

for n = 1:length(xis)
    b = xis(n);
    H = hannpass(xi,b,a);
    FF = F.*H;
    ff = ifft ( fftshift(FF) , L );
    fa = hilbert(real(ff));
    H(n,:) = fa;
    HS(n,:) = abs(fa);
    HP(n,:) = atan2(imag(fa),real(fa));
end

% Plotting the Hilbert Spectrum
figure(FIGabs)
imagesc(t,xis,HS);
axis('normal')
axis('xy')
xlabel('t')
ylabel('xi')
title('Hilbert spectrum')
colormap(hot)

% Plotting the Hilbert Spectrum
figure(FIGphase)
imagesc(t,xis,HP);
axis('normal')
axis('xy')
xlabel('t')
ylabel('xi')
title('Hilbert phase')
colormap(hot)

end
```

C.5 Transformada de *Hilbert-Huang*

C.6 Transformada de *Fourier* e *Wavelet* em duas dimensões

C.6.1 Extração de frequências espaciais de uma imagem

```
%*********************************************************************
%
%       Using the FFT2 function to extract spatial frequencies of a 2D
%       image
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   1
%       last edited in: 02/03/2015
%
%*********************************************************************

A = 10;          % x interval
B = 10;          % y interval
fs = 32;         % Sampling frequency
T = 1/fs;        % Sampling period
M = fs*A;        % Number of samples in x
N = fs*B;        % Number of samples in y

x = (0:M-1)*T;           % x vector
y = (0:N-1)*T;           % y vector
[x y] = meshgrid(x,y);   % xy grid

s = cos(2*pi*(1/10)*( x.^2 + y.^2 ) );

% Plotting of the signal in the space domain
figure(1)
imagesc(x,y,s);
colorbar
xlabel('x');
ylabel('y');

% Creation of spatial frequencies vector
u = linspace(-1,1,M)*fs/2;
v = linspace(-1,1,N)*fs/2;
```

```
% Computing of the DFT (using the FFT pre-built function)
S = fftshift(fft2(s,M,N));

% Plotting of the signal at the frequency domain
figure(2);
imagesc(u,v,abs(S));
title 'absolute value'
colormap(hot)
colorbar
%axis([-10 10 -10 10])
xlabel('u');
ylabel('v');

figure(3);
imagesc(u,v,angle(S));
title 'phase'
colormap(hot)
colorbar
%axis([-10 10 -10 10])
xlabel('u');
ylabel('v');
```

C.6.2 Filtragem de uma imagem no domínio de *Fourier*

```
%*********************************************************************
%
%       Using the FFT2 function to filter a image
%
%       Octave version: 3.8.1
%       Extra packages: none
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   2
%       last edited in: 02/04/2015
%
%*********************************************************************

% Loading image
I = (imread('/home/ebernardes/Dropbox/PIBIC/Capitulos/Cap2D/Imagens/fft2d/cathedral.j
I = I(:,:,1);

% Finding min and max values of image signal
dmin = min(min(abs(I)));
dmax = max(max(abs(I)));

% Plotting of the image
figure 1; imshow(I, [dmin dmax]);
title image;

% Calculating the DFT
fftI = fft2( I ) / ( size(I)(1) * size(I)(2) );
fftAbs = abs(fftshift (fftI) );
fftPhase = angle(fftshift (fftI) );

% Finding min and max values of fft signal
fmin = min(min(abs(fftI)));
fmax = max(max(abs(fftI)));

% Plotting the absolute value and the phase of the DFT
figure 2; imagesc(fftAbs, [0 1]); colormap(gray); colorbar;
title 'absolute value'; xlabel('k_x'); ylabel('k_y');
figure 3; imagesc(fftPhase, [-pi pi]); colormap(jet); colorbar;
title phase; xlabel('k_x'); ylabel('k_y');

% Calculating filter
```

```
pkg load specfun;
xlength = size(fftI)(2);
ylength = size(fftI)(1);
[x y] = meshgrid(1:xlength,1:ylength);
H =  1 - heaviside(  ( (x - xlength/2)/1 ).^2 + ( (y-ylength/2)/1 ).^2 - 50.^2);
figure 4; imagesc(x,y, H, [0 1]); colormap(gray); colorbar
title 'filter'; xlabel('k_x'); ylabel('k_y');

% Applying filter to fft
fftI1 = H.*fftshift(fftI);
fftI2 = (1 - H).*fftshift(fftI);

% Getting back filtered images
If1 = ifft2(fftI1);
If2 = ifft2(fftI2);

% Finding min and max values of fft signal
f1min = min(min(abs(If1)));
f1max = max(max(abs(If1)));
% Finding min and max values of fft signal
f2min = min(min(abs(If2)));
f2max = max(max(abs(If2)));

% Showing filtered images
figure(5)
imshow(abs(If1), [f1min f1max])
title 'low pass image'
figure(6)
imshow(abs(If2), [f2min f2max])
title 'high pass image'
```

C.6.3 Análise de uma imagem com *wavelets*

Script do exemplo na página 7.2.1.

