

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/349747086>

# METODOLOGIAS PARA A APLICAÇÃO DE TÉCNICAS TEMPO-FREQUÊNCIA EM DINÂMICA ESTRUTURAL E AO MÉTODO DOS ELEMENTOS DE CONTORNO

Thesis · September 2001

DOI: 10.13140/RG.2.2.25202.84166

CITATIONS

5

READS

23

5 authors, including:



[Henrique F. Bucher](#)

Vitorian LLC

8 PUBLICATIONS 58 CITATIONS

[SEE PROFILE](#)



[Carlos Magluta](#)

Federal University of Rio de Janeiro

70 PUBLICATIONS 518 CITATIONS

[SEE PROFILE](#)



[N. Roitman](#)

Federal University of Rio de Janeiro

76 PUBLICATIONS 531 CITATIONS

[SEE PROFILE](#)



[Luiz C. Wrobel](#)

Brunel University London

336 PUBLICATIONS 9,459 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Project Boundary element formulations for acoustics problems [View project](#)



Project Moving boundary problems [View project](#)

METODOLOGIAS PARA A APLICAÇÃO DE TÉCNICAS TEMPO-FREQUÊNCIA  
EM DINÂMICA ESTRUTURAL E AO MÉTODO DOS ELEMENTOS DE  
CONTORNO

Henrique Frederico Bucher

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO EM ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA CIVIL.

Aprovada por:

---

Prof. Carlos Magluta, D.Sc.

---

Prof. Webe João Mansur, Ph.D.

---

Prof. Ney Roitman, D.Sc.

---

Prof. Álvaro Luiz Gayoso Azeredo Coutinho, D.Sc.

---

Prof. José Antonio Fontes Santiago, D.Sc.

---

Prof. Paulo Batista Gonçalves, D.Sc.

RIO DE JANEIRO, RJ - BRASIL  
SETEMBRO DE 2001

BUCHER, HENRIQUE FREDERICO

Metodologias para a Aplicação de Técnicas  
Tempo-Frequência em Dinâmica Estrutural e ao  
Método dos Elementos de Contorno [Rio de  
Janeiro] 2001

XII, 216 p., 29,7 cm (COPPE/UFRJ, D.Sc.,  
Engenharia Civil, 2001)

Tese – Universidade Federal do Rio de Janeiro,  
COPPE

1. Técnicas tempo-frequência
  2. Dinâmica estrutural
  3. Compressão de dados
  4. Wavelets
  5. Método dos elementos de contorno
- I. COPPE/UFRJ II. Título ( série )

Dedico este trabalho à minha amada esposa, Selma.

## AGRADECIMENTOS

Agradeço a meus queridos pais, Henrique e Ziléa pelo esforço de uma vida. Esta tese não teria sido possível, ou mesmo não teria sentido, sem vocês. Agradeço a Selma pela paciência bíblica, memorável, na minha aflição diária e pelos momentos maravilhosos que vivemos durante todos estes anos de entendimento.

Gostaria de agradecer de coração à minha irmã, Evelane, por todo o apoio, psicológico e, principalmente, material emprestado durante todo o período de estudos. Às minhas irmãs Hannelore, Ivelise e Dahlen pela presteza com que se dispuseram a ajudar em todas as oportunidades.

Nunca poderia deixar de agradecer aos meus incansáveis orientadores Carlos Magluta e Webe Mansur pela orientação, pelas oportunidades como o estágio em Brunel, pelas idéias das aplicações e, principalmente, pela sabedoria salomônica com que souberam trabalhar as dificuldades surgidas durante o curso.

Na Inglaterra, ao meu orientador, Luiz Wrobel, que tantas contribuições ofereceu ao último capítulo, pela competência na orientação, pela amizade e também pelas referências profissionais. Aos meus amigos Panayotis, Alkimin, Matthew, Zhidong e Peter pela sincera amizade. Aos professores Francisco, Stolarski e Ibi pela grata disposição e pelo espírito aberto com que me receberam.

Ao Professores Ney e Álvaro pela força e pelas referências bibliográficas. Ao meu amigo Ortigão pela amizade, pelas referências profissionais e pelo apoio financeiro no período difícil ao início da tese.

Aos meus amigos da COPPE Renata Faísca, Cláudio, Gadéa, Natália, Ulisses, Ana, Lúcia, Carlos, André Fraga e aos professores Telles, Santiago e Carrer pela amizade demonstrada durante o curso.

À CAPES e ao CNPq pelo apoio financeiro durante o período de estudos desta tese.

Gostaria de deixar registrado um agradecimento especial a todos os funcionários, sem exceção, do departamento de engenharia mecânica da universidade de Brunel, um inesquecível modelo de educação e excelência no atendimento aos alunos.

Como bom cristão não poderia deixar de agradecer a meu Deus, ao meu Senhor Jesus Cristo pelas ferramentas com as quais pude lapidar esta obra. Espero que ela esteja à altura dos talentos que a mim foram confiados.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

METODOLOGIAS PARA A APLICAÇÃO DE TÉCNICAS TEMPO-FREQUÊNCIA  
EM DINÂMICA ESTRUTURAL E AO MÉTODO DOS ELEMENTOS DE  
CONTORNO

Henrique Frederico Bucher

Setembro/2001

Orientadores: Carlos Magluta

Webe João Mansur

Programa: Engenharia Civil

Este trabalho desenvolve uma série de metodologias para adaptar uma classe especial de técnicas de processamento de sinais, denominadas transformadas tempo-frequência, a três problemas distintos em engenharia civil: determinação da taxa de amortecimento de sistemas harmônicos, compressão de sinais provenientes de ensaios de laboratório e compressão de matrizes provenientes do método dos elementos de contorno. Simulações com dados sintéticos e experimentos comprovam os excelentes resultados obtidos.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

METHODOLOGIES FOR THE APPLICATION OF TIME-FREQUENCY  
TECHNIQUES IN STRUCTURAL DYNAMICS AND TO THE BOUNDARY  
ELEMENT METHOD

Henrique Frederico Bucher

September, 2001

Advisors: Carlos Magluta

Webe João Mansur

Department: Civil Engineering

The present work develops some methodologies to adapt a special class of signal processing techniques, called time frequency transforms, to three problems in civil engineering: determination of damping ratio in harmonic systems, compression of signals resulting from laboratory tests and compression of matrices arising from the boundary element method. Simulations with sintetic data and experiments show the efficiency of the proposed methods.

# ÍNDICE

<b>I INTRODUÇÃO .....</b>	<b>1</b>
<b>II TEORIA BÁSICA .....</b>	<b>5</b>
II.1 O SPECTROGRAMA .....	9
II.2 A TRANSFORMADA WAVELET .....	14
II.3 AS TRANSFORMADAS BILINEARES.....	33
<b>III CARACTERIZAÇÃO DA PERDA DE ENERGIA EM SISTEMAS AMORTECIDOS.....</b>	<b>43</b>
III.1 INTRODUÇÃO .....	43
III.2 DESENVOLVIMENTO TEÓRICO .....	44
III.3 IMPLEMENTAÇÃO COMPUTACIONAL .....	60
III.4 SIMULAÇÃO NUMÉRICA.....	64
III.5 TESTES EXPERIMENTAIS COM PÓRTICO .....	69
<b>IV COMPRESSÃO DE SINAIS .....</b>	<b>93</b>
IV.1 COMPRESSÃO DE DADOS .....	93
IV.2 EXEMPLO DE COMPRESSÃO VIA TRANSFORMADA WAVELET .....	96
IV.3 COMPRESSÃO DE SINAIS DE TRÁFEGO .....	102
IV.4 TRANSFORMADAS MULTIDIMENSIONAIS .....	109
<b>V COMPRESSÃO DE MATRIZES RESULTANTES DA DISCRETIZAÇÃO DE OPERADORES INTEGRAIS ....</b>	<b>114</b>
V.1 INTRODUÇÃO .....	114
V.2 INTRODUÇÃO DA TRANSFORMADA WAVELET AO MÉTODO DOS ELEMENTOS DE CONTORNO .....	117
V.3 O MÉTODO DA MONTAGEM VIRTUAL (MMV) DA MATRIZ DO SISTEMA .....	121
V.4 A TRANSFORMADA EM BLOCOS.....	127
V.5 CONDIÇÕES DE CONTORNO NÃO-LINEARES .....	134
V.6 ESTUDO DOS PARÂMETROS PARA ANÁLISE .....	136
V.7 PARALELIZAÇÃO E PARTICIONAMENTO.....	147
V.8 IMPLEMENTAÇÃO COMPUTACIONAL .....	150
V.9 APlicações .....	157
<b>VI CONCLUSÕES.....</b>	<b>188</b>
VI.1 FUTUROS DESENVOLVIMENTOS.....	193
<b>VII APÊNDICE A PADRÃO DE COMPATIBILIDADE BINÁRIA .....</b>	<b>195</b>
<b>VIII APÊNDICE B INTERFACES BÁSICAS PARA O MÉTODO DOS ELEMENTOS DE CONTORNO .....</b>	<b>200</b>
<b>IX REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>207</b>

## ÍNDICE DE FIGURAS

Figura 1	Diagrama esquemático do escopo da teoria das transformadas tempo-frequência	2
Figura 2	O espectroscópio de Kirchhoff e Bunsen	5
Figura 3	Transformação de vetores entre os espaços tempo e frequência	7
Figura 4	Introdução da transformada tempo-frequência	8
Figura 5	Janela retangular	10
Figura 6	Sinal decomposto por janela retangular de largura 0,25s.	10
Figura 7	Sinal utilizado composto por seno e um impacto	11
Figura 8	Espectrograma com janela curta ( $T = 0,01s \equiv 1\%$ )	12
Figura 9	Espectrograma com janela larga ( $T = 0,16s \equiv 16\%$ )	12
Figura 10	Espectrograma com janela média ( $T = 0,04s \equiv 4\%$ )	13
Figura 11	Localização dos coeficientes da série wavelet no domínio transformado (wavelet)	19
Figura 12	Transformada de Fourier das wavelets $\psi_{j,k}(t)$ da família Daubechies-8	20
Figura 13	Funções da base wavelet em frequência, evidenciando o caráter de “fechamento” da função de aproximação $\varphi(t)$	22
Figura 14	Subdivisão e posterior reconstrução do espectro em duas bandas utilizando-se um filtro digital comum	25
Figura 15	Implementação dos blocos básicos da Fast Wavelet Transform (FWT)	26
Figura 16	Encadeamento dos blocos de decomposição para a obtenção dos coeficientes da série wavelet	27
Figura 17	Encadeamento dos blocos de reconstrução para a obtenção do sinal original	28
Figura 18	Esquema ilustrativo da transformada wavelet direta do exemplo	30
Figura 19	Esquema ilustrativo da transformada wavelet inversa do exemplo	31
Figura 20	Número de operações matemáticas realizadas pela FFT e a FWT	33
Figura 21	Harmônicos usados para demonstrar os termos cruzados na distribuição de Wigner-Ville	35
Figura 22	Distribuições de Wigner-Ville calculadas (a) separadamente para cada sinal e posteriormente somadas e (b) para os dois sinais somados no tempo.	35
Figura 23	<i>Aliasing</i> na distribuição de Wigner-Ville	37
Figura 24	Distribuição alias-free de Jeong e Williams	41
Figura 24	Função de Transferência de Energia (FTE)	48
Figura 25	Fatores de contaminação mínimos	51
Figura 26	Senóide a 5Hz, taxa amortecimento 15%, ruído adicionado a SNR=1	53
Figura 27	(1) Senóide originalmente a 5Hz; (2) modulada a 20Hz e (3) modulada a 40Hz	54
Figura 28	Sinal incorretamente modulado (a 20Hz)	54
Figura 29	Fluxo para correta obtenção do sinal real modulado	56
Figura 30	Metodologia de cálculo da taxa de amortecimento	57
Figura 31	Curva de amortecimento demonstrando critérios usados para regressão	58
Figura 32	Sinal original, sem ruído, e sintético/calculado: erro RMS igual a 0,18%	59
Figura 33	Tela do software usado para o cálculo do amortecimento	60
Figura 34	Detalhe da seleção do intervalo de tempo a ser analisado	62
Figura 35	Detalhe dos controles da análise	62
Figura 36	Detalhe da seleção da frequência de análise - cursor horizontal sobre a TFD	62
Figura 37	Curva de amortecimento [ $\log E(t)/\omega$ ] - detalhe da seleção do trecho para regressão linear	63
Figura 38	Seleção da frequência e resultados finais obtidos (taxas de amortecimento)	63
Figura 39	Detalhe dos controles para salvamento dos resultados	63
Figura 40	Sinal objeto da simulação e respectiva transformada de Fourier	64
Figura 41	Curva de amortecimento para a simulação, frequência 40Hz	65
Figura 42	Curvas de amortecimento resultantes da simulação	66
Figura 43	Gráfico de lineariedade calculado para o modo 4 (80Hz) mostrando a dependência do amortecimento com o tempo	68
Figura 44	Desenho esquemático do modelo ensaiado	69
Figura 45	Três primeiros modos X1 (flexão no plano X-Y), Z1 (flexão no plano Y-Z) e T1 (torção em torno do eixo Y)	70
Figura 46	Três últimos modos X2 (flexão no plano X-Y), Z2 (flexão no plano Y-Z) e T2 (torção em torno do eixo Y)	70

Figura 47 Algumas janelas comumente usadas com o espectrograma	73
Figura 48 Curva amortecimento para modo X1 (15,3Hz), teste 0X, AC2X e AC3X	75
Figura 49 Curva amortecimento para modo X1 (15,3Hz), teste 1X, AC2X e AC3X	76
Figura 50 Gráfico de lineariedade calculado para o modo X1 (15.3Hz)	76
Figura 51 Curva amortecimento para modo X1 (15,3Hz), teste 0X, AC1Z	77
Figura 52 Curvas amortecimento para modo Z1 (16,3Hz), teste 2Z e 3Z, AC1Z e AC4Z	79
Figura 53 Curva amortecimento para modo T1, teste 4T, AC1Z e AC4Z	80
Figura 54 Curva de amortecimento para o modo T1, teste 3Z, AC2X e AC3X	81
Figura 55 Gráfico de lineariedade do modo T1 (25.7 Hz)	81
Figura 56 Curva amortecimento, modo X2 (53,4Hz) testes 0X e 1X, AC2X e AC3X	83
Figura 57 Curvas amortecimento para modo Z2 (56,4Hz), teste 2Z, AC1Z e AC4Z	84
Figura 58 Curvas amortecimento para modo Z2 (56,4Hz), teste 3Z, AC1Z e AC4Z	85
Figura 59 Curvas calculadas utilizando o método SAF, teste 2Z, AC1Z e AC4Z	86
Figura 60 Curvas calculadas utilizando o método SAF, teste 3Z, AC1Z e AC4Z	86
Figura 61 Curvas amortecimento para modo T2 (80,2Hz), teste 4T, AC1Z, AC2X, AC3X e AC4Z	87
Figura 62 Riser carregado: sinal e respectiva curva de amortecimento obtida sem pré-modulação	90
Figura 63 Sinal modulado a 6 Hz	91
Figura 64 Sinal experimental e respectiva curva de amortecimento; pré-modulação a 6 Hz	91
Figura 65 Riser sem carga: sinal e respectiva curva de amortecimento	92
Figura 67 Sinais original e obtido após compressão-descompressão com perda	101
Figura 68 Sinal típico de tráfego	102
Figura 69 Sinal de tráfego filtrado com filtro elíptico ordem 8, ripples 0,5dB/40dB, frequência de corte 2Hz	103
Figura 70 Percentagem de energia retida para cada transformação	104
Figura 71 Erro máximo absoluto para cada transformação	106
Figura 72 Reconstituição do sinal original através dos coeficientes comprimidos	107
Figura 73 Detalhe (1) do efeito da compressão sobre o sinal de tráfego	108
Figura 74 Detalhe (2) do efeito da compressão sobre o sinal de tráfego	108
Figura 75 Imagem padrão “peppers”, grayscale, 512x512 pontos	110
Figura 76 Compressão realizada retendo-se os 125 coeficientes de maior módulo	110
Figura 77 Compressão realizada retendo-se os 500 coeficientes de maior módulo	111
Figura 78 Compressão realizada retendo-se os 1000 coeficientes de maior módulo	111
Figura 79 Compressão realizada retendo-se os 3000 coeficientes de maior módulo	111
Figura 80 Resíduo da compressão após retenção de 1500 coeficientes	112
Figura 81 Esquema dos principais solvers rápidos utilizados em conjunto com o método dos elementos de contorno	115
Figura 82 Matrizes $G$ e $H$ geradas a partir do problema apresentado na Figura 99. Valores dos elementos são mostrados em escala logarítmica.	119
Figura 83 Forma e propriedades da matriz “Localizadora-Extratora” (matriz LE)	129
Figura 84 Algoritmo para o cálculo da multiplicação matriz-vetor através de um esquema de particionamento em blocos	130
Figura 85 Algoritmo para o cálculo da multiplicação entre a matriz comprimida em blocos por um vetor como requerido pela equação (93) usando a transformada wavelet em blocos genérica BWT	132
Figura 86 Algoritmo para o cálculo da multiplicação matriz-vetor através de um esquema de particionamento regular em blocos de igual tamanho	133
Figura 87 Tempos de execução da transformada wavelet para algumas famílias, variando-se o tamanho do vetor transformado de 64 a 32768 elementos.	138
Figura 88 Regressão linear e logarítmica realizadas sobre os testes de velocidade da transformada wavelet.	139
Figura 89 Evolução do erro na solução do potencial com o valor do threshold para diferentes particionamentos (tamanhos de bloco)	144
Figura 90 Taxa de compressão resultante do threshold escolhido. Resultados para diferentes tamanhos de blocos.	145
Figura 91 Evolução do erro na solução do potencial com o aumento da taxa de compressão para diferentes particionamentos (tamanhos de bloco)	146
Figura 92 Solução para partição de matriz de tamanho qualquer usando a transformada em blocos genérica BWT	148
Figura 93 Soluções para a partição de matriz de tamanho qualquer para uso com a RBWT	148
Figura 94 Exemplo de armazenamento utilizando o formato comprimido por coordenada (FCCRD)	151
Figura 95 Exemplo de armazenamento utilizando o formato comprimido por linhas (FCL)	151

Figura 96 Comparação da taxa de compressão efetiva utilizando-se o formato comprimido por linhas (FCL) e o formato comprimido por coordenadas (FCC)	152
Figura 97 Avaliação do efeito sobre a taxa de compressão efetiva da substituição dos elementos da matriz esparsa de <i>doubles</i> de 64 bits para <i>floats</i> de 32 bits	153
Figura 98 Avaliação do efeito sobre a taxa de compressão efetiva da substituição dos elementos da matriz esparsa de <i>doubles</i> de 64 bits para <i>floats</i> de 32 bits e da substituição dos índices de 32 bits para inteiros de 16 bits	154
Figura 99 Problema misto padrão, placa quadrada com fluxos prescritos em três lados e potencial constante prescrito no lado esquerdo	159
Figura 100 Versões esparsas das matrizes <i>G</i> e <i>H</i> , respectivamente, geradas para o problema da placa quadrada. Somente elementos cujo módulo excede $10^{-3}$ são mostrados em cor preta.	159
Figura 101 Problema da placa quadrada: erro RMSE (potencial) como função dos valores de threshold <i>G</i> e <i>H</i> (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)	160
Figura 102 Problema da placa quadrada. Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)	161
Figura 103 Problema da placa quadrada: erros do fluxo e potencial como função da taxa de compressão	162
Figura 104 Solução do problema da placa quadrada; taxa de compressão igual a 440	163
Figura 105 Cavidade quadrada com fluxos prescritos em todos os lados	164
Figura 106 Placa infinita com cavidade: erro RMSE (potencial) como função dos valores de threshold <i>G</i> e <i>H</i> (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)	165
Figura 107 Problema da placa infinita com cavidade: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)	165
Figura 108 Problema da placa infinita com cavidade: erro do potencial como função da taxa de compressão	166
Figura 109 Problema da placa infinita com cavidade: soluções do fluxo e potencial, com e sem compressão; lado inferior; taxa de compressão resultante é 5015	166
Figura 110 Modelo de corrosão galvânica	167
Figura 111 Problema do par galvânico: soluções para o fluxo e potencial, com e sem compressão ao longo do par galvânico (lado inferior); taxa de compressão obtida igual a 1220	168
Figura 112 Problema do par catódico: erro RMSE (potencial) como função dos valores de threshold <i>G</i> e <i>H</i> (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)	168
Figura 113 Problema do par galvânico: erros do fluxo e potencial como função da taxa de compressão	169
Figura 114 Problema do par galvânico: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)	169
Figura 115 Geometria e condições de contorno para o problema da coluna de concreto	170
Figura 116 Problema da coluna de concreto: erro RMSE (potencial) como função dos valores de threshold <i>G</i> e <i>H</i> (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho	172
Figura 117 Problema da coluna de concreto: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)	172
Figura 118 Problema da coluna de concreto: erros do fluxo e potencial como função da taxa de compressão	173
Figura 119 Problema da coluna de concreto: soluções aproximadas para o potencial e fluxo, com e sem compressão; taxa de compressão igual a 670	173
Figura 120 Temperaturas obtidas em vários pontos do contorno como uma função de diferentes condições de isolamento na superfície exterior.	175
Figura 121 Fluxo de calor total trocado por unidade de tempo entre a viga de concreto e o ambiente interior como uma função da temperatura exterior e do coeficiente de transferência de calor da superfície externa	175
Figura 122 Estudo de escalamento do método de compressão	177
Figura 123 Problema misto padrão, placa quadrada com fluxos prescritos em três lados e potencial prescrito no lado direito	178
Figura 124 Solução do problema de transferência de calor com condição de radiação com emissividade máxima para a superfície superior.	181

Figura 125 Problema de transferência de calor com radiação e convecção mostrando o isolamento sucessivo da superfície superior	182
Figura 126 Problema de transferência de calor com sucessivo isolamento da superfície superior – temperaturas obtidas na superfície superior para o modelo discretizado com 64 elementos constantes por lado	183
Figura 127 Problema de transferência de calor com sucessivo isolamento da superfície superior – temperaturas obtidas na superfície superior para o modelo discretizado com 2048 elementos constantes por lado	184
Figura 128 Problema de transferência de calor com sucessivo isolamento da superfície superior – fluxo de calor obtido na superfície superior para o modelo discretizado com 64 elementos constantes por lado	185
Figura 129 Problema de transferência de calor com sucessivo isolamento da superfície superior – fluxo de calor obtido na superfície superior para o modelo discretizado com 2048 elementos constantes por lado	185
Figura 130 Solução do problema de transferência de calor com radiação para dois valores extremos de emissividade: nulo e total.	186
Figura 131 Variação da temperatura no ponto superior central devido à variação da emissividade da radiação nesta superfície	187

## ÍNDICE DE TABELAS

Tabela 1 Exemplos de núcleos de distribuições conhecidas	39
Tabela 2 Resumo dos modos detectados teórica e experimentalmente	71
Tabela 3 Percentagem de energia retida no quarto central do intervalo para algumas janelas mais utilizadas com o espectrograma	73
Tabela 4 Taxas de amortecimento obtidas pelo método da TFD	88
Tabela 5 Condições de contorno disponíveis via técnica da montagem virtual do sistema	123
Tabela 6 Derivadas das condições de contorno de acordo com a técnica da montagem virtual	136
Tabela 7 Avaliação de erro da transformada para várias famílias de wavelet. Vetores transformados randômicos com 4096 elementos.	142
Tabela 8 Comparação de tempos de leitura e escrita de números inteiros e ponto flutuante em formato texto (ASCII) e binário (ponto flutuantes são <i>doubles</i> 64 bits e inteiros de 32 bits)	156
Tabela 9 Tempo exigido por tarefa para compressão do modelo de transferência de calor com 131072 equações	179

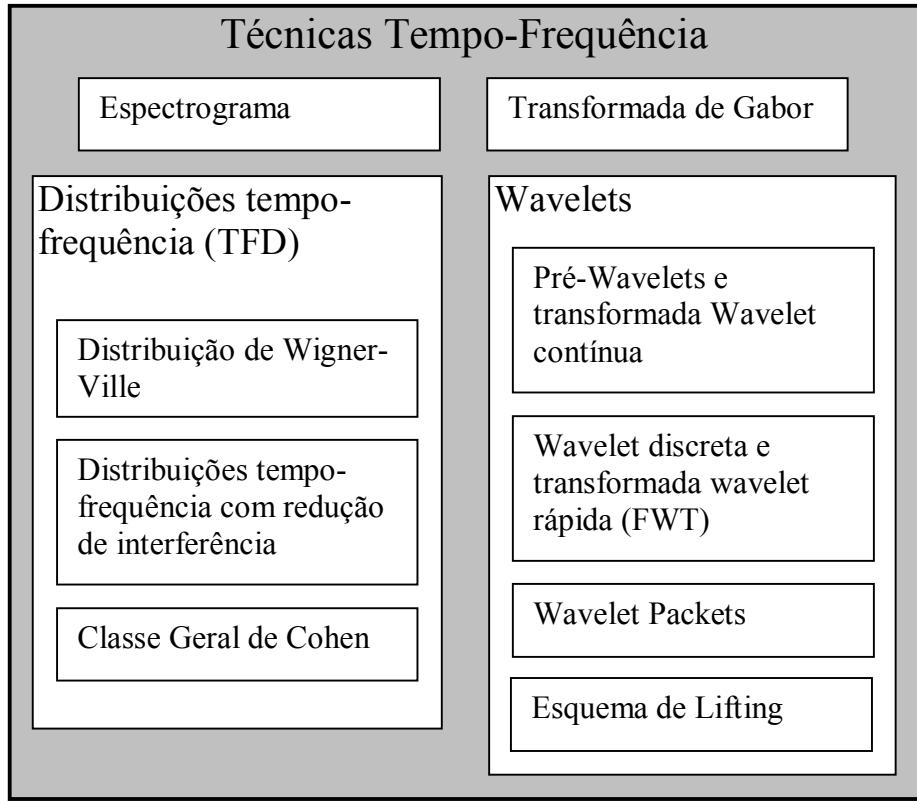
# I INTRODUÇÃO

O presente trabalho tem o objetivo de aplicar um conjunto de técnicas de processamento de sinais, geralmente abrigadas sob a terminologia *técnicas tempo frequência*, ao estudo de alguns problemas em engenharia civil.