```
%*********************************************************************
%
%       Wavelet analysis of image with mexican hat wavelet YAWTb
%
%       Octave version: 3.8.1
%       Extra packages: YAWTb
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   4
%       last edited in: 06/11/2015
%
%*********************************************************************

function wip(I)

% Using a mean of the three color channels
I = (I(:,:,1) + I(:,:,2) + I(:,:,3))/3;

N = size(I)(2); % Samples in dimension x
M = size(I)(1); % Samples in dimension y

% Finding min and max values of image signal
dmin = min(min(abs(I)));
dmax = max(max(abs(I)));

% Plotting of the image
figure(1); imshow(I, [dmin dmax]);
title image;

% Choosing parameters for CWT calculation
a = linspace(100, 1000 , 20);   % Scales for scale-angle representation
theta = linspace(-pi,pi,10);    % Angles for scale-angle representation
apos = [20 50];          % Scales for position representation
thetapos = [0];
mother = 'morlet';               % Mother wavelet to be used

% Calculating scale-angle representation
Wsam = samcwt2d(fft2(I)/(N*M),mother,a,theta);

% Calculating position representations
```

```
%Wpos = cwt2d(fft2(I)/(N*M),mother,apos,thetapos);

% Plotting of scale-angle representation
figure(2)
yashow(Wsam);
axis('normal');
title 'scale-angle representation with morlet wavelet';
colorbar;
colormap(hot)
l = 1;

% Plotting position representations
%for k = 1:length(apos)
%     figure(k + 2);

%     temp = Wpos.data(:,:,k);
%         imagesc(angle(temp));

    %yashow(Wpos,'sc',k,'angle',l);
 %    axis('normal');
  %  xlabel 'x'; ylabel 'y';
   % title(sprintf('phase of pos representation with mhat, for scale = %d, angle = 0'

 %    colormap(customphase);
         %colorbar;
    %print(figure(2+k),'-depsc2',sprintf('a-%d.eps', apos(k)))
%end

%end
```

C.6.4 Representação de posição com YAWTB

```
%*********************************************************************
%
%       Position representation of CWT of image
%
%       Usage:
%       Wpos = mycwt2d_pos(I,mother,a,theta)
%
%       inputs  =       I - input image
%                       mother - mother wavelet
%                       a - scales vector
%                       theta - angles vector
%
%       outputs =       Wpos - coefficients of CWT
%
%       Octave version: 3.8.1
%       Extra packages: YAWTb
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   2
%       last edited in: 02/04/2015
%
%*********************************************************************

function Wpos = mycwt2d_pos(I,mother,a,theta)
I = I(:,:,1);

N = size(I)(2); % Samples in dimension x
M = size(I)(1); % Samples in dimension y

% Finding min and max values of image signal
dmin = min(min(abs(I)));
dmax = max(max(abs(I)));

% Calculating position representations
Wpos = cwt2d(fft2(I)/(N*M),mother,a,theta);

end
```

C.6.5 Representação de escala-ângulo com YAWTB

```
%*********************************************************************
%
%       Scale-angle representation of CWT of image
%
%       Usage:
%       Wsam = mycwt2d(I,mother,a,theta,fig)
%
%       inputs  = I - input image
%                     mother - mother wavelet
%                     a - scales vector
%                     theta - angles vector
%                     fig - number of figure to plot
%
%       outputs = Wsam - coefficients of CWT
%
%       Octave version: 3.8.1
%       Extra packages: YAWTb
%
%       Created by Evandro Bernardes for the science initiation project:
%       "Integral Transforms and Space Technology Applications"
%
%       at INPE/CTE/LAC
%
%       - Associate Laboratory for Computing and Applied Mathematics
%       at the National Institute of Space Research
%
%       Code version:   2
%       last edited in: 06/06/2015
%
%*********************************************************************

function Wsam = mycwt2d_sam(I,mother,a,theta,fig)

        I = I(:,:,1);

        N = size(I)(2); % Samples in dimension x
        M = size(I)(1); % Samples in dimension y

        % Finding min and max values of image signal
        dmin = min(min(abs(I)));
        dmax = max(max(abs(I)));

        % Calculating scale-angle representation
        Wsam = samcwt2d(fft2(I)/(N*M),mother,a,theta);

        % Plotting of scale-angle representation
        figure(fig)
        yashow(Wsam);
```

```
        axis('normal');
        title 'scale-angle representation with morlet wavelet';
        colorbar;
        colormap(hot)
        l = 1;

end
```

Apêndice D

Instalação do YAWTB no GNU/Octave

Como a toolbox é feita para ser instalada no MATLAB, na verdade ela não foi instalada no GNU/Octave. O processo feito foi o de copiar a pasta do pacote em um diretório de minha escolha, e em seguida foram usados no programa os seguintes comandos:

```
addpath(genpath("$DIRETORIO_ESCOLHIDO"))
```

Em que `$DIRETORIO_ESCOLHIDO` é o caminho para onde o pacote do YAWTB foi extraído. O comando `addpath` serve para forçar o Octave a incluir esta pasta na pesquisa por funções, e `genpath` serve para selecionar não só o diretório, como também todos os seus sub diretórios. A combinação dos dois fez com que todos os sub diretórios fossem adicionados na pesquisa por funções, fazendo assim com que todas elas estivessem a minha disposição.

Em seguida, para facilitar o uso futuro da toolbox, o mesmo comando foi inserido no arquivo `octaverc`, que é o arquivo onde todas as funções inseridas nele são executadas no início do programa. Assim, sempre que eu iniciar o GNU/Octave, ele já saberá onde devem ser procuradas estas funções.