As técnicas tempo-freqüência têm origem nos estudos de mecânica quântica (WIGNER, 1932), na década de 20, cujo estudo baseia-se no cálculo do segundo coeficiente virial dos gases, o qual envolve uma distribuição bidimensional de posição e quantidade de movimento. Esta técnica foi aplicada ao processamento de sinais por VILLE (1946) cerca de quinze anos depois ao fazer a analogia posição↔tempo e quantidade de movimento↔freqüência, criando assim a *transformada de Wigner-Ville*. Esta distribuição, apesar de simples, é bastante suscetível ao fenômeno de *interferência* ou *termos cruzados*. No sentido de reduzi-los, dos anos 40 aos anos 60 houve considerável atividade no sentido de se construir distribuições similares como a de PAGE (1952), MARGENOU-HILL (1961) ou RIHACZEK (1968).

Esta primeira fase (1930-1970) corresponde à primeira grande subdivisão das técnicas tempo-frequência (veja a Figura 1), teoria que é sintetizada em COHEN (1966) quando este divisa um método que permite construir de uma simples maneira uma infinita variedade de distribuições tempo-freqüência partindo de uma única função chamada *núcleo da distribuição*, definindo desta forma as ditas distribuições da Classe Geral de Cohen, muitas vezes denominadas simplesmente de Distribuições Tempo-Frequência (*Time-Frequency Distributions* ou TFDs).

Paralelamente ao desenvolvimento das distribuições tempo-frequência, foram publicados trabalhos (HAAR, 1910) relativos à aproximação de funções por meio de bases ortogonais finitas no tempo. Com um estilo bastante casual, MEYER (1985) faz um excelente histórico do período, onde descreve a criação das funções denominadas *pré-wavelets*. Em um trabalho de grande impacto, a francesa DAUBECHIES (1988) elabora uma base de funções não-periódicas e ortogonais denominadas *wavelets* – veja também DAUBECHIES (1992) – que constitui-se na segunda grande subdivisão das técnicas tempo-frequência, como ilustrado na Figura 1.



**Figura 1 Diagrama esquemático do escopo da teoria das transformadas tempo-frequência**

A teoria das wavelets discretas, elaborada pela francesa Ingrid Daubechies, impressiona pela elegância de seu desenvolvimento teórico. Isto fez com que uma “febre de wavelets” desse origem a uma série de trabalhos conceitualmente equivocados onde o erro básico era utilizar a transformada wavelet onde uma simples distribuição tempo-frequência (TFD) resolveria melhor o problema. Parte deste engano deveu-se, como já dito, à elegância do desenvolvimento teórico das wavelets (veja por exemplo DAUBECHIES, 1992) mas outra parte deveu-se ao desconhecimento das origens da teoria, onde inclui-se a criação do espectrograma.

A continuação do estudo da transformada wavelet originou o conceito de bases wavelet redundantes (WICKERHAUSER, 1991), que têm grande aplicação principalmente em técnicas de compressão. Posteriormente, em (SWELDENS, 1997) foi definido um esquema único de implementação computacional da transformada wavelet teoricamente independente da transformada de Fourier, da qual as demais técnicas fazem uso continuado. Esta transformada rápida oferece várias facilidades interessantes como, por exemplo, a obtenção de coeficientes inteiros a partir de funções inteiras sem transformações intermediárias para o domínio real.

As TFDs foram estudadas em tese de mestrado (BUCHER, 1998) e, seguindo esta mesma linha de desenvolvimento, foi apresentado no seminário de qualificação um método original para obtenção de características modais de sistemas harmônicos amortecidos a partir de sinais coletados em testes experimentais. Neste seminário foram mostrados resultados de algumas simulações teóricas com sinais sintéticos, uma análise de 5 experimentos com um modelo de um pórtico e algumas análises de um riser flexível utilizado em plataformas petrolíferas. Todos os resultados indicaram o excelente desempenho do algoritmo, inclusive detectando facilmente quando a taxa de amortecimento varia dentro do intervalo de aquisição do sinal.

A teoria das wavelets, estudada mais profundamente como prosseguimento da pesquisa para a proposta de doutorado, mostrou grande aplicabilidade em uma série de problemas em engenharia civil, principalmente em filtragem e compressão de sinais. A característica de compressibilidade foi posteriormente estudada para acelerar a solução numérica de problemas descritos pelo método dos elementos de contorno (BREBBIA et al., 1984).

Desta forma os bons resultados obtidos em uma grande diversidade de aplicações, aparentemente sem conexão umas com as outras, causou uma dificuldade em estruturar todo o conteúdo levantado para formar um texto coerente. A abordagem escolhida para atingir este objetivo foi estruturar o texto com o centro nas aplicações. A teoria básica para entendimento de todas as aplicações está ao início do texto, no capítulo II. As três aplicações escolhidas são apresentadas nos capítulos III, IV e V. As conclusões e comentários finais estão no capítulo VI.

A primeira aplicação (capítulo III) demonstra como as transformadas bilineares podem ser usadas na determinação do amortecimento de sistemas harmônicos complexos. Após uma conceituação teórica necessária acerca das grandezas que definem o amortecimento e como estas relacionam-se com as técnicas tempo-frequência, será apresentada uma simulação a fim de verificar o método proposto bem como apresentar as principais formas de apresentação dos resultados. A seguir serão apresentados resultados da aplicação destas técnicas aos sinais obtidos de testes em dois modelos experimentais.

A segunda aplicação (capítulo IV) mostra como a transformada wavelet discreta pode ser usada para compressão de sinais e imagens. Após uma breve introdução acerca dos métodos usuais de compressão de dados é explicada a metodologia de compressão com

wavelets. Para verificar a aplicabilidade da teoria frente a situações reais, é realizada a compressão de alguns tipos de sinais que podem ser medidos em estruturas civis, offshore, etc. Estendendo a transformada unidimensional para o domínio bidimensional por um produto de tensores é mostrado um exemplo de compressão de imagens e suas implicações quando aplicada a matrizes resultantes de alguns métodos numéricos.

A terceira aplicação (capítulo V) mostra a utilização da teoria das wavelets à compressão de matrizes resultantes da discretização de equações integrais. Vários exemplos de aplicação são apresentados com diferentes tipos de condições de contorno para verificar sua influência no condicionamento numérico do sistema final (comprimido) de equações. O desempenho computacional do algoritmo de compressão é então avaliado em detalhe para cada caso estudado. A seguir é desenvolvida a teoria do mapeamento necessário para a compressão por blocos e apresentados alguns desenvolvimentos particulares do esquema, como paralelização e casos não-lineares.

## II TEORIA BÁSICA

É bem sabido que uma extensa classe de sinais pode ser expressa como a soma de uma série de senos e co-senos, conhecida como expansão em série de Fourier – a bibliografia resumida e uma cópia do artigo original de 1807 do barão francês Jean Baptiste Joseph Fourier (à direita) podem ser vistos em (GRATTAN-GUINNESS 1972). A primeira aplicação desta série foi para a resolução de problemas relacionados à transmissão de calor – para a qual a série foi originalmente desenvolvida. Mas a descoberta de que os coeficientes da série possuíam significado intrínseco foi feita não antes de 1860, 53 anos após a publicação do artigo original – que, por sinal, foi rejeitado várias vezes antes de ser publicado – ou 35 anos após a morte do autor em 1830.



Um pouco mais tarde KIRCHHOFF e BUNSEN (1860) observaram que o espectro luminoso poderia ser usado para reconhecimento, detecção e classificação de substâncias ao perceberem que cada substância, ao ser queimada, emite uma gama particular de frequências luminosas.

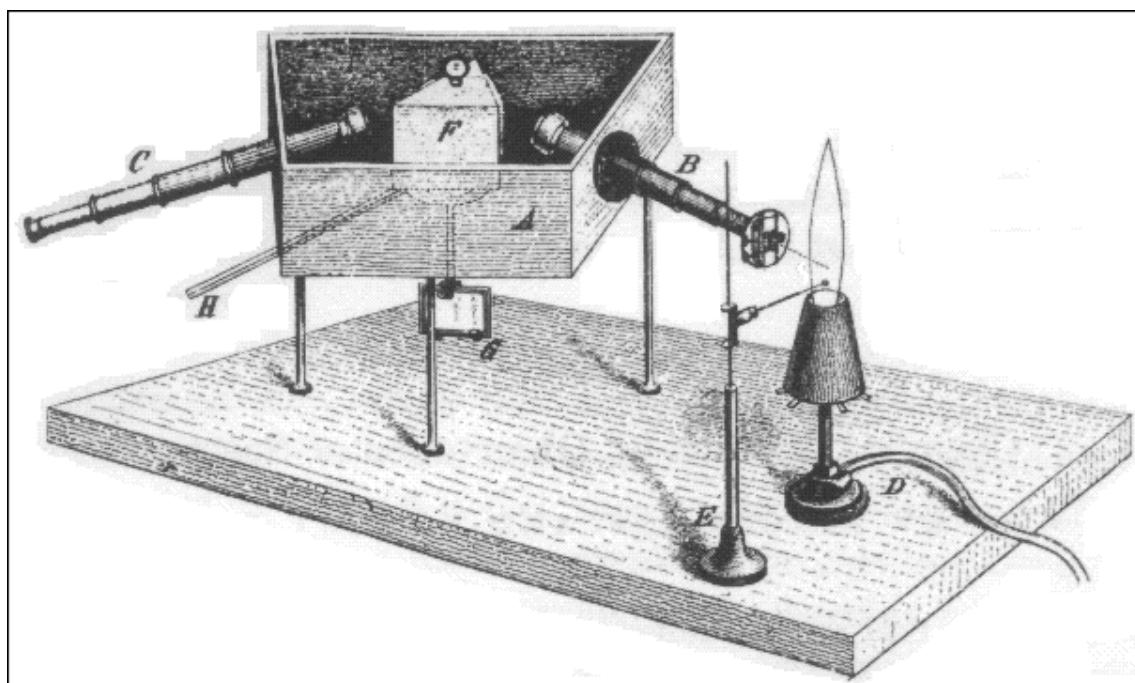


Figura 2 O espectroscópio de Kirchhoff e Bunsen

Kirchhoff e Bunsen não utilizavam a transformada de Fourier para obter o espectro luminoso das substâncias a serem analisadas. A Figura 2 mostra o aparato utilizado pelos cientistas. **A** é uma caixa internamente escura com um prisma trapezoidal fixo ao fundo. As duas paredes oblíquas internas, as quais formam um ângulo de cerca de 58 graus suportam os dois telescópios **B** e **C**. A ocular do telescópio **B** foi removida e substituída por uma placa na qual existe uma fenda formada por dois anteparos de bronze, posicionada exatamente no eixo do foco da objetiva.

A lâmpada **D** é posicionada antes da fenda, na direção do eixo do telescópio **B**. Um pouco abaixo do ponto onde o eixo encontra a chama está a extremidade de um arame de platina, o qual é mantido suspenso por um apoio **E**. Coloca-se um pequeno grão do componente químico a ser investigado soldado ou colado de alguma forma a esta extremidade.

Entre as lentes das objetivas **B** e **C** está o prisma **F** de ângulo interno de 60 graus e preenchido com disulfato de carbono. O prisma é sustentado por uma placa de bronze que pode rodar em torno de um eixo vertical. Este eixo está conectado a um espelho **G** abaixo e também a uma manivela **H** que serve para o operador rodar o conjunto. Um pequeno telescópio (não mostrado na figura) fica direcionado ao espelho de tal forma que faz a leitura de uma escala de “frequências” impressa, posicionada a pequena distância do conjunto.

Rodando-se o prisma com auxílio do manuseador **H** o espectro inteiro de frequências da chama pode ser trazida separadamente através do telescópio **C**. Cada posição do prisma corresponderá portanto a uma leitura de uma frequência do espectro. Após colocar a substância a ser analisada para ser queimada pela chama, o operador leva a manivela **H** até o limite de curso, no infravermelho e então segue lentamente rodando o conjunto em direção ao outro limite, o violeta. Cada vez que surge uma emissão através da objetiva **C** o operador pára e anota a respectiva frequência desta emissão com o auxílio do espelho **G** e da correspondente tabela, indicando também na anotação se a emissão é forte ou fraca.

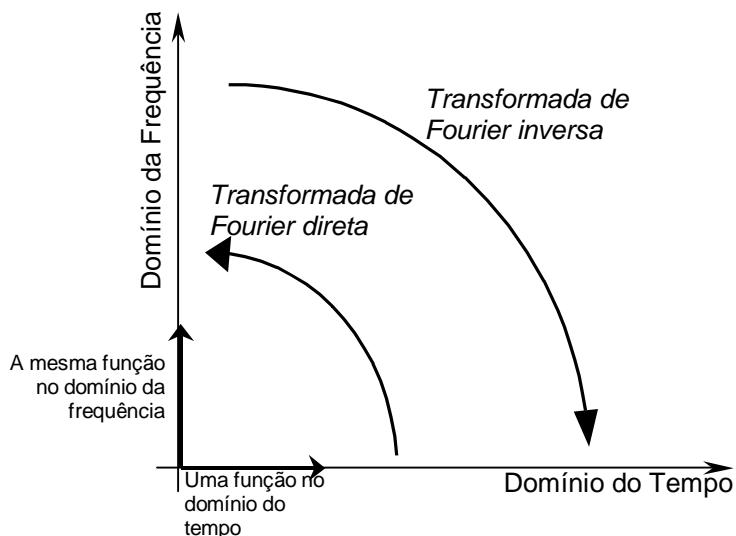
Bunsen e Kirchhoff determinaram que o lítio apresenta duas emissões principais: uma de cor amarela, fraca, e uma vermelha, forte. O potássio apresenta um espectro contínuo, com duas emissões distintas, uma na linha A do espectro solar e outra no extremo violeta, coincidindo com uma linha de Fraunhofer (sic). O cálcio possui uma forte emissão na faixa do verde, combinado com uma emissão média no laranja. O

estrôncio é dito como o mais fácil de ser identificado por possuir oito emissões distintas: seis em diferentes frequências do vermelho, uma laranja e uma azul.

Este apelo que a significação do espectro possui, aliado à propriedade da série de Fourier ser perfeitamente reversível, levou às denominações “domínio do tempo” e “domínio da frequência”, enfatizando o fato de que ambos os domínios serem apenas representações distintas do mesmo fenômeno, apenas em bases distintas do espaço euclidiano.

Fazendo-se uma analogia entre dois conhecidos espaços vetoriais reais a saber, o espaço constituído por todas as funções contínuas e com valores reais em um intervalo fechado  $[a,b]$ , notado por  $\subset[a,b]$  e o espaço unidimensional constituído por valores reais notado por  $\mathbb{R}$ , vê-se que, sob a teoria geral dos espaços vetoriais reais, ambos os espaços são marcadamente semelhantes. “Esta semelhança resulta do fato de que uma adição e uma multiplicação por números reais também podem ser definidas em  $\subset[a,b]$  e de que estas operações gozam das mesmas propriedades que as correspondentes operações em  $\mathbb{R}$ ” (KREIDER et al., 1972).

Assim sendo, as bases do espaço de funções, tempo e frequência, podem ser didaticamente ilustrados como na Figura 3, sendo a transformada de Fourier o operador que transporta uma série de um espaço para o outro de forma reversível.

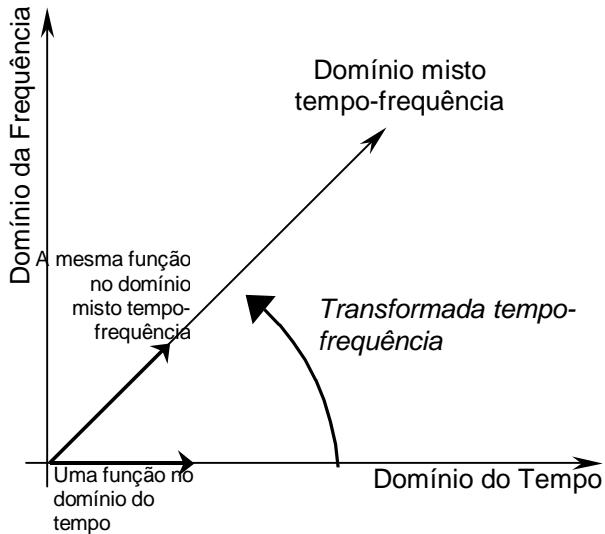


**Figura 3 Transformação de vetores entre os espaços tempo e frequência**

A desvantagem da representação no domínio da frequência é que esta não contém nenhuma informação temporal pois a base das funções de Fourier é ortogonal à base do tempo (MEYER, 1993, COHEN, 1995) como pode-se apreender com a ajuda do

esquema da Figura 3. Isto significa que, apesar de poder-se identificar todas as frequências que ocorrem no sinal, é impossível determinar a partir dos coeficientes calculados *quando* estas ocorrem. Esta tarefa – determinar-se o intervalo de ocorrência das frequências (ou *componentes*) presentes no sinal – é uma necessidade básica em várias áreas como o reconhecimento de voz, por exemplo (ATLAS, LOUGHLIN, PITTON, 1992, FLANDRIN, 1988, ZHAO, ATLAS, MARKS, 1990). Nas aplicações, ao final deste trabalho, mostra-se que um método bastante robusto de análise de amortecimento pode ser obtido desde que esta tarefa seja factível.

Em determinado momento surge então a necessidade de gerar-se uma transformada que leve os vetores do domínio do tempo para um domínio não-ortogonal a este, um domínio onde ambas as informações estivessem presentes, conforme ilustrado na Figura 4.



**Figura 4 Introdução da transformada tempo-frequência**

Deve ser observado que, como o novo domínio misto tempo-frequência não é ortogonal a nenhum dos domínios anteriores, tempo ou freqüência; os vetores neste novo domínio misto trazem informação de ambos os domínios. Portanto a solução do problema resume-se em obter uma transformada adequada que relacione os vetores no domínio do tempo a vetores no domínio misto tempo-frequência. Posteriormente será mostrado que existem duas formas de fazê-lo:

1. através da transformada Wavelet;
2. através das Distribuições Tempo-Freqüência (TFDs).

Embora não tenha sido a primeira das distribuições tempo-frequência (a primeira foi a distribuição de Wigner-Ville), indiscutivelmente a mais utilizada é o spectrograma. Esta distribuição deve sua popularidade atual à simplicidade e ao simples entendimento de seu significado – o de um “espectro instantâneo – que deriva da transformada de Fourier.

Dada a importância estratégica do spectrograma no entendimento da teoria das transformadas tempo-frequência, sua idéia básica será apresentada (item II.1) seguido da transformada wavelet (item II.2) e, finalmente, das distribuições tempo-frequência (item II.3).

## **II.1 O spectrograma**

O spectrograma, também chamado *Short Time Fourier Transform* (STFT), foi uma das primeiras técnicas tempo-freqüência e até hoje é a mais utilizada na análise de sinais transientes por ser rápida e ter interpretação simples, por ser uma derivação da transformada de Fourier.

A idéia básica do spectrograma é simples: ao invés de tomar-se a transformada de Fourier de todo o sinal de uma vez, divide-se o sinal em pedaços e então calcula-se o espectro de cada parte em separado. Esta divisão pode ser feita utilizando-se uma função de *enjanelamento* (ou *window*, em inglês)  $h(t)$ , centrada num instante  $t$ , que irá “fatiar” a função original  $f(t)$  em pedaços.

Para ilustrar melhor como este enjanelamento pode ser realizado, é apresentado, na Figura 6, um exemplo onde um sinal cuja freqüência instantânea é linearmente crescente – também conhecido como chirp linear – foi fatiado por uma janela retangular de largura total 0,25s, mostrada na Figura 5. Este processo de fatiamento seguido da análise espectral é intuitivo e pode fornecer um bom indicativo da dependência do conteúdo do espectro com o tempo.

Formalmente, a função “fatiada”  $f_t(\tau)$  pode ser descrita pela equação

$$f_t(\tau) = f(\tau)h(\tau - t) \quad (1)$$

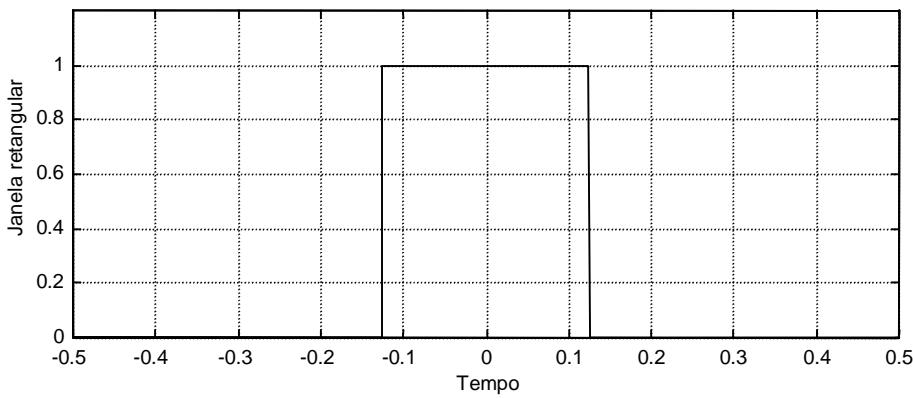
onde sua transformada de Fourier é dada por

$$F_t(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f_t(\tau) e^{-i\omega\tau} d\tau = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(\tau) h(\tau - t) e^{-i\omega\tau} d\tau \quad (2)$$

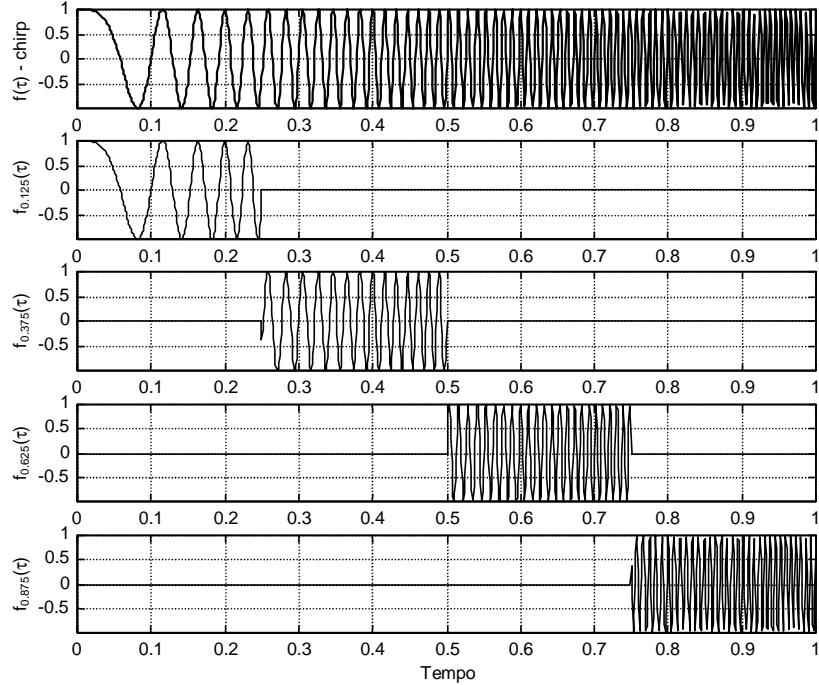
Define-se então o spectrograma como o conjunto dos quadrados dos módulos dos espectros das fatias

$$SP(t, \omega) = |F_t(\omega)|^2 \quad (3)$$

Na Figura 7 mostra-se um sinal no domínio do tempo e da freqüência que será mais útil para exemplificar a técnica do spectrograma. O sinal utilizado foi gerado por uma função matemática e sua frequência de digitalização é de 400 Hz ou, como alguns autores preferem, 400 pontos por segundo. É composto pela soma de dois sinais, um sinal harmônico com freqüência igual a 100 Hz e um pulso triangular com duração total 0.0025 s.

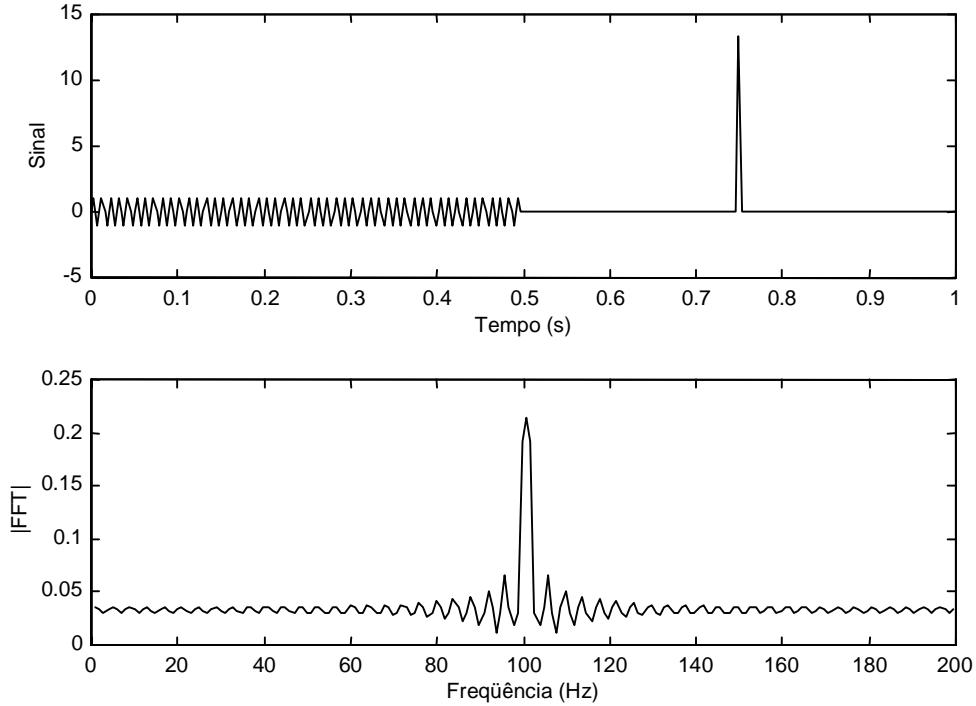


**Figura 5 Janela retangular**



**Figura 6 Sinal decomposto por janela retangular de largura 0,25s.**

Pode-se detectar através dos gráficos da Figura 7 quais e de que tipo são os componentes do sinal mas o momento em que estes ocorrem e sua duração são informações que não estão disponíveis.



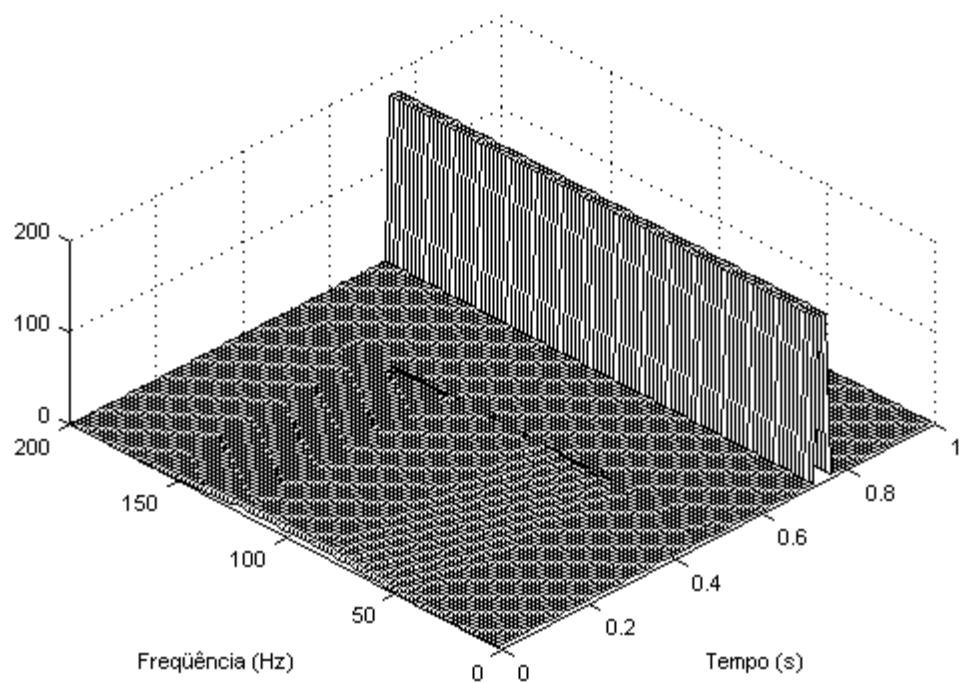
**Figura 7 Sinal utilizado composto por seno e um impacto**

Na Figura 8 é apresentado o spectrograma do sinal onde foi usada uma função de recorte, também denominada por “janela”, com curta duração (0.01 s). Por “curta” entende-se que o tamanho da janela é pequeno em relação à duração total do sinal, no caso 1%. Como o spectrograma é uma função bidimensional (tempo e frequência), a figura apresenta um gráfico 3D onde os eixos são: freqüência, tempo e amplitude da distribuição.

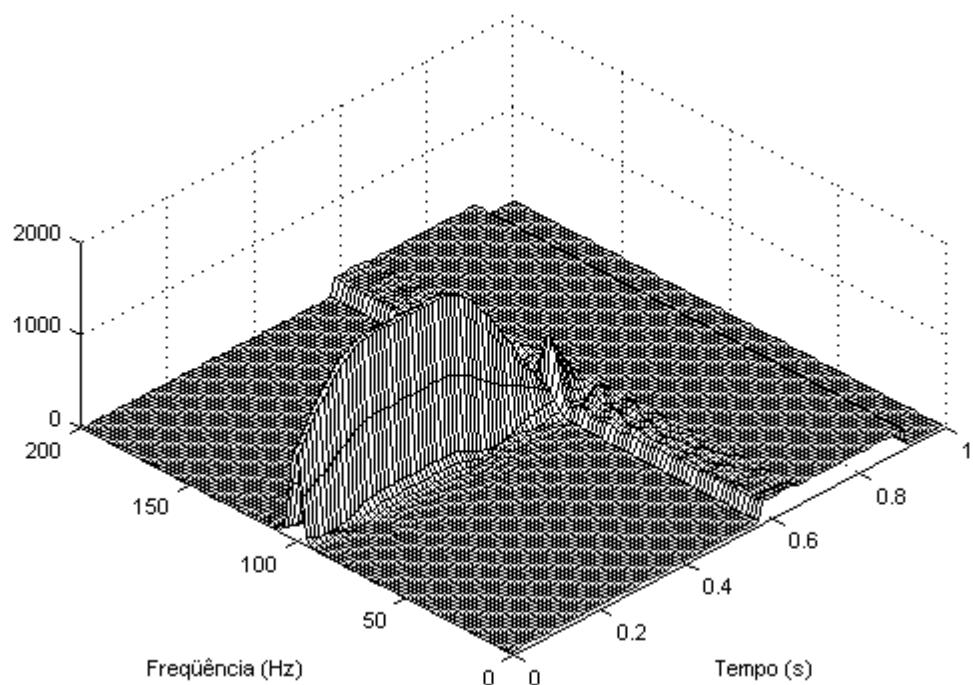
É de uso corrente (COHEN 1995, QIAN, CHEN, 1996, CUNNINGHAM, WILLIAMS, 1996, STANKOVIC, 1997), não se indicar explicitamente que o eixo vertical corresponde à amplitude da distribuição pois considera-se esta informação redundante já que os outros dois eixos já estão indicados. Isto também evita que o aplicativo gráfico reserve espaço para este texto, o que em certos casos prejudica bastante a visualização. Para isto deve-se explicitar corretamente no texto qual a distribuição tempo-freqüência que está sendo apresentada.

Verifica-se na Figura 8 que a STFT capta perfeitamente a presença do pulso triangular mas o sinal harmônico está quase imperceptível. Esta má representação do harmônico pode ser corrigida aumentando-se a duração da janela de 0,01s (1%) para 0,16s (16%).

Espera-se com isto que o sinal harmônico passe a aparecer com maior amplitude na distribuição. O resultado obtido é apresentado na Figura 9.



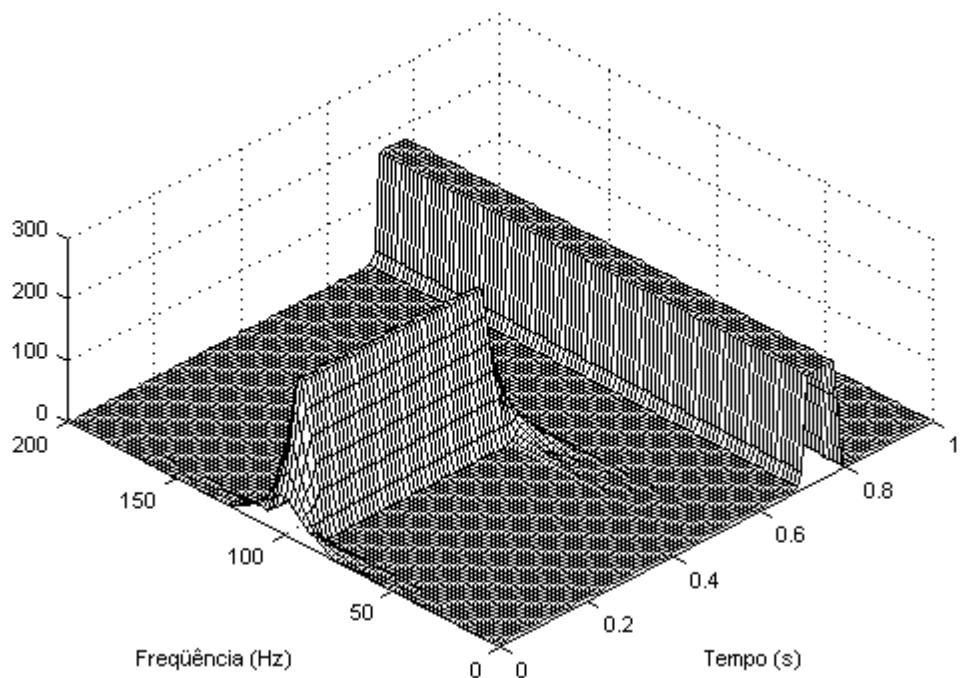
**Figura 8 Espectrograma com janela curta ( $T = 0,01\text{s} \equiv 1\%$ )**



**Figura 9 Espectrograma com janela larga ( $T = 0,16\text{s} \equiv 16\%$ )**

Como pode-se observar na Figura 9 o aumento da janela permite que o sinal harmônico apareça na distribuição, mas às custas de um grande espalhamento do sinal de impacto. Este é o grande paradoxo. Se a janela usada tem duração grande, perde-se resolução no tempo e ganha-se resolução em freqüência. Por outro lado, se a janela é curta, perde-se em freqüência e ganha-se no tempo. A regra então é: quanto mais curta a janela, mais perto do domínio no tempo; quanto mais larga, mais perto do domínio da freqüência. O ideal então é tentar-se buscar para cada tipo de sinal um meio-termo, de maneira a melhor representar o sinal estudado. Voltando ao exemplo e utilizando agora uma janela de 0,04 s (4%) obtém-se uma melhor representação tanto do sinal harmônico quanto do impacto, conforme pode ser visto na Figura 10.

O paradoxo apresentado é melhor descrito pelo *princípio da incerteza* de Heisenberg (WIGNER, 1932, WIGNER, 1971), o qual, traduzido da mecânica quântica para o jargão do processamento de sinais, afirma ser *impossível* saber com precisão arbitrária qual a exata freqüência e o instante de ocorrência de um componente de sinal. Como mostrado nas três figuras anteriores (Figura 8, Figura 9 e Figura 10) é impossível obter precisão simultânea em ambos os domínios – sempre existe uma perda em um domínio opondo-se a um ganho de resolução no outro.



**Figura 10 Espectrograma com janela média ( $T = 0,04\text{s} \equiv 4\%$ )**

O tamanho da janela do espectrograma pode ser relacionado com o ângulo formado entre o eixo do domínio do tempo e o eixo do domínio misto (Figura 4). Quanto menor o tamanho da janela, menor o ângulo e mais perto do domínio do tempo. Inversamente, quanto maior o tamanho da janela, maior o ângulo e mais perto do domínio da frequência.

Como se verá adiante, as técnicas bilineares (as distribuições tempo-frequência ou TFDs) serão bem mais eficazes, isto é, melhores que o espectrograma no momento de gerar um gráfico deste tipo, como pode ser visto mais adiante na Figura 24. Apesar disso o espectrograma continuará a ser mais eficiente, ou seja, realizará um trabalho de menor qualidade porém num tempo bastante menor.

## ***II.2 A transformada wavelet***

A *transformada wavelet* é provavelmente a mais recente solução no sentido de recortar o sinal de forma coerente. Ao invés de deixar indefinida a largura da função de recorte a ser utilizada, a análise wavelet provê um conjunto de funções de recorte escaláveis que resolvem completamente a questão do recorte. A análise wavelet é feita em várias etapas e inicia com uma função de recorte larga que desliza ao longo do sinal e, em cada posição que ocupa, é calculado o espectro do sinal recortado resultante. Este processo de deslize/recorte/espectro é repetido várias vezes com janelas cada vez mais curtas. O resultado final é um conjunto de vários espectros agrupados relativamente a uma largura diferente da função de recorte (resolução). Por ser realizada em vários níveis de resolução (ou larguras ou escalas), diz-se que esta é uma análise de multi-resolução ou multi-escala (*multiscale/multiresolution analysis*, em inglês).

Até este ponto parece ter sido um grande avanço que a teoria das wavelets resolva o problema de recorte do sinal encarregando-se de escolher e aplicar as escalas da forma apropriada. Mas este avanço pode ser considerado irrelevante se comparado às propriedades das séries *wavelet*. Ao contrário da série de Fourier, a série wavelet não é única. Existem várias séries que são agrupadas por famílias. Cada família possui um conjunto de propriedades úteis como ortogonalidade, localização no tempo e em frequência, conservação da energia ou perfeita representação de polinômios. Com uma vantagem: se uma família de wavelets não possui determinada propriedade, simplesmente cria-se uma nova família.

Assim como a série de Fourier, as wavelets apresentam uma transformada rápida que, no entanto, costuma ser várias vezes mais rápida que a FFT (a transformada rápida de Fourier ou *Fast Fourier Transform*) por envolver somente operações algébricas, nunca calculando-se funções transcendentais (seno, cosseno, exponenciais, etc). As séries são, no entanto, um assunto que será introduzido na seção III.5.3. Por enquanto será introduzida a transformada contínua e suas propriedades.

### A transformada wavelet contínua

A análise wavelet descrita suscintamente no item anterior é conhecida como *transformada wavelet contínua* (CWT ou *Continuous Wavelet Transform*). É formalmente descrita pela equação

$$\gamma(s, \tau) = \int f(t) \cdot \psi_{s,\tau}^*(t) \cdot dt \quad (4)$$

onde o asterisco indica o complexo conjugado. Esta equação mostra como uma função  $f(t)$  é decomposta em coeficientes  $\gamma(s, \tau)$  por uma base de funções  $\psi_{s,\tau}(t)$  denominadas *wavelets*. As variáveis  $s$  (escala) e  $\tau$  (translação imposta) são as dimensões do domínio wavelet. A transformada inversa é dada por

$$f(t) = \iint \gamma(s, \tau) \psi_{s,\tau}(t) ds d\tau \quad (5)$$

As wavelets  $\psi_{s,\tau}(t)$  (funções da base wavelet) são geradas a partir de uma única wavelet  $\psi(t)$ , denominada *wavelet-mãe* aplicando-se um escalamento e uma translação da forma

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t-\tau}{s}\right) \quad (6)$$

onde fator  $1/\sqrt{s}$  foi inserido por uma questão de normalização de energia.

É importante notar que em nenhuma das três equações anteriores as wavelets  $\psi_{s,\tau}(t)$  foram explicitamente definidas. A teoria geral das wavelets somente exigirá três propriedades: admissibilidade, regularidade e ortogonalidade. Estas propriedades serão insuficientes para definir a wavelet-mãe, de forma que existe espaço para a inserção de novas propriedades, gerando-se assim diversas famílias de wavelets.

### Propriedades essenciais: admissibilidade e regularidade

Nem todas as funções podem ser wavelets, por dois motivos: primeiramente, a análise wavelet pretende ser um tipo de análise espectral e, portanto, as funções de análise

precisam ter caráter oscilatório, isto é, precisam ser ondas (*waves*). Esta é a condição chamada de *admissibilidade* e pode ser expressa pela simples equação

$$\int \psi(t) dt = 0 \quad (7)$$

que significa simplesmente que as funções da base precisam ter média nula excluindo-se, evidentemente, a função trivial  $\psi(t) = 0$ . Esta condição implica que a transformada de Fourier da wavelet seja nula na origem, isto é,  $\Psi(\omega) = 0$  se  $\omega = 0$ .

Em segundo lugar, para que a análise forneça localização simultânea no tempo e em frequência, as funções da base também precisam ter localização no tempo e em frequência. Esta condição é denominada *regularidade* e tem um tratamento matemático bastante complexo. Em suma, o que se faz é exigir-se que a transformada de Fourier da wavelet seja nula em exatamente  $N$  pontos do eixo das frequências além da origem, isto é,  $\Psi(\omega) = 0$  para  $\omega = \omega_1, \omega_2, \dots, \omega_N$ . Isto assegurará que a wavelet tenha localização em frequência bem como localização no tempo, isto é,  $\psi(t) \rightarrow 0$  quando  $t \rightarrow \pm\infty$ . Quanto maior o número  $N$ , também denominado como *número de momentos nulos* (*vanishing moments*) ou *ordem* da wavelet, maior a localização em frequência e menor a localização no tempo, isto é, mais espalhada no tempo e mais concentrada em frequência será a wavelet.

Estas propriedades são as únicas a serem exigidas para que uma função seja candidata a base para uma análise wavelet contínua. Somente com estas definições teóricas muitos trabalhos já foram publicados e análises foram feitas mas as melhores aplicações das wavelets começam a aparecer quando discretiza-se a transformada contínua dando origem à transformada wavelet discreta, comentada a seguir.

### A terceira propriedade: (bi) ortogonalidade

A terceira propriedade diz respeito à possibilidade de reverter a transformada contínua conforme expressa pela equação (5).

Até o presente ponto não foi dada nenhuma garantia de que a transformada contínua (4) e (5) fosse reversível. A condição necessária e suficiente para a perfeita reconstrução foi apresentada em (DAUBECHIES,1992) e tem a forma de uma dupla inequação

$$A \leq \frac{\|\gamma\|^2}{\|f\|^2} \leq B \quad (8)$$

onde

$$\|f\|^2 = \int |f(t)|^2 dt \quad (9)$$

é energia, ou norma euclidiana  $L^2$  da função  $f$ , e

$$\|\gamma\|^2 = \iint |\gamma(s, t)|^2 ds \cdot dt \quad (10)$$

é norma euclidiana da transformada wavelet da função  $f$ .  $A$  e  $B$  são constantes reais, finitas e independentes da função  $f$ , isto é, são dependentes apenas da família wavelet usada na transformada.

O significado da inequação (8) é que, seja qual for a função  $f$ , a razão entre as normas, da transformada e da função, deverá estar sempre limitada por extremos finitos. Sendo atendida esta condição, a transformada é garantida ser perfeitamente reversível. Quando  $A=B$  a transformada é dita *ortogonal*. Quando  $A=B=1$  a transformada é dita *ortogonal e normalizada* ou simplesmente *ortonormal*. Quando  $A \neq B$  a transformada é dita *biortogonal*.

Na verdade poderia-se perguntar porque não exigir-se logo que  $A=B=1$ , quando se teria uma base ortonormal, garantida a perfeita reconstrução e também a conservação de energia? De fato isto pode ser alcançado e durante os primeiros anos após a descoberta de Daubechies a procura por outras famílias ortonormais dominou o centro das atenções. No entanto, existe um limite para a quantidade de propriedades que se pode exigir de uma família de wavelets. Além da admissibilidade, da regularidade e da reversibilidade (ortogonalidade, ortonormalidade ou biortogonalidade) poucas propriedades podem ser inseridas até que a família esteja totalmente definida, ou seja, até que a wavelet-mãe  $\psi(t)$  seja uma função conhecida, completamente determinada. Quando as propriedades excedem este limite, algumas prioridades devem ser revistas e, acentue-se, ortogonalidade e normalização são conceitos elegantes mas pouco úteis em certas aplicações. De fato, quando aplicadas como pré-condicionadores para a resolução de equações diferenciais parciais (EDPs), são utilizadas com sucesso wavelets B-Splines, que são apenas biortogonais.

## A Transformada Wavelet Discreta

Assim como a transformada de Fourier necessita de sua forma discreta (a DFT ou *Discrete Fourier Transform*) para tornar-se útil computacionalmente, necessita-se de uma forma discreta (DWT ou *Discrete Wavelet Transform*) para a transformada wavelet. Melhor ainda, o que se deseja é uma série wavelet similar à série de Fourier

que forma uma base completa, não-redundante nem insuficiente. Isto implica em que a dimensão da base do domínio transformado (o número de coeficientes) deve ser igual à dimensão da base do domínio temporal (o número de pontos do sinal). Também é desejável que a DWT tenha uma transformada rápida (FWT ou *Fast Wavelet Transform*). Nas seções a seguir se verificará a satisfação de todas estas expectativas mas para tanto será necessário remover toda a redundância da transformada contínua (CWT).

A redundância citada deve-se basicamente a dois motivos: (1) a transformada é contínua e precisa ser discretizada como um somatório e (2) este somatório precisa ter um número finito de termos. Mais exatamente, precisa ter um número de coeficientes (parcelas) igual ao número de pontos do sinal. Inicia-se portanto com a discretização e deixa-se a redução de termos do somatório para a próxima seção.

A tarefa de discretizar a transformada (4) consiste em nada mais que discretizar as variáveis de domínio  $s$  (escalamento) e  $\tau$  (translação), substituindo-as pelas versões discretas  $j$  (escalamento) e  $k$  (translação) de tal forma que

$$s = s_0^j \quad \text{e} \quad \tau = k\tau_0 s_0^j \quad (11)$$

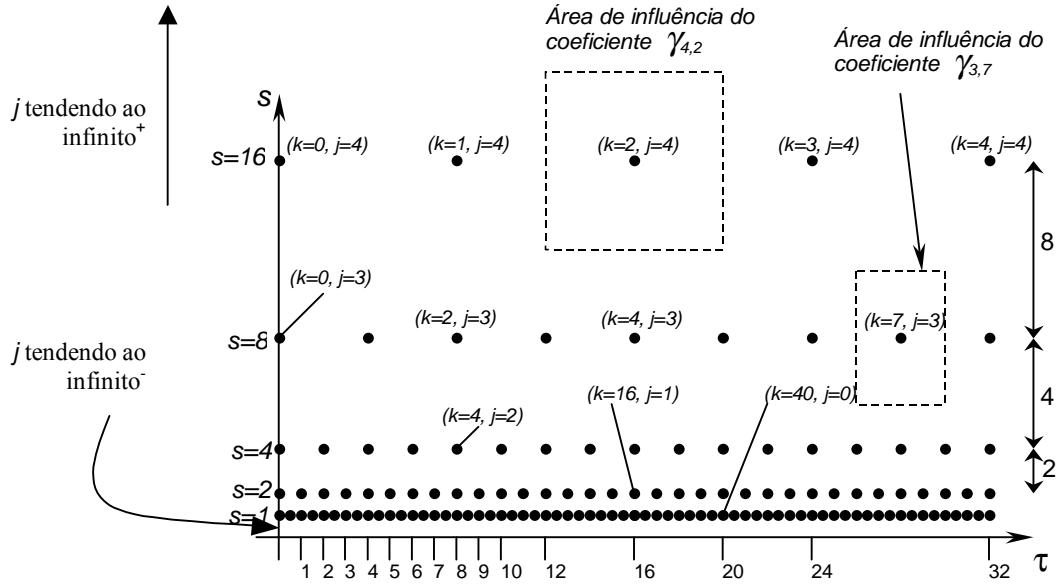
onde  $\tau_0$  e  $s_0$  são constantes que precisam ser definidas. A escolha unanimemente adotada é  $s_0 = 2$  e  $\tau_0 = 1$  pois é a mais natural tanto no sentido computacional quanto da percepção humana (MEYER, 1993). Desta forma as funções da base (wavelets) tomam a forma usual

$$\Psi_{j,k}(t) = \frac{1}{\sqrt{2^j}} \Psi\left(\frac{t}{2^j} - k\right) \quad (12)$$

Deve ser observado que somente o domínio wavelet foi discretizado e não o domínio do tempo, que permanece contínuo.

A localização dos coeficientes discretos no espaço transformado tempo-escala pode ser melhor visualizado com a ajuda do esquema da Figura 11, onde são representados por pontos sobre o domínio tempo-escala. Nesta figura só foram mostrados os coeficientes para  $j=0$  até  $j=4$ . Os coeficientes para  $j \rightarrow -\infty$  não foram mostrados mas estão concentrados entre o eixo do tempo ( $s=0$ ) e  $s=1$ , ou seja, estão ainda dentro da figura mas não estão visíveis por uma questão óbvia de limitação gráfica. Já os coeficientes para  $j \rightarrow +\infty$  estão localizados fora do gráfico, acima e para fora da página. Cada coeficiente tem uma área de influência como mostrado para os coeficientes  $\gamma_{4,2}$  e  $\gamma_{3,7}$ .

Quanto maior a escala discreta  $j$ , maior será a área de influência do coeficiente, tanto no tempo quanto na escala contínua  $s$ .



**Figura 11 Localização dos coeficientes da série wavelet no domínio transformado (wavelet)**

Utilizando a discretização sugerida, a série wavelet toma a forma discretizada da transformada wavelet inversa (5):

$$f(t) = \sum_{j=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} \gamma_{j,k} \cdot \psi_{j,k}(t) \quad (13)$$

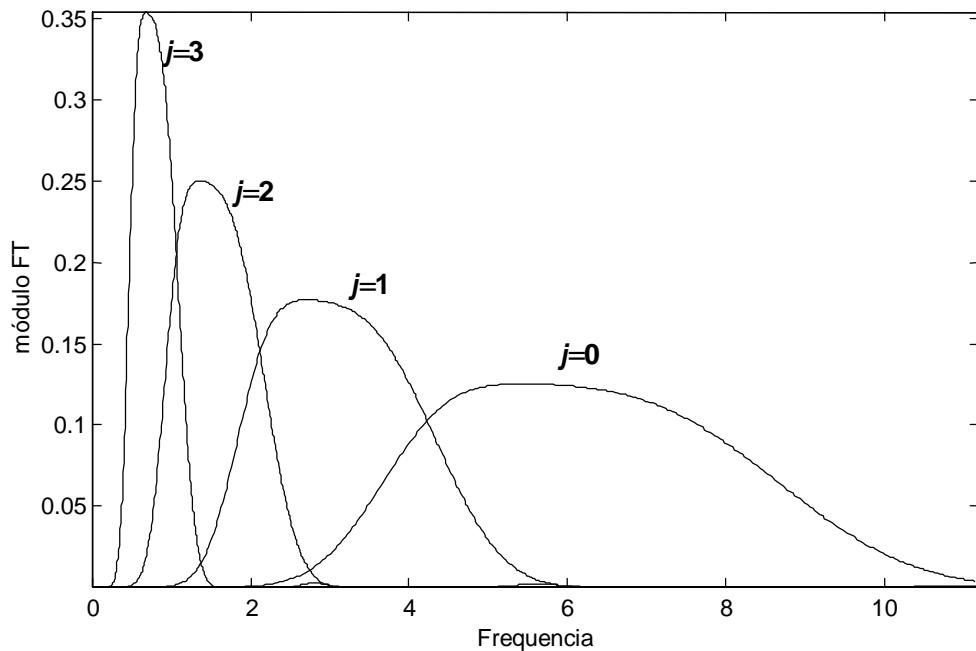
onde os coeficientes  $\gamma_{j,k}$  da série wavelet são calculados através da forma discretizada da transformada direta (4):

$$\gamma_{j,k} = \int f(t) \cdot \psi_{j,k}^*(t) \cdot dt \quad (14)$$

Analizando-se a série wavelet (13), nota-se que as duas variáveis  $j$  (escala) e  $k$  (translação) têm somatório de  $-\infty$  a  $+\infty$ . No entanto, todos os sinais obtidos por digitalização de uma grandeza física contínua são finitos no tempo. Desde que é necessário compatibilizar o sinal existente com a teoria disponível que exige integração até o infinito, assume-se que o sinal seja nulo, de valor zero, fora do domínio conhecido. Isto significa que o índice  $k$  da série wavelet, responsável pela translação, até poderá seguir viajando indefinidamente pelo eixo temporal mas todos os coeficientes  $\gamma_{j,k}$  resultantes serão nulos pois o sinal será, por convenção, zero em tais regiões. Portanto uma parte do problema está resolvido com a limitação automática do índice  $k$  à região coberta pelo sinal.

A segunda parte do problema é: quantas escalas ( $j$ ) são necessárias? Neste caso a resposta vem em duas partes:

Quanto ao limite inferior ( $j \rightarrow -\infty$ ), correspondente à diminuição do suporte (domínio efetivo) da wavelet, fica claro que uma wavelet  $\psi_{j,k}(t)$  não pode ser menor que o período de digitalização do sinal. Isto equivale a que, em frequência, não possam existir coeficientes  $\gamma_{j,k}$  cuja respectiva wavelet  $\psi_{j,k}(t)$  tenha frequências superiores à taxa de digitalização. Isto pode ser melhor apreendido com a ajuda da Figura 12, onde estão plotadas as transformadas de Fourier das wavelets  $\psi_{j,k}(t)$  da família Daubechies-8 (DAUBECHIES, 1988).



**Figura 12 Transformada de Fourier das wavelets  $\psi_{j,k}(t)$  da família Daubechies-8**

Formalmente, o módulo da transformada de Fourier destas funções, relativa à wavelet-mãe  $\psi(t)$  é

$$|\Psi_{j,k}(\omega)| = \sqrt{2^j} \cdot |\Psi(2^j \omega)| \quad (15)$$

onde  $\Psi(\omega)$  é a transformada de Fourier da wavelet-mãe  $\psi(t)$  e  $\Psi_{j,k}(\omega)$  é a transformada de Fourier da wavelet  $\psi_{j,k}(t)$ . O efeito da diminuição da escala, como pode ser visto na figura, é o “escorregamento” da transformada de Fourier em direção ao infinito. Como se verá mais adiante, a numeração das escalas é totalmente arbitrária e, portanto, convencionou-se neste ponto, e sem perda de generalidade, que a *menor* escala duma análise wavelet será sempre a de número zero ( $j=0$ ).

Quanto ao limite superior ( $j \rightarrow +\infty$ ), que corresponde ao aumento do suporte (domínio efetivo) da wavelet, o limite verdadeiramente não existe, ou seja, se for fixado arbitrariamente um limite superior para a escala  $j$ , informação relevante certamente será perdida. O efeito do aumento da escala sobre a transformada de Fourier da wavelet é que esta vai sendo “comprimida” contra a origem, o que pode ser claramente percebido na Figura 12. Este processo parece prosseguir indefinidamente e interromper esta sequência não parece ser uma boa idéia pois o valor máximo da transformada aumenta a cada acréscimo na escala devido à normalização de energia entre escalas.

Felizmente a teoria das séries infinitas diz que existem séries infinitas que convergem para funções conhecidas como a função seno, por exemplo, que nos computadores é implementada como uma série polinomial infinita.

O comportamento sem limites da série em direção ao zero se parece bastante com o paradoxo de Zeno e Elea (490 A.C.). Diz ele que uma pessoa, para viajar da cidade A até a cidade B precisa viajar antes metade da distância entre as cidades. Mas para viajar esta metade ela precisa viajar a metade desta metade, ou seja, o primeiro quarto da distância que, por sua vez, necessitará que seja percorrida sua primeira metade, ou seja, o primeiro oitavo e assim por diante. Desta forma conclui-se que esta pessoa nunca chegará à cidade B pois sempre necessitará antes de passar por um número infinito de pontos. Algum pensador mais pragmático quebraria este impasse simplesmente dando um passo à frente, num único movimento que percorreria um número infinito de pontos e transformaria o resto da viagem numa sequência finita de pontos, resolvendo o problema. Na teoria das wavelets este primeiro passo é denominado *aproximação* e a função responsável por isso é notada por  $\phi(t)$  e denominada *função de aproximação* ou *função de escala* ou ainda *wavelet-pai*. É para esta função que se fará uma parte da série wavelet infinita convergir.

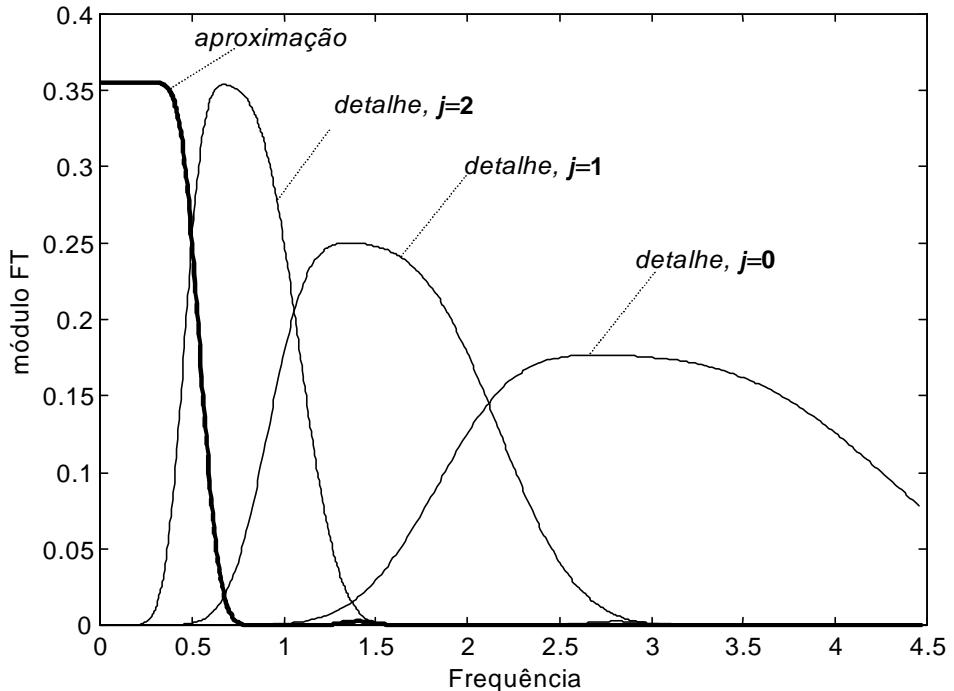
Não é qualquer função que pode candidatar-se para tal tarefa. A condição suficiente e necessária para realizá-la é que a função de escala obedeça a seguinte fórmula de recorrência:

$$\phi(t) = \sum_k a_k \phi(2t - k) \quad (16)$$

onde o conjunto dos coeficientes  $a_k$  é conhecido como *máscara da função de aproximação* ou simplesmente *máscara da wavelet*.

Deve-se acentuar que a máscara da wavelet traz embutida em si todas as características das famílias assim como os coeficientes dos filtros digitais trazem embutidos em seus valores todas as suas propriedades como frequência de corte, precisão nas bandas de passagem e de parada, inserção de fase, etc. É como uma assinatura digital particular da família wavelet.

Esta analogia com os filtros digitais vai mais além e será objeto de estudo na próxima seção. Pode-se afirmar por enquanto que a função de aproximação  $\varphi(t)$  age como um filtro passa-baixa e as wavelets  $\psi_{j,k}(t)$  como filtros passa-faixa, o que pode ser melhor compreendido com a ajuda do gráfico da Figura 13.



**Figura 13 Funções da base wavelet em frequência, evidenciando o caráter de “fechamento” da função de aproximação  $\varphi(t)$**

Desta forma a função de aproximação “cobre” as baixas frequências desde o zero até uma determinada frequência enquanto as wavelets, também chamadas de *funções de detalhe* exatamente por este motivo, “cobrem” faixas do espectro das frequências mais altas, ou seja, vão acrescentando detalhes a cada escala adicionada. No caso particular da Figura 13 resolveu-se parar na escala  $j=2$  e então fechar o espectro com a função de aproximação. Mas poderia-se continuar acrescentando escalas para  $j=3, 4, 5$  até quando se desejar ou até a frequência mínima do sinal ser atingida (nominalmente =  $1/(N\text{pontos} * \Delta t)$ ) e então aplicar-se o fechamento com a função de aproximação. Isto imporá às escalas um limite superior, notado por  $j_{max} = J$ , o qual será discutido na próxima seção.

Por enquanto basta entender como as wavelets e a função de aproximação dividem o espectro e como esta divisão restringe as escalas  $j$  a um intervalo finito dentro do conjunto dos números inteiros.

Como último passo falta somente atualizar a série wavelet de forma a refletir as últimas considerações. Considerando que as escalas das wavelets estão limitadas ao intervalo  $0 \leq j \leq J$  e que as parcelas relativas às maiores escalas foram substituídas pela função de aproximação, a série wavelet (13) – agora finita – toma a forma final

$$f(t) = \sum_k \lambda_k \phi(t - k) + \sum_{j=0}^J \sum_k \gamma_{j,k} \psi_{j,k}(t) \quad (17)$$

onde os novos coeficientes  $\lambda_k$  são conhecidos como *coeficientes de aproximação* e são calculados da forma usual, com o produto interno

$$\lambda_k = \int f(t) \phi(t - k) dt \quad (18)$$

### A transformada wavelet rápida (FWT)

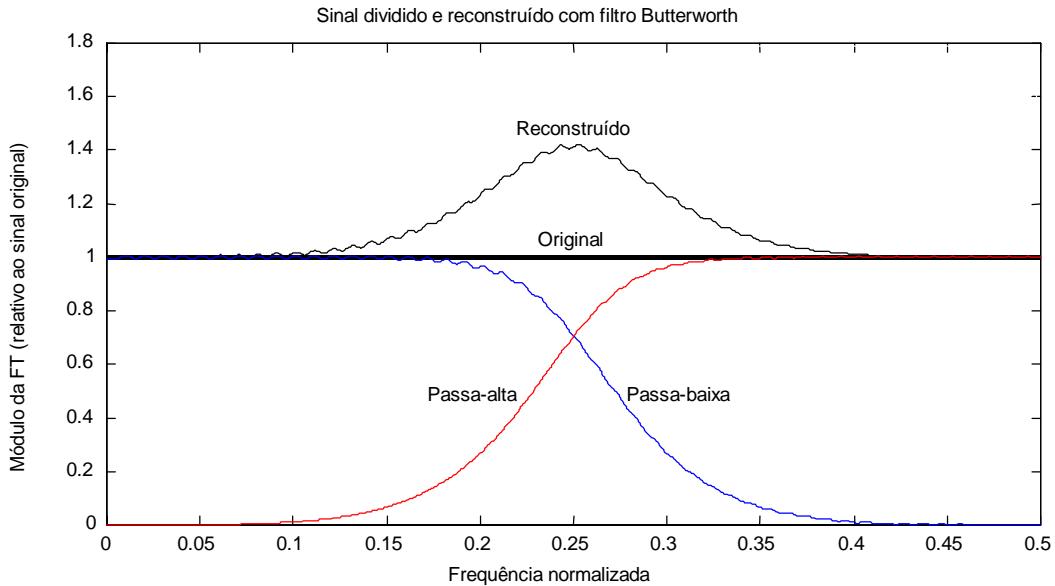
Neste ponto será feito um salto sobre a teoria. O motivo para isto é que o conteúdo ultrapassado consiste de um extenso desenvolvimento envolvendo simples álgebra matricial que, no final das contas, acrescentará somente um conceito novo: as wavelets contam com uma transformada rápida que baseia-se num esquema particular de filtragem em cascata, que será objeto de estudo desta seção. Assim como ninguém precisa saber em profundidade os detalhes do algoritmo da transformada rápida de Fourier (FFT) para utilizá-la, o mesmo aplica-se à transformada wavelet rápida. Os detalhes deste desenvolvimento podem ser obtidos em MALLAT (1986).

Este processo é bastante similar a um tipo de metodologia utilizada há algum tempo em processamento de imagens, denominada *subband coding*, que o autor não se arrisca a traduzir. A idéia básica de *subband coding* é separar o espectro do sinal (ou imagem) em faixas (bandas), tal qual um módulo equalizador de som faz. Uma forma trivial de fazê-lo é construir vários filtros passa-faixa e então aplicá-los todos ao sinal. A desvantagem deste processo é que o projeto de um filtro, principalmente o cálculo dos coeficientes (*taps*) é uma tarefa computacionalmente intensiva, não raramente exigindo vários ciclos de iteração até a convergência dos coeficientes. Em algumas aplicações, o tempo exigido para o cálculo dos coeficientes poderia chegar a algumas ordens de grandeza maior que o tempo exigido para a operação de filtragem do sinal em si, especialmente se o número de bandas for grande.

Uma forma mais simples é construir apenas dois filtros: um passa-baixa (LP ou *Low Pass*) e um passa-alta (HP ou *High Pass*) que são aplicados ao sinal, gerando dois sinais “filhos”, um com conteúdo de baixas frequências, denominado *aproximação*, e o outro com o conteúdo das altas frequências, denominado *detalhe*, o que dividirá o espectro total do sinal em duas metades de igual largura de banda. Como estes sinais estão com folga no espectro, não existe necessidade de mantê-los integralmente na memória (do computador). Se o sinal contém frequências apenas até a metade da taxa de Nyquist (ou um quarto da taxa de digitalização) como ocorre com o sinal de aproximação, então pode-se dobrar o período de digitalização eliminando-se um em cada dois pontos do sinal, processo denominado *decimação*, sem perda de informação relevante pois a taxa de Nyquist foi respeitada. O mesmo se aplica ao sinal de detalhe que contém apenas a metade superior do espectro, embora a prova do raciocínio seja um pouco diverso. Desta forma substitui-se o sinal original com  $N$  pontos por dois sinais equivalentes, a aproximação e o detalhe, cada um com  $N/2$  pontos e responsável por compor cada metade do espectro original. Percebe-se que o número de pontos do sinal deve ser par para que o esquema citado seja realizado, o que não é difícil de conseguir desde que sempre se pode acrescentar um zero ao final do sinal, no caso de seu tamanho ser ímpar.

Observe o caso da Figura 14. Um sinal arbitrário foi separado em dois sinais distintos usando-se um par de filtros tipo Butterworth, um passa-baixa e o outro passa-alta, ambos projetados para ter a mesma frequência de corte. Após calculados, os dois sinais resultantes foram então somados, obtendo-se um sinal dito “reconstruído”. Observa-se que o sinal original, representado por uma linha espessa tem valor constante 1 para todas as frequências enquanto o sinal reconstruído tem valor maior que um, principalmente por volta da frequência 0,25.

Alguém poderia desejar construir um filtro que não apresentasse este tipo de irregularidade, isto é, cuja transformada de Fourier do sinal reconstruído fosse exatamente igual à do sinal original. Se houver sucesso nesta empreitada, terá sido construído um par de filtros ditos *ortogonais*.



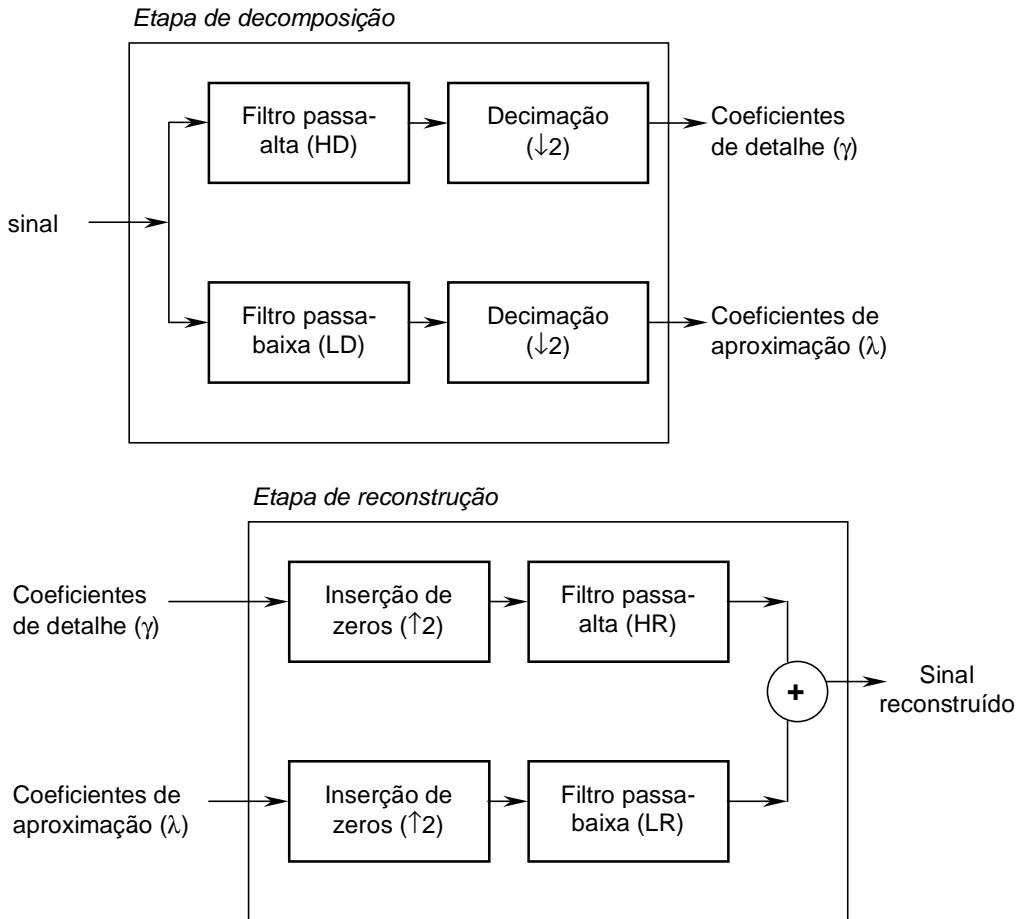
**Figura 14 Subdivisão e posterior reconstrução do espectro em duas bandas utilizando-se um filtro digital comum**

A transformada wavelet rápida envolve não apenas um, mas *dois* pares de filtros ortogonais do tipo resposta finita (FIR ou *Finite Impulse Response*): um par utilizado na decomposição (denominados LD e HD) e outro par usado na reconstrução (LR e HR). Se, adicionalmente, os filtros (LD,HR) e (HD,LR) também forem ortogonais então a transformada wavelet é dita ortogonal. Se não, a transformada wavelet é dita biortogonal que, acentue-se, é uma exigência mais fraca que a ortogonalidade.

A transformada rápida de Fourier (FFT) e a transformada rápida wavelet (FWT), além das semelhanças teóricas, compartilham também alguns detalhes na implementação computacional. Analogamente à *butterfly* (COOLEY, TUKEY, 1965) da transformada rápida de Fourier (FFT), a transformada rápida wavelet também tem um ciclo básico de operação, na verdade dois *duais*, que estão esquematizados na Figura 15. Nesta figura são apresentados dois blocos: o de decomposição e o de reconstrução.

O bloco de decomposição começa por aplicar dois filtros ortogonais, HD e LD já citados, obtendo-se duas novas séries de coeficientes. Como estas séries são redundantes, pode-se eliminar um de cada dois coeficientes calculados, obtendo-se então os coeficientes de detalhe ( $\gamma$ ) e os coeficientes de aproximação ( $\lambda$ ) desta etapa. As novas séries de coeficientes geradas terão tamanho  $N/2$  por causa da decimação, onde  $N$  é o número de pontos do sinal original, de forma que o número de coeficientes da base permanece inalterado após a decomposição, ou seja,  $N/2$  (aproximação) +  $N/2$  (detalhe) =  $N$  (tamanho original). Isto ocorre porque a operação de filtragem descrita consiste

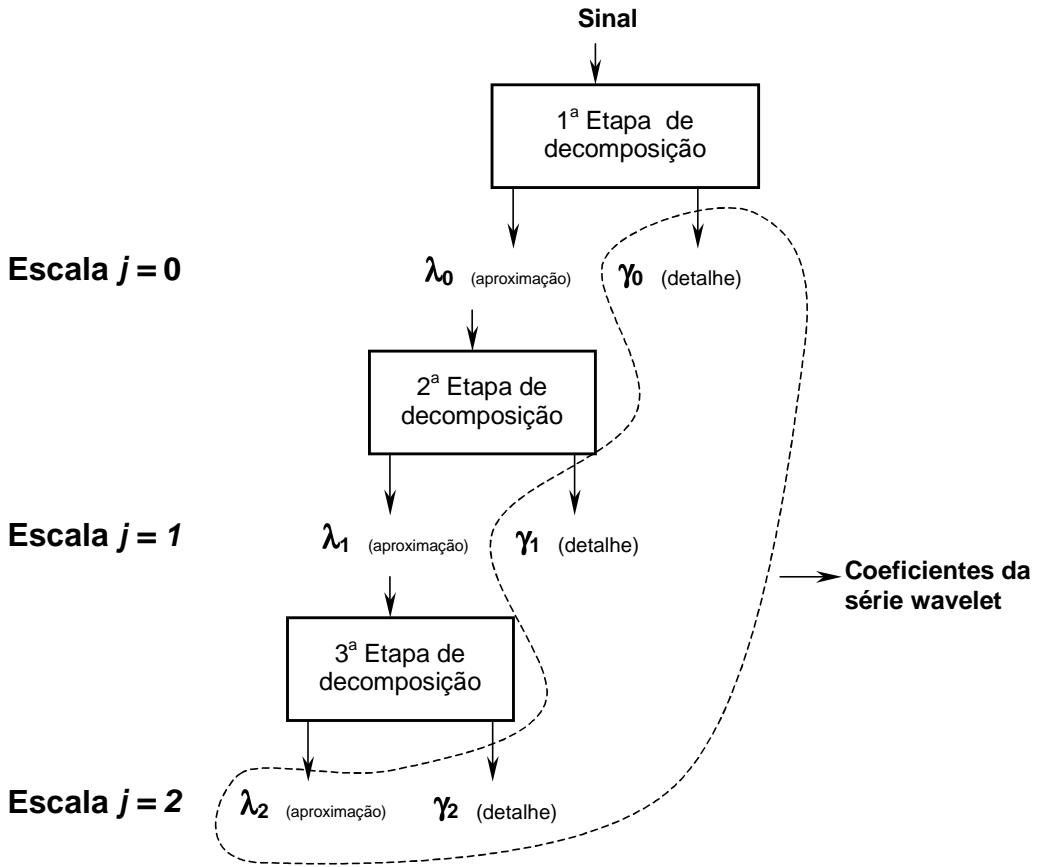
apenas em uma rotação dos vetores (sinais) para uma nova base, operação que mantém inalterada a dimensão destes desde que continuam a pertencer ao mesmo espaço.



**Figura 15 Implementação dos blocos básicos da Fast Wavelet Transform (FWT)**

A etapa de reconstrução atua sobre os coeficientes calculados na etapa anterior e inicialmente insere zeros (um após cada coeficiente) para restituir a eles o tamanho da série original e então aplicando-se os filtros de reconstrução para obter os dois sinais finais que, somados, darão origem ao sinal reconstruído.

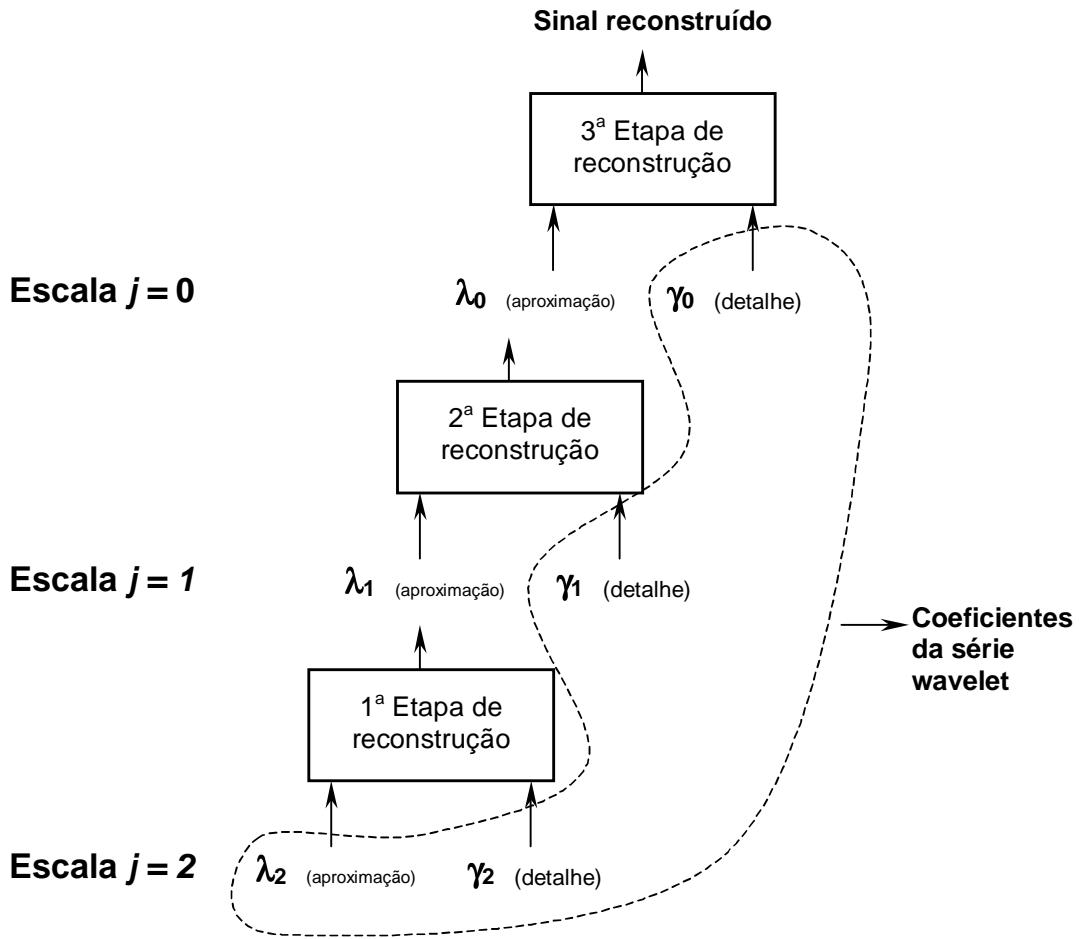
Estes blocos básicos devem ser adequadamente montados para se formar a transformada wavelet rápida. O esquema de montagem da transformada direta está mostrado na Figura 16 enquanto o esquema da transformada inversa está mostrado na Figura 17.



**Figura 16 Encadeamento dos blocos de decomposição para a obtenção dos coeficientes da série wavelet**

A Figura 16 mostra como os blocos de decomposição podem ser arranjados de forma a obter-se os coeficientes da série wavelet finita (17). Desde que o bloco básico de decomposição seja entendido é fácil entender o todo, pois é necessário somente aplicar o bloco de decomposição aos coeficientes de aproximação calculados na escala (etapa) anterior para obter os novos coeficientes de aproximação  $\lambda_j$  e de detalhe  $\gamma_j$ . Este processo pode seguir indefinidamente mas chegará um ponto onde os vetores de coeficientes terão um tamanho tão pequeno que a convolução com qualquer um dos filtros de decomposição será impossível. Deve-se relembrar que a cada etapa o tamanho dos vetores de coeficientes diminui pela metade por causa da decimação.

O processo de reconstrução é bastante parecido, só que no sentido inverso, ilustrado na Figura 17. Por fim, resta apenas associar os filtros de decomposição e reconstrução com a máscara da função de aproximação. Relembrando-se que a máscara - os coeficientes  $a_k$  da equação de escala (16) – contém todas as propriedades da wavelet. Os quatro filtros, portanto, são derivados desta máscara e sua obtenção envolve duas operações simples de ser implementadas computacionalmente.



**Figura 17 Encadeamento dos blocos de reconstrução para a obtenção do sinal original**

Formalmente, para máscaras com número de coeficientes par, os filtros podem ser calculados por:

$$LR = \{a_k\}$$

$$HR = REV( CHSODD( \{a_k\} ) )$$

$$LD = REV( \{a_k\} )$$

$$HD = CHSODD( \{a_k\} )$$

Neste esquema a operação indicada por REV é a transposição do vetor-argumento enquanto a operação indicada por CHSODD é a troca de sinal dos coeficientes ímpares. A transposição é a inversão da ordem dos elementos de um vetor finito, ou seja, se o vetor possui N elementos então o elemento  $k$  deve trocar de posição com o elemento  $N-k+1$ , sendo o primeiro elemento o de número 1 (convenção da linguagem Fortran).

Por exemplo, a máscara da wavelet Daubechies-2 é

$$LR = \{ 0.4830 \quad 0.8365 \quad 0.2241 \quad -0.1294 \}$$

e, portanto, os filtros FIR associados são

$$LR = \{ 0.4830 \quad 0.8365 \quad 0.2241 \quad -0.1294 \}$$

$$HR = \{ -0.1294 \quad -0.2241 \quad 0.8365 \quad -0.4830 \}$$

$$LD = \{ -0.1294 \quad 0.2241 \quad 0.8365 \quad 0.4830 \}$$

$$HD = \{ -0.4830 \quad 0.8365 \quad -0.2241 \quad -0.1294 \}$$

A obtenção do vetor-máscara é um trabalho particular de desenvolvimento. Cada pesquisador, ao exigir uma determinada propriedade da wavelet, deve saber expressar algebricamente suas necessidades para, assim, determinar os coeficientes da máscara, invariavelmente através da resolução de um sistema de equações. Se o leitor estiver interessado na construção de novas famílias de wavelet, a referência mais indicada é (DAUBECHIES 1992). No entanto existem repositórios de wavelets espalhados pela internet que podem fornecer estes coeficientes com um grande número de dígitos decimais, bem como referências ou mesmo os próprios trabalhos originais em formato postscript.

### **Um exemplo simples de aplicação da FWT**

Para não deixar o que foi explicado a nível abstrato, será desenvolvido um exemplo simples que faz o cálculo dos coeficientes de um vetor (sinal)  $x$  com somente 4 pontos, dado por:

$$x = \{ 111.4 \quad -107.8 \quad -51.6 \quad 61.5 \}$$

A wavelet escolhida para a transformação será a wavelet de Haar, também denominada Daubechies-1, cuja máscara é

$$MASK = \{ +1/\sqrt{2} \quad +1/\sqrt{2} \}$$

Os filtros FIR associados são, seguindo a metodologia apresentada anteriormente:

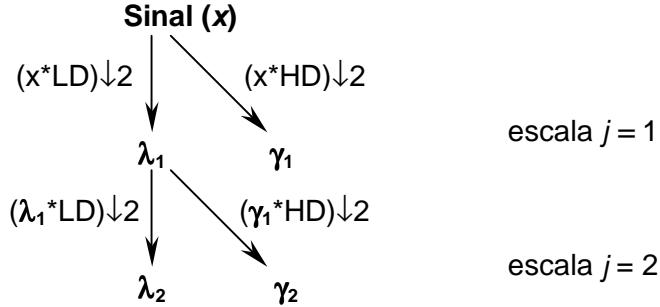
$$LR = MÁSCARA = \{ +1/\sqrt{2} \quad +1/\sqrt{2} \} = \{ 0.707 \quad 0.707 \}$$

$$HR = REV(CHSODD(LR)) = REV(\{ -1/\sqrt{2} \quad +1/\sqrt{2} \}) = \{ 0.707 \quad -0.707 \}$$

$$LD = REV(LR) = REV(\{ +1/\sqrt{2} \quad +1/\sqrt{2} \}) = \{ 0.707 \quad 0.707 \}$$

$$HD = REV(HR) = REV(\{ -0.707 \quad 0.707 \}) = \{ -0.707 \quad 0.707 \}$$

Como o sinal tem somente 4 pontos, somente 2 níveis de coeficientes podem ser calculados por causa da decimação. O esquema desta transformação particular está ilustrada na Figura 18.



**Figura 18 Esquema ilustrativo da transformada wavelet direta do exemplo**

Inicialmente calcula-se os coeficientes de detalhe ( $\gamma_1$ ) do primeiro nível (1) através da convolução do sinal  $x$  com o filtro passa-alta associado ( $HD$ ) e posterior *decimação*. Esta última operação, indicada pela notação  $\downarrow 2$ , consiste em manter-se somente os coeficientes ímpares de um vetor, retirando-se os elementos pares. Como o vetor  $x$  tem 4 elementos (e também a convolução), a decimação fará com que os vetores  $\gamma_1$  e  $\lambda_1$  tenham somente 2 elementos.

A seguir calcula-se os coeficientes de aproximação  $\lambda_1$  através da convolução do sinal  $x$  com o filtro passa-baixa ( $LD$ ) associado e posterior decimação. Os coeficientes do segundo nível são obtidos de forma análoga utilizando-se agora o vetor de aproximação do nível anterior  $\lambda_1$  como base.

Desta forma os coeficientes do nível 1 são:

$$\begin{aligned}
 \gamma_1 &= \text{conv}(x, HD) \downarrow 2 \\
 &= \{155.0 \quad -39.7 \quad -80.0 \quad 43.5\} \downarrow 2 \\
 &= \{155.0 \quad -80.0\}
 \end{aligned}$$

$$\begin{aligned}
 \lambda_1 &= \text{conv}(x, LD) \downarrow 2 \\
 &= \{2.5 \quad -112.7 \quad 7.0 \quad 43.5\} \downarrow 2 \\
 &= \{2.5 \quad 7.0\}
 \end{aligned}$$

Os coeficientes do nível 2 são:

$$\begin{aligned}
 \gamma_2 &= \text{conv}(\lambda_1, HD) \downarrow 2 \\
 &= \{-3.2 \quad 4.9\} \downarrow 2 \\
 &= \{-3.2\}
 \end{aligned}$$

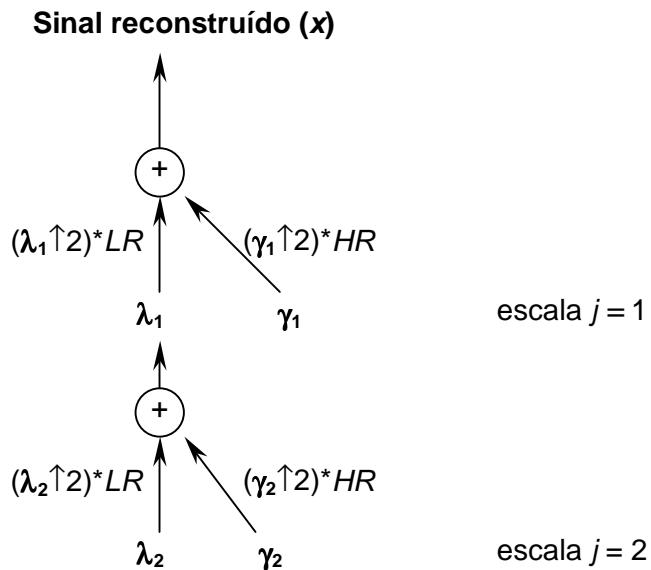
$$\begin{aligned}
 \lambda_2 &= \text{conv}(\lambda_1, \text{LD}) \downarrow 2 \\
 &= \{6.7 \quad 4.9\} \downarrow 2 \\
 &= \{6.7\}
 \end{aligned}$$

Relembra-se que a convolução é uma operação executada pela inversão da ordem dos elementos do segundo termo e multiplicando-se ponto-a-ponto pelo primeiro. Por exemplo, o primeiro e segundo termos da convolução entre o vetor  $x$  e o filtro HD serão:

$$\begin{aligned}
 1^{\circ} \text{ termo } \text{conv}(x, \text{HD}) &= (+111.4)(+0.707) + (-107.8)(-0.707) = 155.0 \\
 2^{\circ} \text{ termo } \text{conv}(x, \text{HD}) &= (-107.8)(+0.707) + (-51.6)(-0.707) = -39.7
 \end{aligned}$$

A forma usual de agrupar-se os coeficientes da transformada wavelet num único vetor é agrupar-se primeiro os coeficientes de detalhe, do menor para o maior nível, e finalmente o coeficiente de aproximação do último nível. Portanto os coeficientes wavelet resultantes da transformação do vetor original  $x$  utilizando-se a wavelet de Haar são:

$$C = \{\gamma_1 \quad \gamma_2 \quad \lambda_2\} = \{155.0 \quad -80.0 \quad -3.2 \quad 6.7\}$$



**Figura 19** Esquema ilustrativo da transformada wavelet inversa do exemplo

A reconstrução é um processo similar, só que na direção oposta conforme ilustrado Figura 19.

Deve-se, inicialmente, montar o vetor de coeficientes de aproximação do primeiro nível:

$$\lambda_1 = \text{conv}(\lambda_2 \uparrow 2, \text{LR}) + \text{conv}(\gamma_2 \uparrow 2, \text{HR})$$

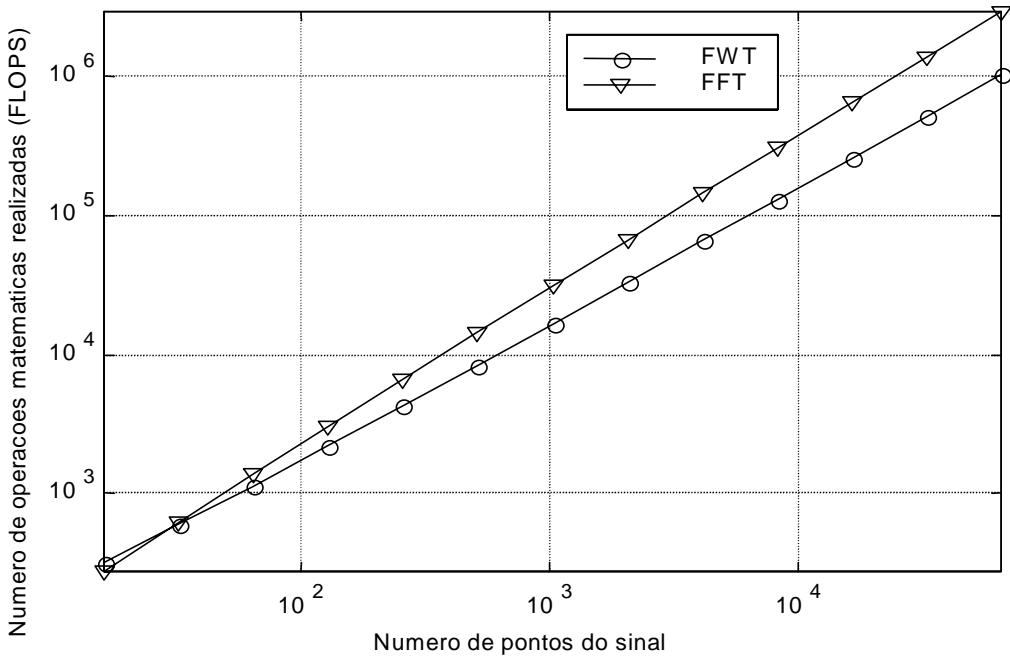
$$\begin{aligned}
&= \text{conv}(\{6.7 \ 0\}, \{0.707 \ 0.707\}) + \\
&\quad \text{conv}(\{-3.2 \ 0\}, \{0.707 \ -0.707\}) \\
&= \{4.737 \ 4.737\} + \{-2.262 \ +2.262\}\} \\
&= \{2.5 \ 7.0\}
\end{aligned}$$

para então calcular-se o sinal original

$$\begin{aligned}
\mathbf{x} &= \text{conv}(\boldsymbol{\lambda}_1 \uparrow 2, \text{LR}) + \text{conv}(\boldsymbol{\gamma}_1 \uparrow 2, \text{HR}) \\
&= \text{conv}(\{2.5 \ 0.0 \ 7.0 \ 0.0\}, \{0.707 \ 0.707\}) + \\
&\quad \text{conv}(\{155.0 \ 0.0 \ -80.0 \ 0.0\}, \{0.707 \ -0.707\}) \\
&= \{1.8 \ 1.8 \ 4.9 \ 4.9\} + \\
&\quad \{109.6 \ -109.6 \ -56.5 \ 56.5\} \\
&= \{111.4 \ -107.8 \ -51.6 \ 61.4\}
\end{aligned}$$

e o processo está terminado. Existe uma diferença mínima entre o sinal reconstruído e o original que é notada no último elemento que deveria ser 61.5 ao invés de 61.4 mas isto é devido ao fato de que foi utilizada somente uma casa decimal de precisão para representar os coeficientes a fim de não sobrecarregar o texto. É natural, portanto, que os erros de arredondamento se acumulem a cada etapa. No entanto nas análises reais a precisão utilizada é bem superior permitindo atingir a reconstrução das séries com uma maior precisão.

Um aspecto importante da FWT é que ela não envolve o cálculo de nenhuma função transcendental (senos, cossenos, exponenciais), apenas operações algébricas. Isto fornece à FWT uma rapidez muito grande além da operação poder ser feita *in loco*, ou seja, a FWT é adequada a situações críticas de escassez de memória ou tempo de CPU. Além disto a FWT é computada em aproximadamente  $16 N$  operações algébricas (FLOPS ou *FLOATing Point operationS*) para uma máscara com 8 coeficientes enquanto a FFT é computada em aproximadamente  $5 N \log N$  flops. Esta diferença fica mais clara na Figura 20 na qual é mostrado o número de operações matemáticas gastas por cada uma das técnicas.



**Figura 20 Número de operações matemáticas realizadas pela FFT e a FWT**

Deve-se observar que, apesar de os valores das séries apresentarem-se bem próximos, o gráfico tem ambos os eixos em escala logarítmica. Deve-se levar em conta também que uma operação aritmética simples ou uma exponencial complexa são computadas como apenas uma operação matemática (flop) mas a primeira utiliza muito mais clocks de CPU que a segunda. Portanto, a diferença em tempo de processamento é maior que a indicada no gráfico da Figura 20.

### II.3 As transformadas bilineares

Todas as transformadas vistas até o momento são classificadas como lineares visto que as transformadas obedecem os princípios de lineariedade usuais. No entanto estas transformadas pecam por não obter suficiente precisão, ou seja, não se aproximam-se suficientemente do limite mínimo imposto pelo princípio da incerteza. As transformadas bilineares conseguem aproximar-se deste limite ao apresentar uma forma bilinear, ou seja, a função analisada aparece multiplicando-se duas vezes na expressão da transformada. Apesar de chamada bilinear, estas transformadas não são, evidentemente, lineares.

#### A transformada de Wigner-Ville

Na procura por um “espectro instantâneo”, VILLE (1946, 1948) queria visualizar a energia de um sinal  $f(t)$  no plano tempo-freqüência e obter uma densidade de energia

que satisfizesse as condições marginais. Para tal conseguiu ver a analogia existente entre mecânica quântica e processamento de sinais, de forma que redefiniu a distribuição de WIGNER (1932) para o domínio tempo-freqüência como

$$WD_f(t, \omega) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} f^*(t + \frac{1}{2}\tau) f(t - \frac{1}{2}\tau) e^{-i\omega\tau} d\tau \quad (19)$$

O entendimento do significado da distribuição de Wigner-Ville poderá ser ilustrada com um exemplo. Observa-se que o sinal  $f(t) = e^{i\Omega t}$  é um harmônico com amplitude constante cuja frequência angular é  $\Omega$ . A transformada de Wigner para este harmônico é

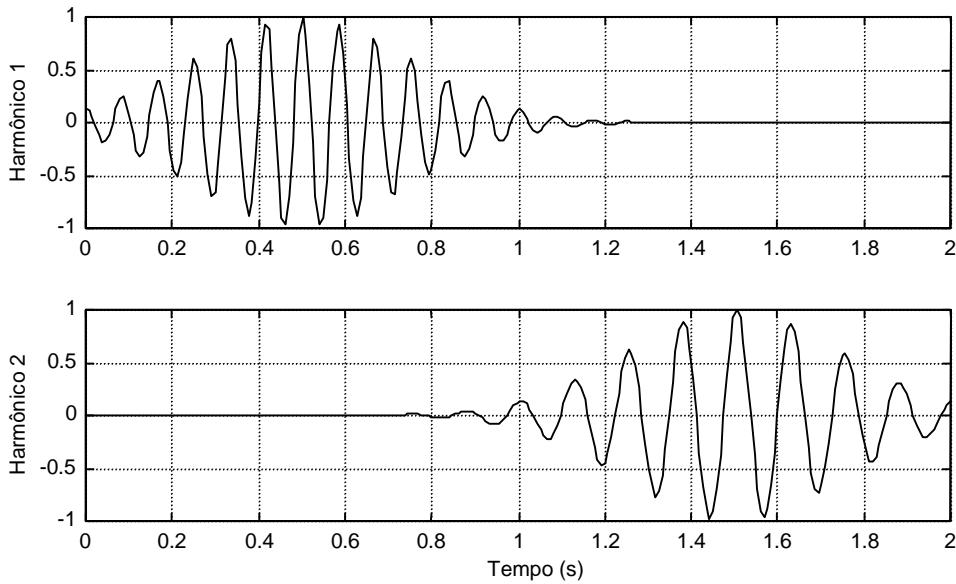
$$WD_f(t, \omega) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{i\varphi(t)\tau} e^{-i\omega\tau} d\tau = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{-i(\omega-\Omega)\tau} d\tau = \frac{1}{2\pi} \delta(\omega - \Omega) \quad (20)$$

ou seja, uma linha de funções delta ao longo da freqüência do harmônico. Apesar de este resultado só se aplicar a este caso em particular, felizmente a transformação tende sempre a concentrar bastante energia em torno das freqüências instantâneas para uma grande variedade de sinais.

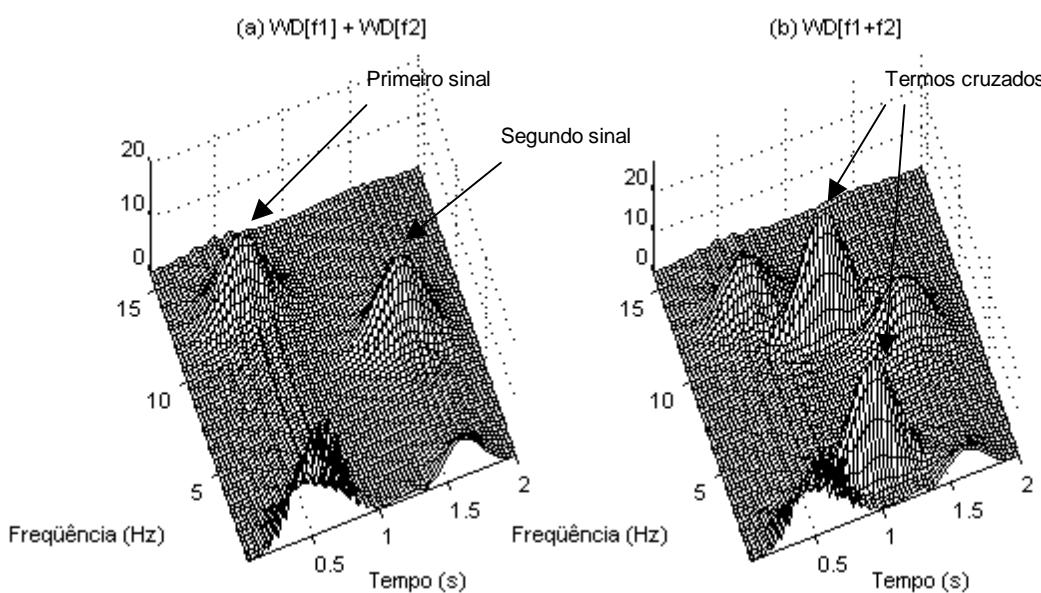
A grande deficiência da distribuição de Wigner-Ville é a ocorrência de termos cruzados, também chamados de interferência. Para exemplificar graficamente o que são os termos cruzados na transformada de Wigner-Ville, usou-se dois sinais, obtidos através do produto de um harmônico por uma janela gaussiana. Os sinais assim obtidos são mostrados na Figura 21. O primeiro sinal está centrado no instante 0,5s e seu harmônico tem freqüência 12Hz. O segundo sinal está centrado em 1,5s e seu harmônico tem freqüência 8Hz.

Na primeira etapa foram obtidas as transformadas de cada sinal separadamente, sendo os resultados obtidos então somados. Numa segunda etapa somaram-se os dois sinais no tempo e a partir deste novo sinal foi calculada a transformada de Wigner-Ville. As duas transformadas obtidas desta forma são mostradas na Figura 22.

Como esperado aparecem na Figura 22a duas elevações separadas correspondentes aos dois sinais analisados. Próximo ao eixo do tempo aparecem elevações que também podem ser consideradas como interferência, sendo no entanto de mais difícil compreensão pois devem-se à interferência entre a parte positiva do espectro do sinal com sua parte negativa (simétrica).



**Figura 21** Harmônicos usados para demonstrar os termos cruzados na distribuição de Wigner-Ville



**Figura 22** Distribuições de Wigner-Ville calculadas (a) separadamente para cada sinal e posteriormente somadas e (b) para os dois sinais somados no tempo.

Os termos cruzados são os quatro picos adicionais indicados na Figura 22b. A localização dos picos de interferência será sempre dada pelas coordenadas

$$\left( \frac{t_1 + t_2}{2}, \frac{\omega_1 + \omega_2}{2} \right) \text{ e } \left( \frac{t_1 + t_2}{2}, \frac{\omega_1 - \omega_2}{2} \right) \quad (21)$$

Neste exemplo da Figura 22, as coordenadas serão

$$\left( \frac{0.5 + 1.5}{2}, \frac{12 + 8}{2} \right) = (1.0s, 10Hz) \text{ (localização do primeiro pico de interferência)}$$

$$\left( \frac{0.5+1.5}{2}, \frac{12-8}{2} \right) = (1.0s, 2Hz) \text{ (localização do segundo pico de interferência)}$$

Os termos cruzados que aparecem na Figura 22b podem ser previstos analiticamente tomando-se a transformada do sinal total

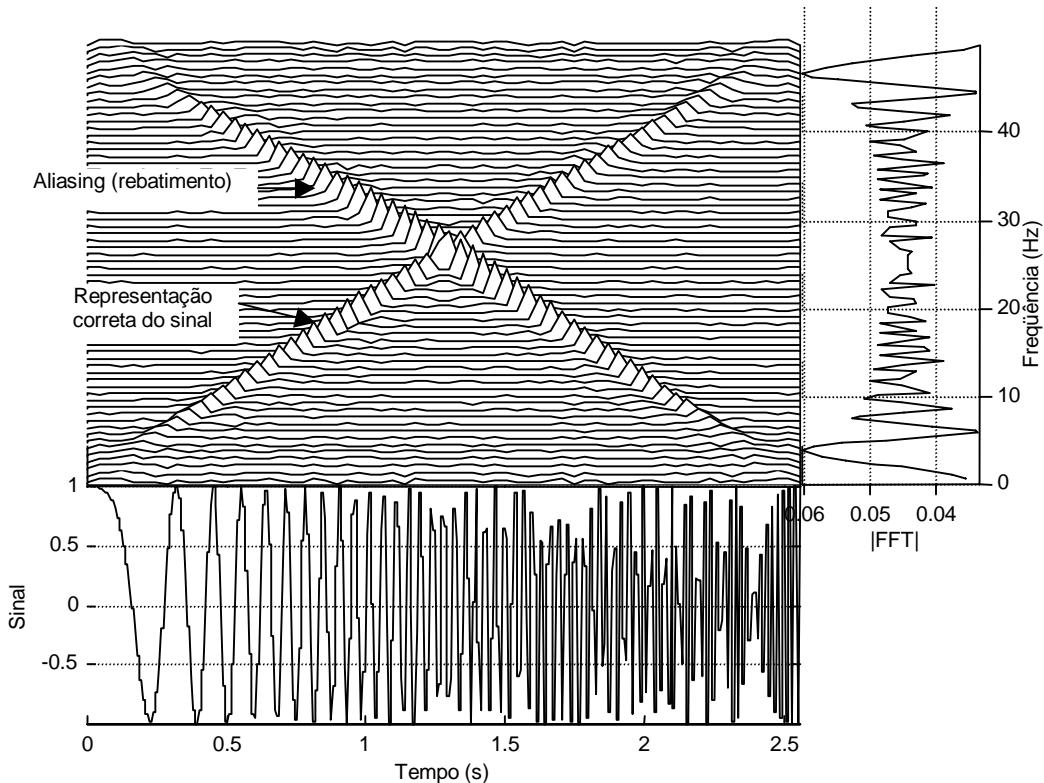
$$\begin{aligned} WD[f_1(t) + f_2(t)](t, \omega) &= \\ &= \frac{1}{2\pi} \int_{-\infty}^{+\infty} \left[ f_1\left(t + \frac{\tau}{2}\right) + f_2\left(t + \frac{\tau}{2}\right) \right] \left[ f_1\left(t - \frac{\tau}{2}\right) + f_2\left(t - \frac{\tau}{2}\right) \right] e^{-i\omega t} d\tau \\ &= WD_1(t, \omega) + WD_2(t, \omega) + \frac{1}{2\pi} \int_{-\infty}^{+\infty} f_1\left(t + \frac{\tau}{2}\right) f_2\left(t - \frac{\tau}{2}\right) e^{-i\omega t} d\tau \quad (22) \\ &\quad + \frac{1}{2\pi} \int_{-\infty}^{+\infty} f_2\left(t + \frac{\tau}{2}\right) f_1\left(t - \frac{\tau}{2}\right) e^{-i\omega t} d\tau \\ &= WD_1(t, \omega) + WD_2(t, \omega) + WD_{12}(t, \omega) + WD_{21}(t, \omega) \end{aligned}$$

Vê-se que na transformada do sinal total aparecem os termos cruzados  $WD_{12}$  e  $WD_{21}$  responsáveis pela interferência indesejada. Estes termos não tendem a zero quando separam-se mais os harmônicos no tempo ou em freqüência. O pior ainda é que estes termos cruzados gerarão valores negativos para a distribuição, o que é uma incorreção física desde que a transformada de Wigner-Ville pretende fornecer uma distribuição de energia.

A versão discreta da distribuição de Wigner-Ville (assim como todas as da classe geral apresentadas na próxima seção) apresenta *aliasing* se o sinal contiver freqüências superiores a um quarto da freqüência de digitalização, ou metade da taxa de Nyquist. Isto se deve ao termo  $t \pm \tau/2$  que, discretizado da forma  $\tau = m\Delta t$  e  $t = n\Delta t$ , permitirá que  $m$  assuma apenas valores pares pois deve-se garantir que  $f(t \pm \tau/2) = f[n \pm m/2]$ . Desta forma apenas metade da informação estará sendo efetivamente utilizada para calcular a distribuição.

O sinal mostrado na Figura 23 é um chirp linear que possui freqüência instantânea zero em  $t=0s$  e freqüência 50Hz (taxa de Nyquist) em  $t=2.56s$ , intante final do sinal. A freqüência de digitalização é de 100Hz e, portanto, a taxa de Nyquist não é violada. Apresenta-se também no mesmo gráfico o módulo da FFT e a distribuição de Wigner-Ville. Esta forma de apresentação conjunta dos três gráficos (sinal no tempo, módulo do espectro e distribuição tempo-freqüência) facilita bastante a visualização e será bastante empregada neste trabalho deste ponto em diante. Pode-se observar na Figura

23 que existe um rebatimento da onda no sentido das freqüências, caracterizando completamente a existência de *aliasing*. Conclui-se que a distribuição de Wigner-Ville deverá ser calculada a partir de sinais para os quais o espectro é nulo para freqüências maiores que um quarto da taxa de digitalização.



**Figura 23 Aliasing na distribuição de Wigner-Ville**

A ocorrência de termos cruzados (ou interferência) quase que inviabiliza totalmente a aplicação da transformada de Wigner em aplicações sérias de processamento de sinais. Por isso muitas pesquisas foram feitas no sentido de eliminar estes termos, culminando com a técnica do núcleo proposta por Leon Cohen e descrita no próximo item.

### Generalização - a técnica do núcleo

COHEN (1966, 1976, 1996) percebeu que todas as distribuições tempo-freqüência podem ser generalizadas de forma a pertencerem a uma única família, chamada por homenagem ao seu criador de *família de Cohen*. Também é comumente conhecida como *Generalized Time-Frequency Distribution* (GTFD). A generalização proposta por Cohen afirma que *qualquer* distribuição tempo-freqüência  $C_f(t, \omega)$  pode ser escrita

como a transformada de Fourier<sup>1</sup> de um tipo de função de autocorrelação  $R_f(t, \tau)$  da função  $f(t)$ ,

$$C_f(t, \omega) = \frac{1}{4\pi^2} \int_{-\infty}^{+\infty} R_f(t, \tau) e^{-i\omega\tau} d\tau \quad (23)$$

onde

$$R_f(t, \tau) = \int_{-\infty}^{+\infty} f(u + \frac{1}{2}\tau) f^*(u - \frac{1}{2}\tau) \cdot \Phi(t - u, \tau) du \quad (24)$$

A função  $\Phi(t, \tau)$  é a janela utilizada, cuja principal função é eliminar os termos cruzados da distribuição. A janela  $\Phi(t, \tau)$  é a transformada de Fourier<sup>1</sup> da função  $\phi(\theta, \tau)$ , denominada *núcleo da distribuição*, ou seja,

$$\Phi(t, \tau) = \int_{-\infty}^{+\infty} \phi(\theta, \tau) e^{-it\theta} d\theta \quad (25)$$

Existem vários núcleos desenvolvidos, cada qual com suas propriedades particulares o que os tornam vantajosos ou não para determinado tipo de aplicação. A Tabela 1 apresenta uma relação de alguns deles e os respectivos trabalhos de onde se originaram.

Foi visto anteriormente que a distribuição de Wigner possui uma grande tendência a apresentar termos cruzados, também chamados de interferência. Esta não é uma propriedade somente da transformada de Wigner. *Todas* as distribuições tempo-freqüência, algumas mais, outras menos, apresentam esta característica por serem *bilineares*, termo cunhado pelo próprio Wigner para indicar que o sinal aparece duas vezes na expressão da distribuição.

Para que a interferência seja reduzida deve-se requerer do núcleo que

$$\phi(\theta, \tau) \rightarrow 0 \text{ para } |\theta\tau| \rightarrow \infty \quad (26)$$

A distribuição de Wigner possui núcleo constante,  $\phi(\theta, \tau) = 1$ , e por isso é tão sensível à interferência. Esta é a idéia principal das técnicas anti-interferência sugeridas por uma série de autores, tais como CHOI, WILLIAMS, 1989, ZHAO, ATLAS, MARKS, 1990, ATLAS, LOUGHLIN, PITTON, 1992, CUNNINGHAM, WILLIAMS, 1993.

---

<sup>1</sup> A menos de uma constante de proporcionalidade.

Existem quatro razões muito fortes para representar uma distribuição tempo-freqüência através de seu núcleo  $\phi(\theta, \tau)$  ao invés da função-janela  $\Phi(t, \tau)$ , as quais são listadas a seguir:

1. Se existe uma propriedade especial que se deseja que a distribuição atenda, quase sempre será possível exigir-la diretamente de seu núcleo e, via de regra, a equação é mais simples.
2. Dada uma distribuição, todas as suas propriedades podem ser avaliadas por simples inspeção de seu núcleo.
3. Dado um núcleo, a distribuição é fácil de ser calculada.
4. Pode-se criar uma rotina computacional genérica para o cálculo das distribuições, implementando-se um ou outro núcleo com poucas alterações no código.

**Tabela 1 Exemplos de núcleos de distribuições conhecidas**

Distribuição	Núcleo: $\phi(\theta, \tau)$
WIGNER (1932)	1
MARGENOU-HILL (1961)	$\cos\left(\frac{1}{2}\theta\tau\right)$
RIHACZEK (1968)	$e^{i\frac{\theta\tau}{2}}$
COHEN (1966)	$\frac{\sin\frac{1}{2}\theta\tau}{\frac{1}{2}\theta\tau}$
PAGE (1952)	$e^{i\theta \tau }$
CHOI-WILLIAMS (1989)	$e^{-\frac{(\theta\tau)^2}{2}}$
Espectrograma <sup>2</sup>	$\int_{-\infty}^{+\infty} h^*(u - \frac{1}{2}\tau)h(u + \frac{1}{2}\tau)e^{-i\theta u} du$
ZHAO-ATLAS-MARKS (1990)	$g(\tau)\tau \frac{\sin a\theta\tau}{a\theta\tau}$

A seguir serão listadas as possíveis restrições a serem aplicadas ao núcleo com as respectivas propriedades que resultam na distribuição. Deve-se ter em mente que uma distribuição nunca poderá atender a todas as propriedades listadas pois algumas são incompatíveis entre si.

## Preservação da energia

Para que seja preservada a energia do sinal, isto é,

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} C_f(t, \omega) d\omega dt = \int_{-\infty}^{+\infty} |f(t)|^2 dt = \int_{-\infty}^{+\infty} |F(\omega)|^2 d\omega \quad (27)$$

é suficiente exigir-se que o núcleo atenda a condição

$$\phi(0,0)=1 \quad (28)$$

## Satisfação da condição marginal de tempo

Para que a condição marginal no tempo

$$\int_{-\infty}^{+\infty} C_f(t, \omega) d\omega = |f(t)|^2 \quad (29)$$

seja atendida é suficiente que

$$\phi(\theta,0)=1, \forall \theta \in \Re \quad (30)$$

## Satisfação da condição marginal de freqüência

Da mesma forma, para que a condição marginal em freqüência

$$\int_{-\infty}^{+\infty} C_f(t, \omega) dt = |F(\omega)|^2 \quad (31)$$

seja atendida é suficiente que

$$\phi(0,\tau)=1, \forall \tau \in \Re. \quad (32)$$

## Distribuição real

Para que a distribuição seja real, isto é,  $C_f(t, \omega) \in \Re$ , é suficiente exigir-se do núcleo que

$$\phi(\theta, \tau) = \phi^*(-\theta, -\tau). \quad (33)$$

## Aliasing

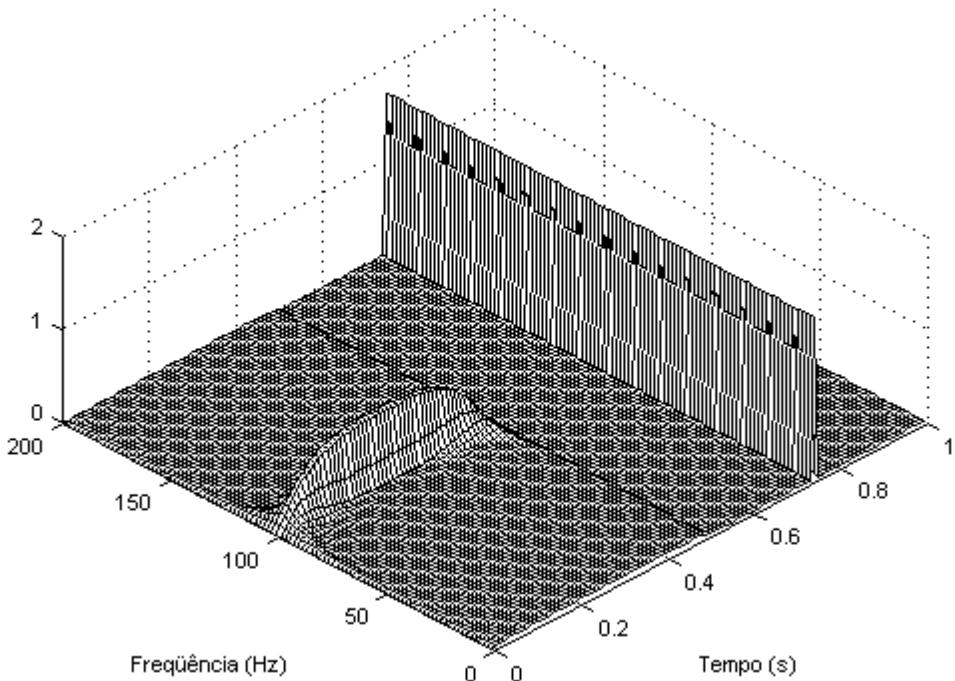
O mesmo que foi dito para a transformada de Wigner se aplica a toda família de Cohen com respeito ao *aliasing*. Toda distribuição tempo-freqüência, ao ser discretizada *da forma padrão*, estará sujeita a *aliasing* se existirem no sinal freqüências maiores que um quarto da freqüência de digitalização. Entenda-se pelo grifo que o problema do *aliasing* está relacionado com a forma inicialmente escolhida por muitos autores (*forma padrão*)

<sup>2</sup>  $\mathbf{h}(t)$  é a gaussiana normalizada.

para discretizar a expressão da classe geral. Esta discretização torna imediatamente inutilizável metade dos pontos do sinal durante o cálculo.

Uma forma não tão elegante mas que erradica completamente o *aliasing* da distribuição foi desenvolvida por JEONG e WILLIAMS (1991), que denominaram a técnica por *Aliasing - Free Generalized Discrete Time Frequency Distribution* (AF-GDTFD). No mesmo trabalho que definem a AF-GDTFD os autores sugerem um núcleo baseado numa janela Hamming que provou ser realmente eficiente. No entanto é preciso diferenciar a técnica anti-*aliasing* sugerida pelos autores da distribuição baseada na janela Hamming. Por isso se denominará esta distribuição particular, baseada na janela Hamming, agora por diante de *alias-free*, como simples referência ao trabalho que a originou.

Apresenta-se na Figura 24 a distribuição alias-free calculada para o sinal do exemplo dado da Figura 8 à Figura 10.



**Figura 24 Distribuição alias-free de Jeong e Williams**

Vê-se na figura Figura 24 que a distribuição alias-free tem excelente comportamento em comparação com o espectrograma - vide Figura 10. Além de boa resolução no tempo e em freqüência, erradicação completa do problema de *aliasing*, possui baixa sensibilidade à interferência, principal problema da distribuição de Wigner-Ville.

Logicamente o limite do princípio da indeterminação também se aplica a esta distribuição, mas ela está bem mais próxima do mínimo que a maioria das distribuições.

Este método é descrito em detalhes em BUCHER (1998), onde é proposto um algoritmo eficiente e genérico para o cálculo de qualquer distribuição tempo-frequência da classe de Cohen.

# **III CARACTERIZAÇÃO DA PERDA DE ENERGIA EM SISTEMAS AMORTECIDOS**

## ***III.1 Introdução***

Neste capítulo é apresentada uma técnica para estimar a taxa de amortecimento de estruturas – ou qualquer sistema harmônico em geral – utilizando-se as distribuições tempo-frequência. Esta técnica pode ser utilizada em estruturas que apresentem variações do valor da taxa de amortecimento ao longo do tempo.

A caracterização de um sistema harmônico amortecido pode ser extremamente fácil quando apenas uma frequência (modo) está presente no sinal. Neste caso mais simples, pode-se observar claramente o amortecimento através do decréscimo da amplitude dos picos e, através de um método simples como o decaimento logarítmico, pode-se então caracterizar a perda de energia, inclusive verificando se ela é compatível com a premissa assumida de uma taxa de amortecimento constante ao longo do tempo.

A análise torna-se mais complexa quando existem várias frequências, ou modos, presentes no sinal. Pelo fato de que as diferentes frequências se confundem na representação no tempo, não se pode mais caracterizar a perda de energia de cada modo através do decréscimo de amplitude dos picos. Desta forma torna-se necessário a utilização de técnicas mais sofisticadas tanto no domínio do tempo como no domínio da frequência. Deve-se no entanto destacar que grande parte das técnicas descritas na literatura técnica baseiam-se na premissa de que a estrutura possua um comportamento linear e que a taxa de amortecimento de cada modo seja invariante com o tempo.

Utilizando-se as distribuições tempo-frequência (TFDs) pode-se então transformar casos complexos, com vários modos muito próximos e amortecimento não-linear por exemplo, em vários casos simples, que serão finalmente analisados com uma técnica similar ao decaimento logarítmico.

Deve-se deixar claro, desde o princípio, que o método apresentado propõe-se apenas e tão somente calcular curvas de energia. Estas curvas podem ser usadas posteriormente, via modelos simplificados ou não, para caracterizar amortecimentos de qualquer natureza, sejam eles histeréticos, viscosos ou de Coulomb. Ao final será visto que as

taxas de amortecimento, sejam elas de qualquer natureza, podem ser obtidas com uma simples operação aritmética a partir das curvas de energia.

### **III.2 Desenvolvimento teórico**

Inicialmente é preciso introduzir o conceito de amortecimento, o que não é uma tarefa trivial visto que existem várias grandezas físicas que podem caracterizá-lo. Uma delas, o decaimento logarítmico, de acordo com RAO (1995), pode ser definido como metade da perda (relativa) de energia por ciclo, ou seja,

$$\delta = -\frac{1}{2} \frac{\Delta E}{E} = -\frac{1}{2} \frac{E(t+T) - E(t)}{E(t)} \quad (34)$$

onde  $E$  é a energia total (cinética + potencial) do modo,  $\Delta E$  é a variação de energia em um ciclo e  $T$  é o período de vibração do modo. A energia  $E$  é dependente do tempo, ou seja,  $E=E(t)$ . Pode-se verificar que  $\Delta E$  é apenas o resultado de uma expansão em série de Taylor da energia  $E(t+T)$  em torno do instante  $t$ , da qual foi retida apenas a parcela constante  $E(t)$  e a segunda parcela (primeira derivada). Desta forma para expressar o decaimento logarítmico com maior generalidade pode-se escrever

$$\delta = -\frac{1}{2} \frac{\left( \frac{\partial E}{\partial t} \times T \right)}{E} \quad (35)$$

A forma contínua (35) envolvendo uma derivada no tempo é, além de mais geral, mais útil à presente análise que a forma discreta (34). Como é mais comum expressar a frequência de vibração do sistema através de sua frequência angular  $\Omega$  (em rad/s) do que por seu período  $T$ , modifica-se (35) para obter

$$\delta = -\frac{\pi}{\Omega} \frac{\partial E / \partial t}{E} \quad (36)$$

Usando a propriedade da derivada do logaritmo natural, escreve-se finalmente

$$\delta = -\frac{\pi}{\Omega} \frac{\partial}{\partial t} \log E \quad (37)$$

que expressa bem apropriadamente o termo “decaimento logarítmico”. Outras grandezas equivalentes que definem o amortecimento como a constante de amortecimento histerético  $\beta$  e a taxa de amortecimento viscoso  $\xi$  podem ser relacionadas ao decaimento logarítmico através da idéia de perda de energia por ciclo (RAO, 1995), de tal forma que

$$\delta = \pi\beta = 2\pi\xi \quad (38)$$

Deve-se esclarecer que a igualdade (38) é válida tendo em vista apenas um modo em particular e que os amortecimentos sejam pequenos, ou seja, em termos práticos,  $\xi$  menor que 30% (RAO, 1995). Apesar de, inicialmente, ter-se dado maior atenção ao decaimento logarítmico  $\delta$ , isto foi feito com o objetivo didático de relacionar-se as diversas taxas de amortecimento à taxa de perda de energia. Contudo a prática favorece o uso da taxa de amortecimento viscoso  $\xi$  devido ao fato dos modelos numéricos resultantes de seu uso serem bem mais simples. Para isto deve-se substituir (38) em (37) obtendo

$$\xi = -\frac{1}{2\Omega} \frac{\partial}{\partial t} \log E \quad (39)$$

Pode-se reescrever esta última equação de uma forma análoga, porém mais útil à análise numérica. Esta forma define a taxa de amortecimento viscoso como a taxa de variação (com o tempo) de uma nova variável chamada *amplitude normalizada*:

$$\xi = -\frac{\partial}{\partial t} \hat{v} \quad (40)$$

onde a amplitude normalizada é definida através da seguinte expressão:

$$\hat{v} = \frac{\log(E)}{2\Omega} \quad (41)$$

Utilizando-se o conceito de perda de energia instantânea (36) pode-se obter quaisquer taxas de amortecimento como função do decaimento da amplitude normalizada como, por exemplo, a taxa de amortecimento histerético

$$\beta = -2 \frac{\partial}{\partial t} \hat{v} \quad (42)$$

Nota-se que as duas últimas equações, (40) e (41), são exatamente iguais às primeiras, (37) e (38), somente a forma matemática foi um pouco alterada, e portanto a definição continua exatamente a mesma.

Estas equações permitem obter o amortecimento dado que uma curva de decréscimo da energia  $E = E_\Omega(t)$  seja obtida para cada frequência  $\Omega$  contida no sinal, o que não é possível nem com o sinal no tempo nem no domínio da frequência. Este é justamente o problema que as distribuições tempo-frequência propõem-se a resolver.

Para que esta tarefa seja realizada, é preciso definir-se uma nova função que permita obter a energia modal  $E = E_\Omega(t)$  a partir de uma distribuição tempo-frequência

$C_f(t, \omega)$  qualquer. A esta função de mapeamento dar-se-á a denominação de *Função de Transferência de Energia* ou simplesmente FTE, definida de tal forma que

$$C_f(t, \omega) = \alpha(\omega, \Omega) \cdot E_\Omega(t) \quad (43)$$

onde  $C_f(t, \omega)$  é a distribuição tempo-frequência (TFD) da função  $f(t)$  e  $\alpha(\omega, \Omega)$  é a função de transferência de energia (FTE), independente do tempo. Esta equação diz que a energia relacionada a uma determinada frequência (ou modo) é proporcional ao valor da TFD nas vizinhanças daquela frequência. Em outras palavras, a energia  $E = E_\Omega(t)$  relacionada a uma frequência  $\Omega$  pode ser estimada através do valor de  $C_f(t, \omega)$  para valores de  $\omega$  nas vizinhanças de  $\Omega$ .

Para que a equação (43) seja válida é necessário (e suficiente) que a distribuição tempo-frequência  $C_f(t, \omega)$ , respeite as condições de conservação de energia, a marginal no tempo e a marginal em frequência.

Em termos práticos, o valor de  $\alpha$  realmente não é importante *para o cálculo da taxa de amortecimento*. Se a FTE for independente do tempo, a sequência de operações log+derivada irá anular qualquer variável independente do tempo, eliminando assim a FTE das expressões resultantes, ou seja,

$$\begin{aligned} \xi &= -\frac{1}{2\Omega} \frac{\partial}{\partial t} \log E \\ &= -\frac{1}{2\Omega} \frac{\partial}{\partial t} \log(C/\alpha) \\ &= -\frac{1}{2\Omega} \frac{\partial}{\partial t} [\log(C) - \log(\alpha)] \\ &= -\frac{1}{2\Omega} \frac{\partial}{\partial t} \log(C) \end{aligned} \quad (44)$$

Se a FTE for dependente do tempo então o processo será mais trabalhoso pois sua expressão aparecerá explicitamente nas equações resultantes. Por sorte a grande maioria das transformadas tempo-frequência fornecem FTEs independentes do tempo, inclusive o espectrograma. Para o espectrograma, em particular, é relativamente fácil obter-se uma forma analítica fechada para a FTE partindo-se do modelo de um sistema harmônico amortecido simples, regido pela equação

$$v(t) = \rho e^{i(\Omega t + \theta)} e^{-\xi \Omega t} \quad (45)$$

onde  $\rho$  e  $\theta$  são constantes dependentes das condições iniciais do movimento,  $\Omega$  é a frequência de vibração e  $\xi$  é a taxa de amortecimento viscoso – suposta suficientemente pequena. A energia deste sinal, definida como o quadrado do módulo do sinal complexo, pode ser obtida da equação (45):

$$E_\Omega(t) = \rho^2 e^{-2\xi\Omega t} \quad (46)$$

O spectrograma, por sua vez, é definido por

$$SP(t, \omega) = \int v(\tau) h(\tau - t) e^{-i\omega\tau} d\tau \quad (47)$$

onde  $SP(t, \omega)$  é o spectrograma e  $h(t)$  é uma função de enjanelamento qualquer. Assumindo-se como função de enjanelamento a gaussiana  $h(t) = e^{-(t/\sigma)^2}$ , substituindo na equação anterior a expressão do sinal amortecido, equação (45), e calculando-se analiticamente o resultado, o spectrograma terá a forma

$$SP(t, \omega) = \sigma\sqrt{\pi} \cdot e^{\frac{\sigma^2\Omega^2\xi^2}{4} - \frac{\sigma^2}{4}(\Omega - \omega)^2 - 2\xi\Omega t} \quad (48)$$

Substituindo-se as expressões do spectrograma (48) e da energia (46) na definição da TFE (43), acha-se a função de transferência de energia para o spectrograma

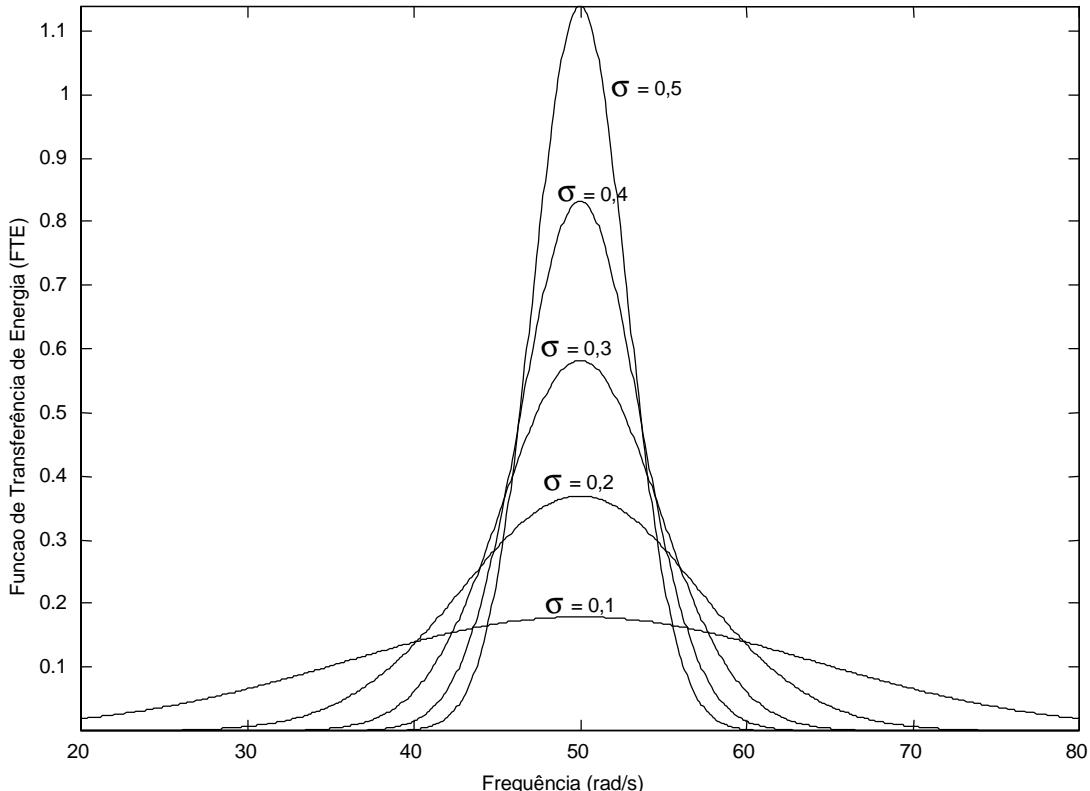
$$\alpha(\omega, \Omega) = \frac{\sigma\sqrt{\pi}}{\rho^2} \cdot e^{\frac{\sigma^2\Omega^2\xi^2}{4} - \frac{\sigma^2}{4}[\Omega - \omega]^2} \quad (49)$$

Observe que a função de transferência de energia não é somente dependente das frequências  $\Omega$  e  $\omega$  como também do parâmetro da gaussiana  $\sigma$ , da taxa de amortecimento  $\xi$  e da amplitude inicial  $\rho$ , ou seja,  $\alpha = \alpha(\omega, \Omega, \sigma, \rho, \xi)$ . De uma forma geral pode-se escrever que  $\alpha = \alpha(\text{TFD, MODELO})$ , isto é, a função de transferência de energia é dependente tanto do tipo de transformada tempo-frequência (TFD) utilizada – o spectrograma no caso – como também do modelo matemático usado para definir o amortecimento – no caso a equação (45). Devido a esta complexidade é de bom senso fazê-la independente do tempo tal que seja eliminada das equações finais.

Para ter completo entendimento do que seja a função de transferência de energia (FTE) mostra-se na Figura 24 um gráfico da FTE para um modelo definido pela equação (45) com os seguintes parâmetros:

$$\begin{aligned}\rho &= 1 \\ \xi &= 2\% \\ \Omega &= 50 \text{ rad/s}\end{aligned}$$

A largura efetiva da janela gaussiana  $\sigma$  variou de 0,1 a 0,5 conforme indicado na Figura 24.



**Figura 24 Função de Transferência de Energia (FTE)**

Deve-se observar na Figura 24 que quanto maior a largura  $\sigma$  da janela do espectrograma mais concentrada estará a função de transferência de energia em torno da frequência modal e, consequentemente, também mais concentrada estará a transformada tempo-frequência. Isto se traduz em maior precisão em frequência e menor precisão no tempo.

Inversamente, quanto menor a largura da janela  $\sigma$  do espectrograma, mais espalhada estará a função de transferência de energia, o que resulta em menor precisão em frequência. No entanto quanto menor a largura da janela, menores são os problemas com efeitos de arredondamento de bordas como será visto adiante.

O ideal, portanto, é ter-se uma ferramenta de software que permita refazer os cálculos com rapidez para vários valores de largura de janela  $\sigma$  para que o usuário possa escolher o valor intermediário que melhor se adeque ao problema estudado.

## Análise de qualidade

O método apresentado, como está, possui algumas limitações graves que dificultam seu uso em situações não raras como, por exemplo, na análise de modos que possuam baixas frequências e/ou altas taxas de amortecimento.

Não há exatamente um limite, a qualidade da análise vai deteriorando-se gradativamente. O perigo é que o gráfico resultante da transformada continua mostrando-se suave e bem comportado, apesar de o sinal estar sendo muito mal representado. Antecipando alguns resultados dos ensaios experimentais, a Figura 58 demonstra bem esta perda de qualidade da análise. Veja-se que, apesar de a curva mostrar-se suave, o intervalo de incerteza ( $0,75s$ , igual a  $\frac{1}{4}$  do tamanho da janela utilizada) é apenas metade do tamanho da duração do sinal, que é de aproximadamente  $1,5s$ . O tamanho da janela está, portanto, claramente superdimensionado. A solução, neste caso, é diminuir o tamanho da janela de forma que este seja suficientemente menor que a duração do sinal. Mas esta redução não pode ser levada continuadamente pois haverá o risco de a largura da janela chegar à mesma ordem de grandeza do período do sinal. Existe a necessidade, portanto, de estabelecer-se limites. Para tanto, definem-se os termos:

- $\sigma$  - Largura útil da janela (*window*) do spectrograma. No caso da Blackman, gaussiana, Hamming e Hanning (OPPENHEIM, SCHAFER, 1989), pode ser tomada como  $\frac{1}{4}$  da largura nominal  $L$ .
- $T$  – Período do modo, igual ao inverso da frequência do modo  $f$ , ou igual a  $2\pi/\Omega$ ;
- $f$  - Frequência do modo (Hz);
- $\Omega$  - Frequência angular do modo em rad/s;
- $Dm$  – Duração do modo, ou seja, o tempo útil de existência do modo no sinal. Para um experimento com impacto, por exemplo, é igual ao intervalo de tempo entre o instante do impacto e o instante quando o amortecimento leva a amplitude do modo à mesma ordem de grandeza do ruído de fundo;
- $\eta$  - fator de contaminação, menor que 1,0. Indica quanto do trecho analisado está sujeito a interferência de efeitos de borda. Como limite de qualidade adota-se empiricamente o valor máximo de 0,25.

Tendo-se estabelecido estas variáveis básicas, pode-se definir os limites para a largura da janela em dois extremos, a saber:

- Limite inferior da janela: quando  $\sigma \approx T$ ;
- Limite superior da janela: quando  $\sigma \approx Dm$ ;

Estes limites podem ser definidos com relação ao fator de contaminação  $\eta$  como

$$\frac{T}{\eta} = \frac{1}{\eta f} < \sigma < \eta Dm \quad (50)$$

Vê-se que existe um limite onde os valores extremos são iguais, ou seja, a possibilidade de uso do espectrograma no cálculo do amortecimento é ditado pelo limite último

$$\eta^2 > \frac{1}{f \cdot Dm} \quad (51)$$

Para o caso particular da Figura 60, novamente antecipando-se alguns resultados dos ensaios experimentais, tem-se que  $Dm \approx 0.4s$ ,  $f = 56Hz$ . Adotando-se um fator de contaminação de 0,25 tem-se os limites:

$$\frac{1}{0,25 \cdot 56Hz} < \sigma < 0,25 \cdot 0,4s \quad \text{ou} \quad 0,07s < \sigma < 0,10s \quad (52)$$

e

$$\eta > \sqrt{\frac{1}{56 \cdot 0,4}} = 21\% \quad (53)$$

Nota-se que o índice de contaminação mínimo é elevado (21%) e o valor útil para a largura da janela adotado nas análises (0,75s) é irreal para este modo em particular.

Este tipo de análise de qualidade deve ser feita para cada modo separadamente em todas as análises. Para tanto a duração do modo deveria ser obtida olhando-se o gráfico de uma TFD de boa qualidade como a SAF. No entanto isto nem sempre é possível ou realizável em um tempo limitado. Seria interessante, portanto, poder-se expressar a duração do modo  $Dm$  através da taxa de amortecimento  $\xi$ , para a qual podem ser obtidos limites máximos através da experiência. Desta forma é necessário criar-se uma constante adicional, a razão de amplitudes  $R$ , definida como a razão entre a amplitude máxima do modo ( $A_{max}$ ) e a amplitude mínima ( $A_{min}$ ) – a amplitude do ruído.

$$R : \frac{A_{max}}{A_{min}} \quad (54)$$

Sendo a energia do modo igual ao quadrado do módulo da amplitude, então

$$R^2 = \frac{E_{max}}{E_{min}} \quad (55)$$

Usando esta última equação e (41) tem-se que

$$v_0 = \frac{\log(E_{\max})}{2\Omega} = \frac{\log(R^2 E_{\min})}{2\Omega} = \frac{2\log(R)}{2\Omega} + \frac{\log(E_{\min})}{2\Omega} \quad (56)$$

$$v_1 = \frac{\log(E_{\min})}{2\Omega} \quad (57)$$

Supondo-se a taxa de amortecimento  $\xi$  constante, pode-se reescrever (40) como

$$\xi = -\frac{\partial v}{\partial t} = -\frac{v_1 - v_0}{Dm} = \frac{2\log R}{2\Omega Dm} = \frac{\log R}{2\pi f \cdot Dm} \quad (58)$$

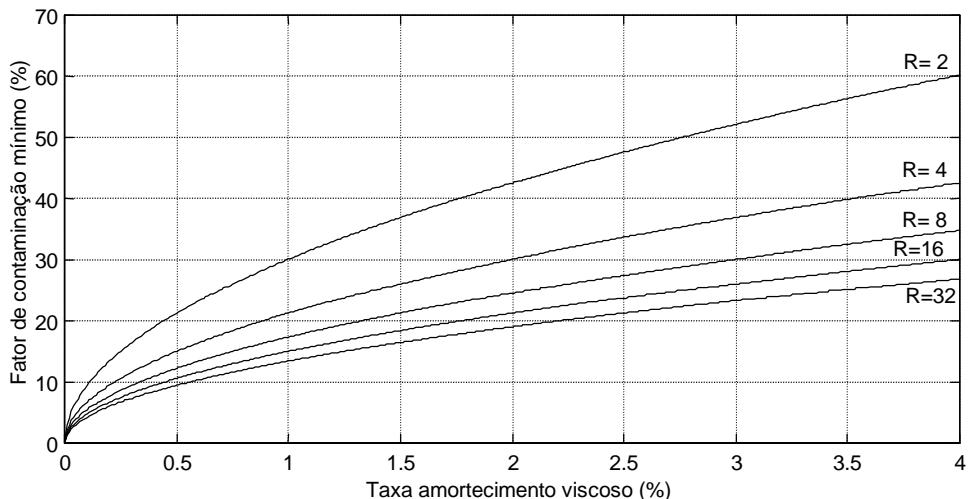
Tendo-se em vista (51) chega-se finalmente à expressão para o fator de contaminação  $\eta$  em função da taxa de amortecimento  $\xi$

$$\eta > \sqrt{\frac{2\pi\xi}{\log R}} \quad (59)$$

Esta expressão informa que se um sinal tiver uma taxa de amortecimento constante  $\xi$  e a razão entre sua amplitude máxima e mínima for  $R$  então o menor fator de contaminação possível de ser obtido com o uso do espectrograma, seja qual for o tipo ou largura da janela, é

$$\eta_{\min} = \sqrt{\frac{2\pi\xi}{\log R}} \quad (60)$$

A Figura 25 mostra os fatores de contaminação mínimos calculados de acordo com os resultados obtidos. Nota-se que para altas taxas de amortecimento fica difícil manter-se fatores de contaminação baixos caso o modo não seja excitado com amplitude suficientemente maior que o ruído de fundo (valores de  $R$  baixos). Para o caso da meia-vida ( $R=2$ ) nota-se que uma taxa de amortecimento de apenas 1% induz um comprometimento *mínimo* de 30% do trecho útil do sinal.



**Figura 25 Fatores de contaminação mínimos**

A relação de amplitudes  $R$  pode ser obtida diretamente do gráfico da curva de amortecimento pela relação

$$R = e^{-\Omega \Delta v} \quad (61)$$

Para o caso da Figura 60 observa-se que  $\Delta v = 0.016 - 0.0178 = -1.8E-3$ . Deve-se observar que o valor de  $\Delta v$  será sempre negativo pois a amplitude estará diminuindo. O valor da exponencial deverá, portanto, ser sempre positivo. A relação de amplitudes é, portanto, igual a

$$R = e^{-(2\pi 56)(-1.8E-3)} = e^{0.633} = 1.88 \quad (62)$$

O fator de contaminação mínimo para esta análise utilizando-se o spectrograma será, considerando-se *a priori* uma taxa de amortecimento de 0,5%,

$$\eta_{min} = \sqrt{\frac{2\pi(0.5\%)}{0,633}} = 22\% \quad (63)$$

A largura de janela (útil) correspondente a este fator de qualidade máximo será, de acordo com (50),

$$\sigma = \frac{1}{\eta_{min} \cdot f} = \frac{1}{0,22 \cdot 56Hz} = 0.08s \quad (64)$$

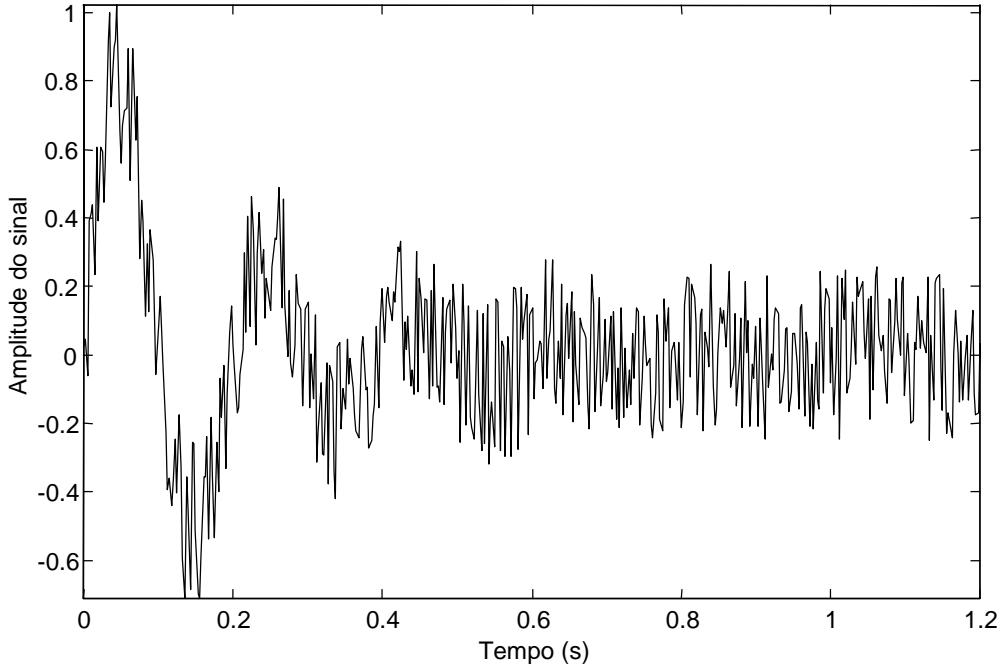
que corresponde a uma janela de largura nominal igual a  $4 \times 0,08 \approx 0,3s$ .

## Modulação

Existem casos extremos, no entanto, nos quais o limite último (51) fornece índices de contaminação acima de 100%. Este é o caso de modos de baixa frequência com taxas de amortecimento muito altas, como é o caso mostrado na Figura 26, onde se mostra um sinal constituído de uma senóide a 5Hz amortecida a uma taxa de 15% à qual foi adicionado ruído da ordem de SNR=1 (relação entre a norma quadrática do sinal e a norma quadrática do ruído). A amplitude inicial da senóide é 1,0.

Vê-se que o sinal inicia com uma alta amplitude mas, devido à alta taxa de amortecimento, o sinal já está coberto pelo ruído de fundo por volta do instante 0,7s. Analisando-se este trecho válido verifica-se, entretanto, que só existem dois ou no máximo três ciclos para serem analisados, o que é muito pouco devido ao fato de o spectrograma, ou mesmo outras técnicas tempo-frequência – precisarem de vários ciclos para obterem um bom resultado. A análise do sinal mostrado na Figura 26 por meio somente das metodologias desenvolvidas até este ponto do texto revelaria-se uma

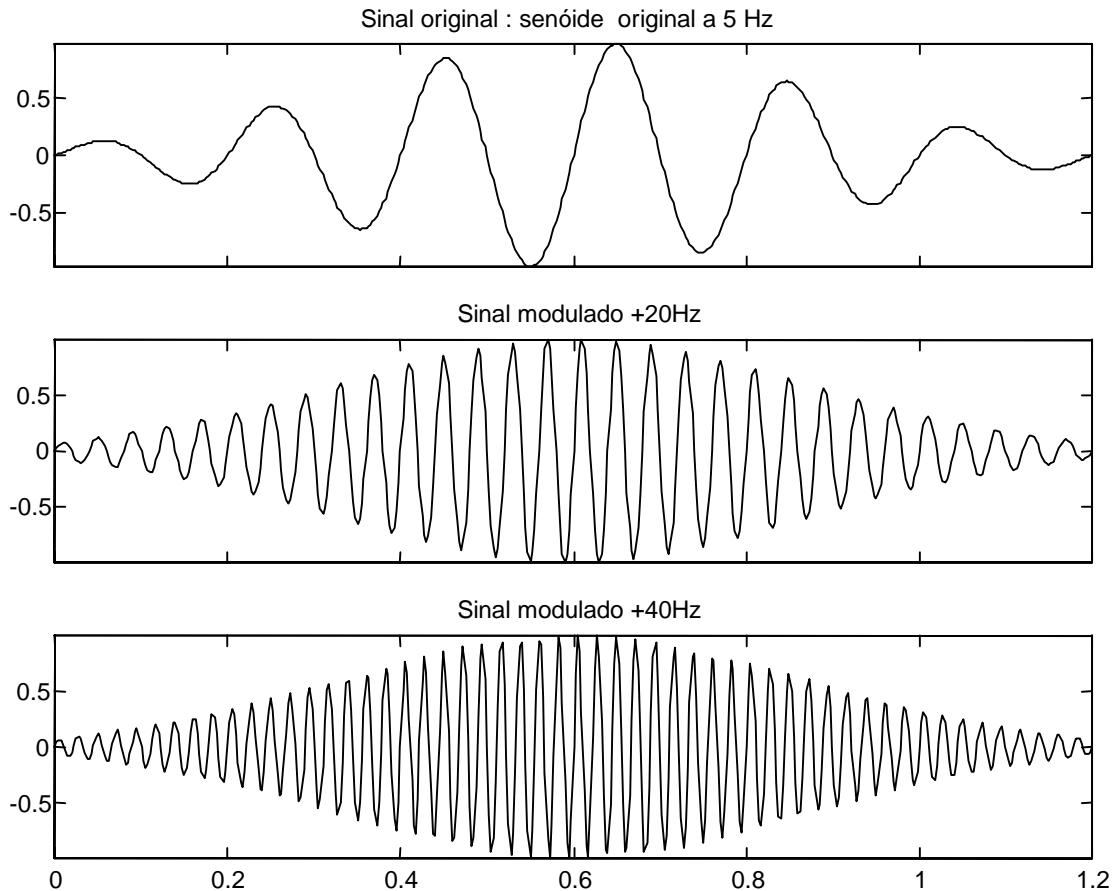
grande decepção exatamente por este motivo. Existe felizmente uma forma de contornar esta situação.



**Figura 26 Senóide a 5Hz, taxa amortecimento 15%, ruído adicionado a SNR=1**

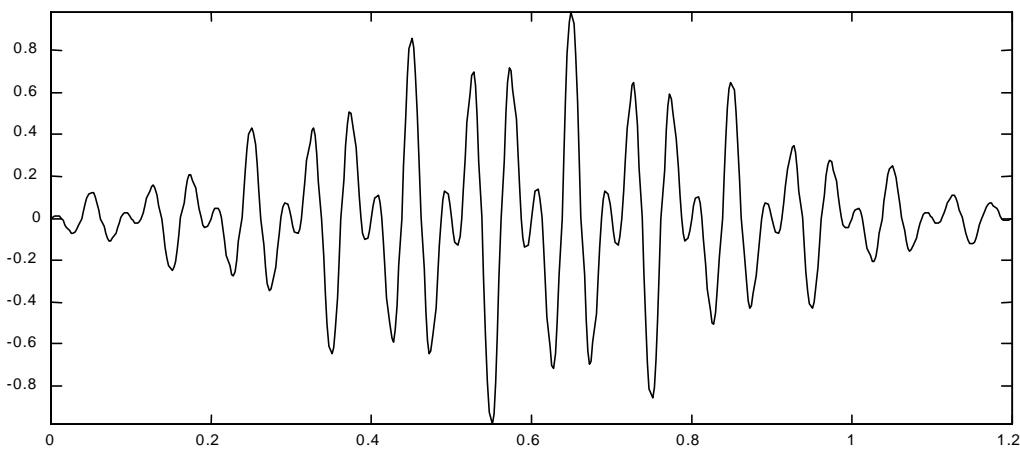
Observe-se que a inequação (51) relaciona o índice de contaminação  $\eta$  a duas outras características modais, a saber, sua frequência  $f$  e seu período de vida útil  $Dm$ . Considerando-se impossível modificar a duração  $Dm$  do modo, resta tentar de algum modo alterar sua frequência  $f$  de tal forma a reduzir-se o fator de contaminação  $\eta$ . O aumento artificial da frequência que diminuirá o fator de contaminação é conseguido através de uma fase de pré-processamento do sinal onde aplica-se uma operação denominada *modulação*, isto é, um translado do espectro em frequência.

A Figura 27 mostra um sinal-exemplo que foi pré-modulado duas vezes. O sinal é um seno com frequência de 5Hz envolvido por uma janela gaussiana. A esta frequência este sinal possui 6 ciclos no intervalo de 0 a 1,2s. Ao modular-se o sinal em 20Hz, sua frequência aumentará para 25Hz, o que deverá quintuplicar o número de ciclos no intervalo referido. De fato, pode-se contar que o número de ciclos na segunda figura é 30. Modulando-se a 40Hz, sua frequência aumentará para 45Hz, uma frequência nove vezes maior que a original. Contando-se o número de ciclos da última figura tem-se o número de 54 como esperado. Deve-se observar que a envoltória dos picos continua exatamente a mesma, isto é, a continuidade da energia do sinal é preservada durante a modulação. Esta continuidade da energia faz-se extremamente importante pois é através de sua variação que calculam-se as taxas de amortecimento.



**Figura 27 (1) Senóide originalmente a 5Hz; (2) modulada a 20Hz e (3) modulada a 40Hz**

É relativamente simples realizar-se um translado em frequência num sinal complexo. Deve-se apenas multiplicá-lo por  $e^{i\Delta\Omega t}$  onde  $\Delta\Omega$  é o translado (*shift*) requerido em frequência. No presente caso existe a complicação de que o resultado da modulação deve ser ainda um sinal real. Visto que a modulação é realizada multiplicando-se o sinal original por uma exponencial complexa, isto não é possível a menos que certos cuidados sejam tomados.



**Figura 28 Sinal incorretamente modulado (a 20Hz)**

Apenas para maior esclarecimento do problema, mostra-se na Figura 28 um sinal que foi modulado apenas multiplicando-se o sinal original por  $e^{i\Delta\Omega t}$  e retendo-se a parte real do sinal complexo obtido. Não é preciso plotar a transformada de Fourier deste sinal para provar-se que o resultado é incompatível com o objetivo desejado, isto é, a correta representação da continuidade da energia do sinal.

Modulação é uma técnica já amplamente conhecida no campo da teoria de comunicação e processamento de sinais (VOELCKER, 1966, BRUN, 1991, RUBIN, FRANCO, 1966). A tarefa de modular um sinal real passa pela definição de conceitos como *amplitude instantânea* e *fase instantânea* (PANTER, 1965, WOZENCRAFT, JACOBS, 1965, OPPENHEIM, WILLSKY, 1983, SCHWARTZ, 1987, TAUB, SCHILLING, 1986). Neste escopo, a frequência – também instantânea – é definida como a derivada (no tempo) da fase. Desta forma um sinal real qualquer  $x(t)$  passa a ser descrito pela forma geral

$$x(t) = a(t) \cos(\phi(t)) \quad (65)$$

onde  $a(t)$  é a amplitude instantânea do sinal e  $\phi(t)$  sua fase instantânea, ambas funções reais.

É trivial obter-se a função real a partir de sua amplitude e fase instantânea mas o caminho inverso, isto é, obter-se um par único  $[a(t), \phi(t)]$  a partir de um sinal real  $x(t)$ , não tem solução única. Uma solução particular para (65) é  $\phi(t)=K$  e  $a(t)=x(t)/K$ , onde  $K$  é uma constante arbitrária. Outras soluções podem ser criadas por simples inspeção visual.

Na realidade a forma de obter-se uma relação biunívoca entre estas funções é bem conhecida e explicitamente descrita por PICINBONO (1983), PICINBONO (1988) e também por BOASHASH (1992). VAKMAN (1972) e VAKMAN (1996) mostram a prova de que, assumindo-se certas condições físicas *a priori*, esta é a única forma de associar-se uma fase e amplitude instantâneas a um sinal real. Outras suposições podem, no entanto, levar a definições diversas (LOUGHLIN, TACER, 1996).

Associado ao par de funções  $[a(t), \phi(t)]$  está uma função complexa, denominada de *sinal analítico* associado à função real  $x(t)$  (GABOR, 1946, VILE, 1948, OSWALD, 1956, DUGUNDJI, 1958, BEDROSIAN, 1962)

$$z(t) = a(t) \cdot e^{i \cdot \phi(t)} \quad (66)$$

Uma discussão a respeito do significado físico do sinal analítico pode ser vista em (BOASHASH, 1992). Utilizando-se o teorema de Bedrosian (BEDROSIAN, 1963, GENDRIN, ROBERT, 1982) pode-se associar o sinal analítico  $z(t)$  ao sinal real  $x(t)$  através da transformada de Hilbert (OPPENHEIM, SCHAFFER, 1989)

$$z(t) = x(t) + i \cdot \mathbf{H}[x(t)] \quad (67)$$

A partir do sinal analítico pode-se obter um único par de amplitude e fase instantâneas por

$$a(t) = |z(t)| \quad (68)$$

$$\phi(t) = \arg(z(t)/a(t)) \quad (69)$$

Se a freqüência instantânea do sinal é a derivada da fase, ou seja,  $\dot{\phi}(t) = d\phi(t)/dt$  então a freqüência instantânea do sinal modulado deverá ser

$$\dot{\phi}_{\text{mod}}(t) = \dot{\phi}(t) + \Delta\Omega \quad (70)$$

que, integrando-se no tempo resulta em

$$\phi_{\text{mod}}(t) = \phi(t) + \Delta\Omega \cdot t + \phi_0 \quad (71)$$

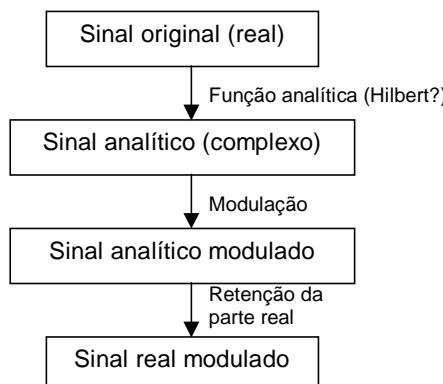
O sinal modulado será, portanto, segundo a definição (65)

$$\hat{x}(t) = a(t) \cos(\phi(t) + \Delta\Omega \cdot t + \phi_0) \quad (72)$$

Observe que os sinais original e modulado,  $x(t)$  e  $\hat{x}(t)$ , possuem a mesma amplitude instantânea, ou seja,  $|\hat{x}(t)| = |x(t)| = a(t)$ , o que garante que a modulação não modificará o desenvolvimento da energia  $a^2(t)$  com o tempo. Este último resultado pode ser reescrito de uma forma mais compacta, porém equivalente, como

$$\hat{x}(t) = \Re \langle [x(t) + i \cdot H(x)] \cdot e^{i\Delta\Omega t} \rangle \quad (73)$$

O algoritmo genérico de modulação, portanto, foi implementado como indicado na Figura 29.



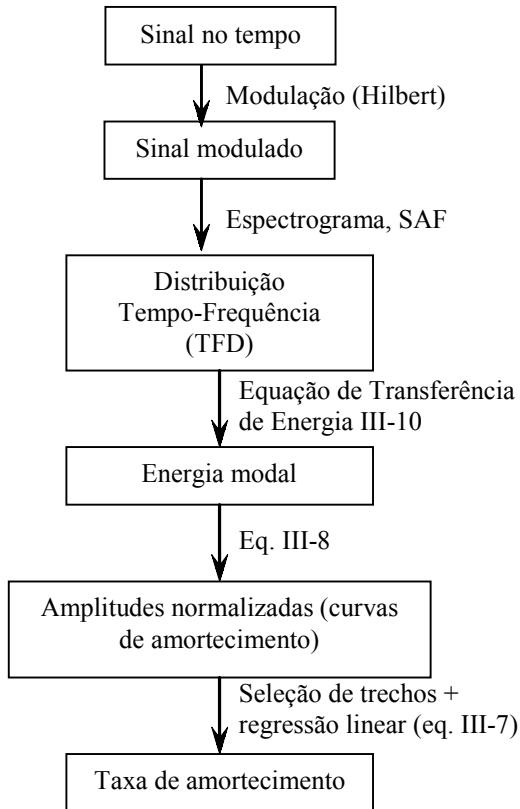
**Figura 29 Fluxo para correta obtenção do sinal real modulado**

Resta ainda uma correção de última hora. Supõe-se que o sinal pré-modulado seja submetido ao algoritmo padrão de cálculo das taxas de amortecimento desenvolvido anteriormente. Deve-se observar, no entanto, que a frequência utilizada na equação (41) não é a frequência real mas sim a frequência real acrescida da modulação. Portanto as taxas de amortecimento obtidas devem ser corrigidas segundo a razão destas frequências a menos que tenha sido previsto utilizar em (41) a frequência real e não a frequência pré-modulada. No caso do algoritmo não prever a pré-modulação deve-se corrigir a taxa de amortecimento obtida com

$$\xi_{corrigido} = \frac{\Omega + \Delta\Omega}{\Omega} \cdot \xi_{calculado} \quad (74)$$

Desta forma o limite (51) deixa de ser problema e a escolha de índices de contaminação arbitrariamente baixos está condicionada somente à pré-modulação utilizando-se um valor de translado  $\Delta\Omega$  suficientemente grande.

A metodologia de cálculo do amortecimento utilizando as técnicas tempo-frequência, acrescida da técnica de modulação, pode ser condensada da forma ilustrada na Figura 30.

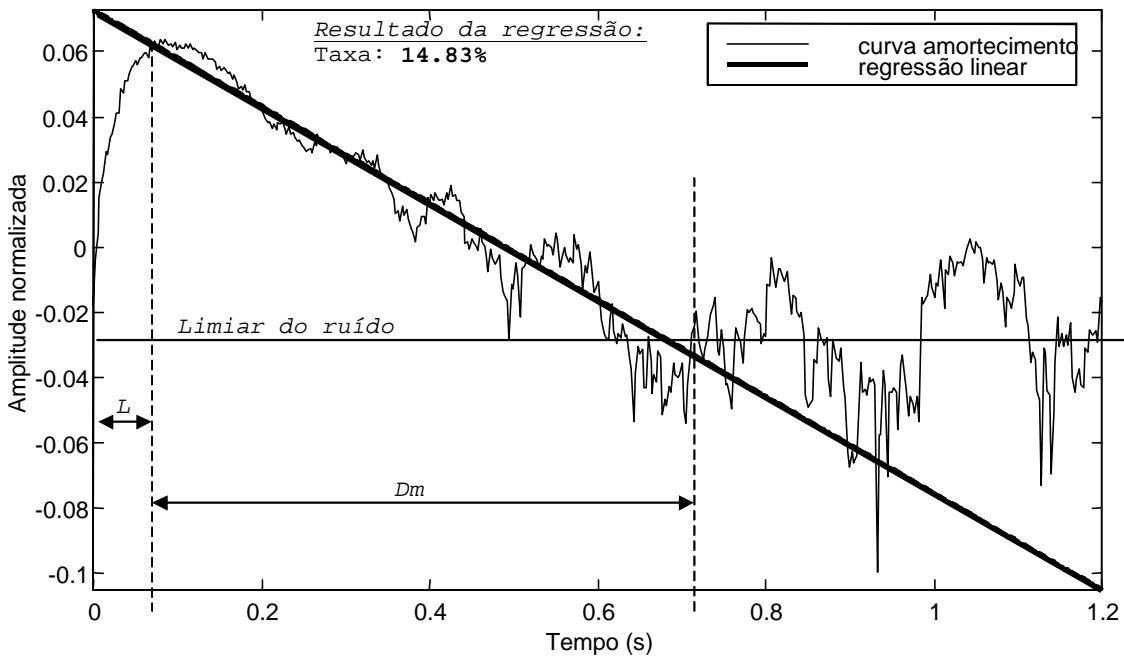


**Figura 30 Metodologia de cálculo da taxa de amortecimento**

Inicialmente calcula-se a TFD através de alguma distribuição. No caso presente, foi utilizado o spectrograma por ser rápido. A próxima etapa só existe na teoria, que seria aplicar a função de transferência de energia (43) à TFD recém-calculada para obter-se a energia relacionada ao modo. Na prática, como já foi sinalizado, utilizam-se os valores da TFD diretamente, isto é,  $E_\Omega(t) = C_f(t, \Omega)$  desde que a função de transferência de energia FTE seja independente do tempo. Como pretende-se utilizar o spectrograma, esta exigência está automaticamente garantida.

A partir da curva de energia  $E = E_\Omega(t)$ , utiliza-se a equação (41) para calcular a amplitude normalizada e obter a *curva de amortecimento*, isto é, um gráfico de  $\hat{v}(t)$ . Os trechos de interesse são então separados e usa-se regressão linear para ajustar uma reta a cada trecho. A inclinação da reta, de acordo com a equação (40) será igual ao amortecimento médio no trecho.

A Figura 31 mostra a curva de amortecimento obtida para o sinal mostrado na Figura 26 e também os critérios utilizados para seleção do trecho válido para regressão linear.

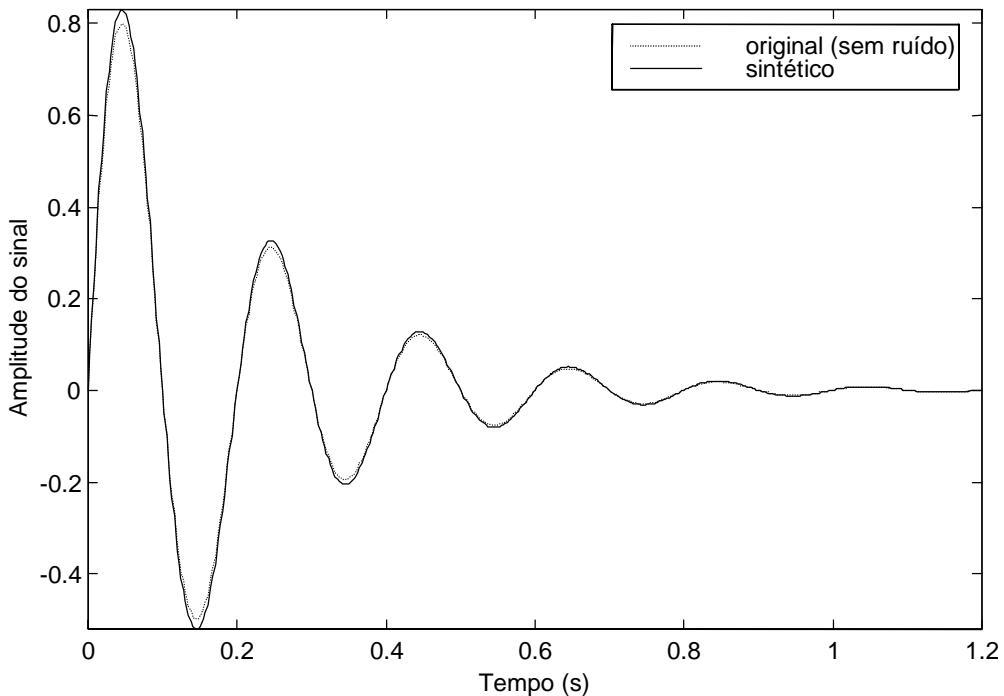


**Figura 31 Curva de amortecimento demonstrando critérios usados para regressão**  
Vê-se que a curva de amortecimento inicia com uma curva ascendente curta. Este é o trecho dito “contaminado”, isto é, o intervalo que a TFD leva para estabilizar e começar a responder a um sinal subitamente introduzido. Este trecho tem largura aproximadamente igual à largura da janela, se o critério for conservador, ou a  $\frac{1}{4}$  ou ainda  $\frac{1}{2}$  deste valor se o critério for mais liberal. A seguir a energia do sinal cai

gradativamente e então a curva entra numa região na qual este comportamento desaparece dando lugar a uma oscilação sem sentido aparente.

Um auxílio geométrico, como se mostra na Figura 31, pode ser obtido traçando-se sobre o gráfico uma linha denominada *limiar do ruído*, que é uma linha horizontal que indica a amplitude média estimada do ruído de fundo. Com esta linha posta sobre a curva de amortecimento fica fácil estimar a posição da segunda linha vertical indicatória do fim do intervalo válido para regressão. Este intervalo válido, entre o fim da área contaminada e o início da cobertura do sinal pelo ruído é que deve ser submetido à regressão linear.

Tendo-se feita a regressão linear, a taxa de amortecimento viscoso é diretamente obtida do coeficiente do termo de ordem um (valor negativo deste). Para definir-se completamente a curva original resta apenas obter a fase que pode ser calculada através de um método de mínimos quadrados ou um método de minimização de erro simples como uma procura binária no intervalo  $[0, 2\pi]$ . Contudo a fase não é uma constante importante de ser conhecida, a menos que se queira estimar o sinal original calculando-se o sinal sintético, isto é, calculado usando-se o modelo numérico, como mostrado na Figura 32. O erro RMS (*Root Mean Square*) entre os sinais original – sem ruído – e o sintético, neste caso, foi de apenas 0,18%.



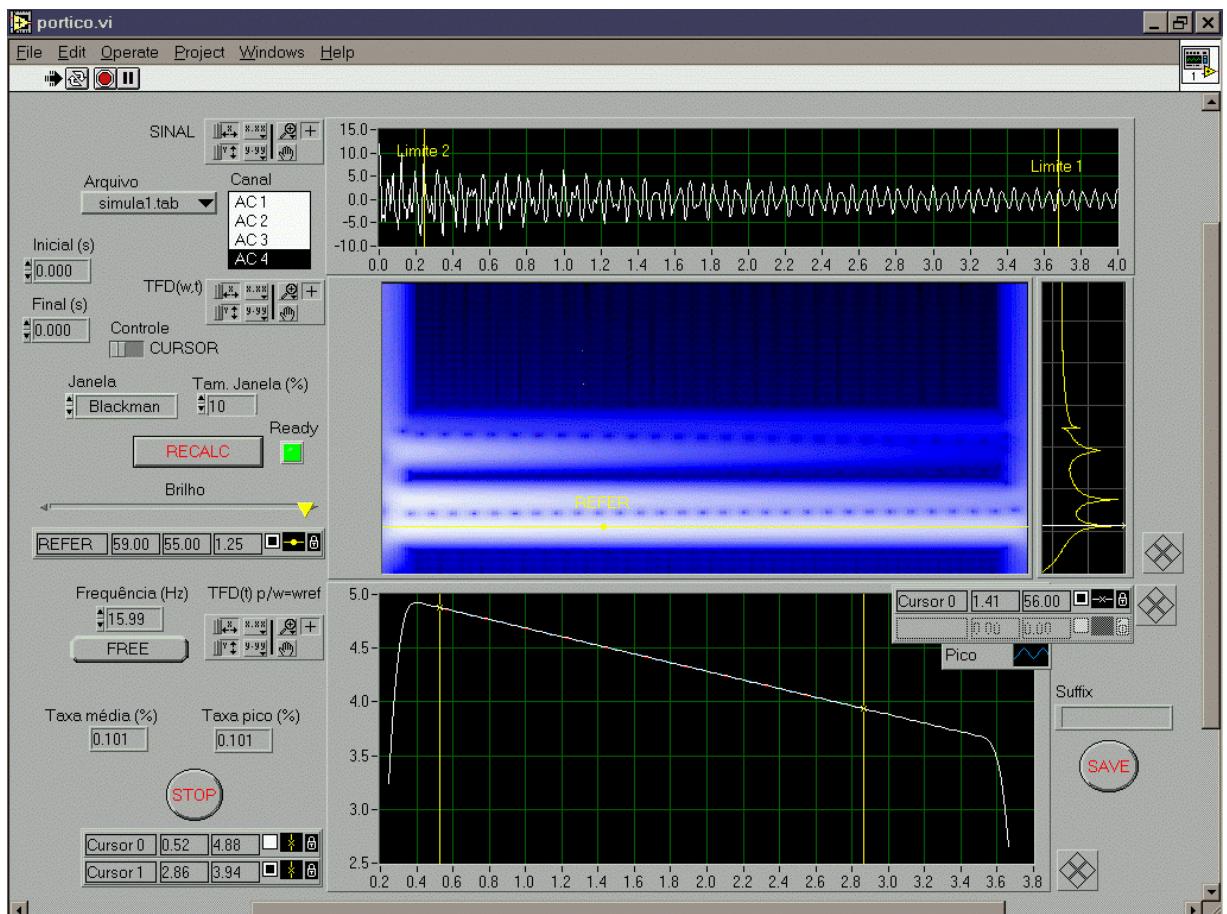
**Figura 32 Sinal original, sem ruído, e sintético/calculado: erro RMS igual a 0,18%**

A Figura 32 mostra o sinal original sem ruído adicionado e o sinal reconstruído utilizando-se a taxa de amortecimento obtida. Vê-se que a correlação entre os gráficos é perfeita.

Neste caso o resultado obtido para a taxa de amortecimento apresentou um erro de apenas 0.17% (de um valor total de 15%) apesar de o sinal apresentar apenas três ciclos e estar bastante contaminado pelo ruído. Isto demonstra a robustez do método desenvolvido.

### **III.3 Implementação computacional**

Para aumentar a iteratividade do programa, originalmente desenvolvido como um script em Matlab, produziu-se um software gráfico baseado na ferramenta LabView.



**Figura 33 Tela do software usado para o cálculo do amortecimento**

Na Figura 33 mostra-se a tela principal do programa contendo quatro gráficos principais: o sinal no tempo (superior), a TFD calculada (centro, esquerda), a transformada de Fourier (centro, direita) e a curva de amortecimento (inferior). Para um melhor entendimento do sistema desenvolvido as principais partes da tela serão

mostradas em detalhes. Na Figura 34 é mostrado o sinal no tempo, onde pode-se observar dois cursores (*Límite 1* e *Límite 2*). Estes cursores, posicionados sobre o sinal no tempo, limitam a região sobre a qual a TFD será calculada (Figura 34).

É importante delimitar este trecho com cuidado pois, via de regra, quanto menor o trecho, menor o tempo de cálculo. Estes cursores também são importantes quando no sinal existem modos com amortecimento muito alto ao lado de modos com baixo amortecimento. Para estudar os sinais com baixo amortecimento, estendem-se os cursores aos extremos. Para os modos de mais alto amortecimento, encurta-se o domínio destes cursores em torno do instante do impacto, visto que estes modos de alto amortecimento têm curta duração.

Na Figura 35 são mostradas as funções de alguns controles que o usuário possui para a análise.

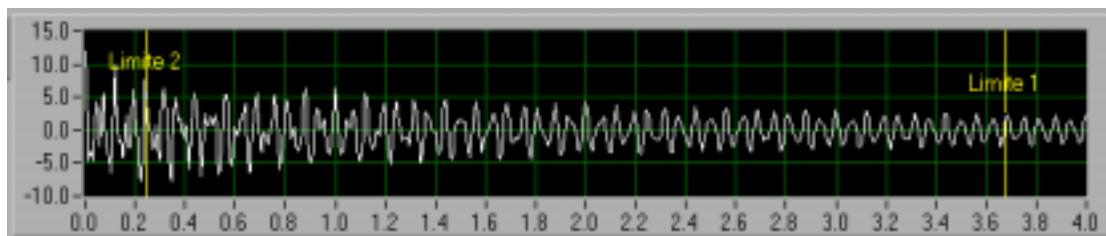
Como o espectrograma está sendo utilizado, existem dois controles (Figura 35) que possibilitam escolher o tipo da janela (Blackman, Hanning, Hamming, Gaussian ou Retangular) e seu tamanho (%) relativo ao comprimento total do sinal. O botão *RECALC* é usado para iniciar o cálculo da TFD. Após o cálculo, um controle *brilho* é usado para destacar/ocultar os detalhes da TFD.

Existe um cursor sobre a TFD que se move verticalmente e que permite escolher a frequência para a qual o amortecimento será calculado conforme pode ser visto na Figura 36. A frequência escolhida aparece num indicador à esquerda. Uma vez posicionado o cursor sobre a frequência desejada na TFD, a curva de amortecimento, já calculada, aparece no gráfico inferior.

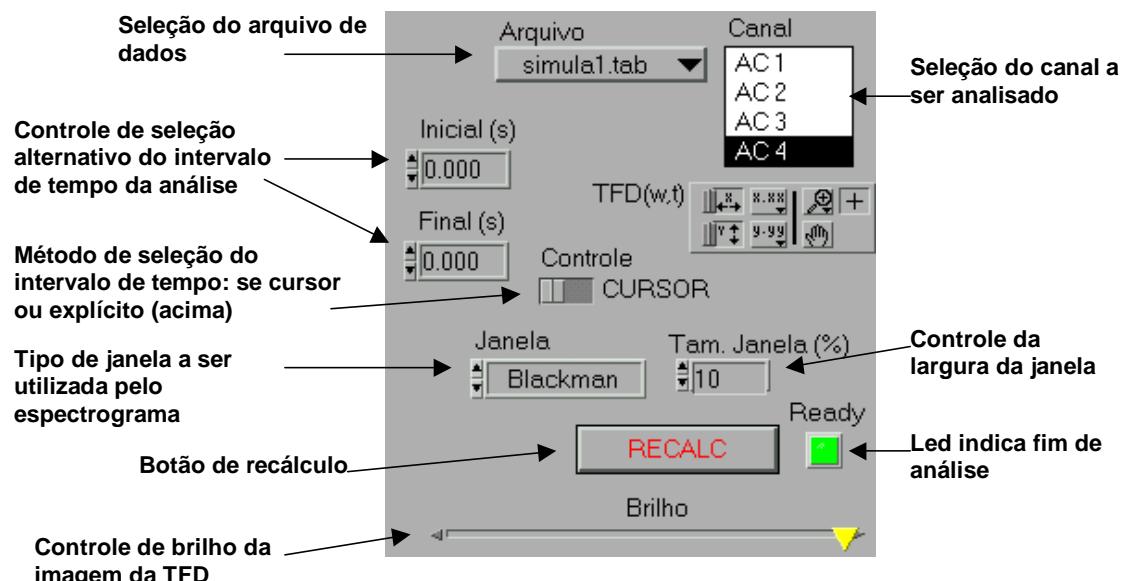
Dois cursores sobre a curva de amortecimento (Figura 37) permitem que se selecione o trecho sobre o qual será aplicada a regressão linear para calcular a inclinação da curva, que será exatamente o *amortecimento médio*, cujo valor é mostrado num indicador à esquerda (Figura 38).

Este valor é denominado *médio* em contraste com outro valor de amortecimento calculado chamado de *amortecimento de pico*, que é calculado através da inclinação da reta que une os dois pontos extremos criados pela interseção do cursor com a curva de amortecimento. Este amortecimento de pico foi adicionado pela necessidade de uma maior flexibilidade em casos de difícil interpretação quando, por exemplo, da ocorrência de acoplamento modal.

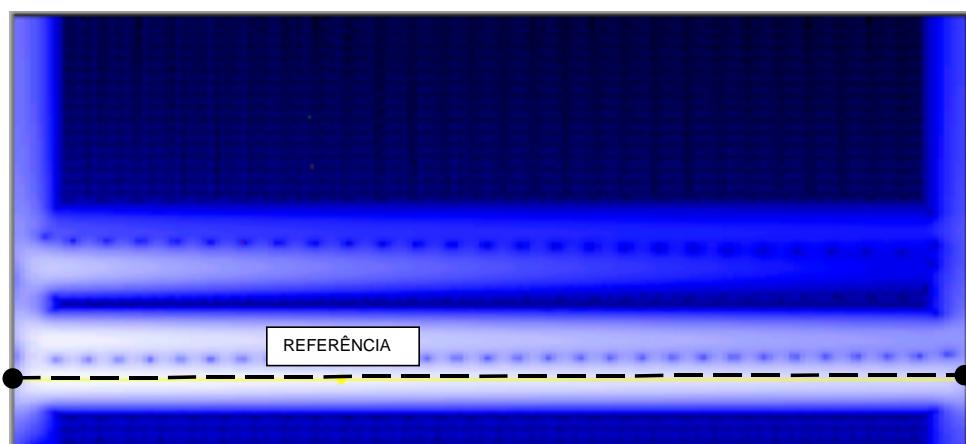
Todos os resultados podem ser salvos em um arquivo texto pressionando-se o botão **SAVE** (Figura 39). O nome do arquivo é gerado automaticamente a partir do nome do arquivo de dados inicial, do canal e da frequência fixada. A este nome pode ser acrescentado um sufixo qualquer a ser definido no momento do salvamento dos resultados.



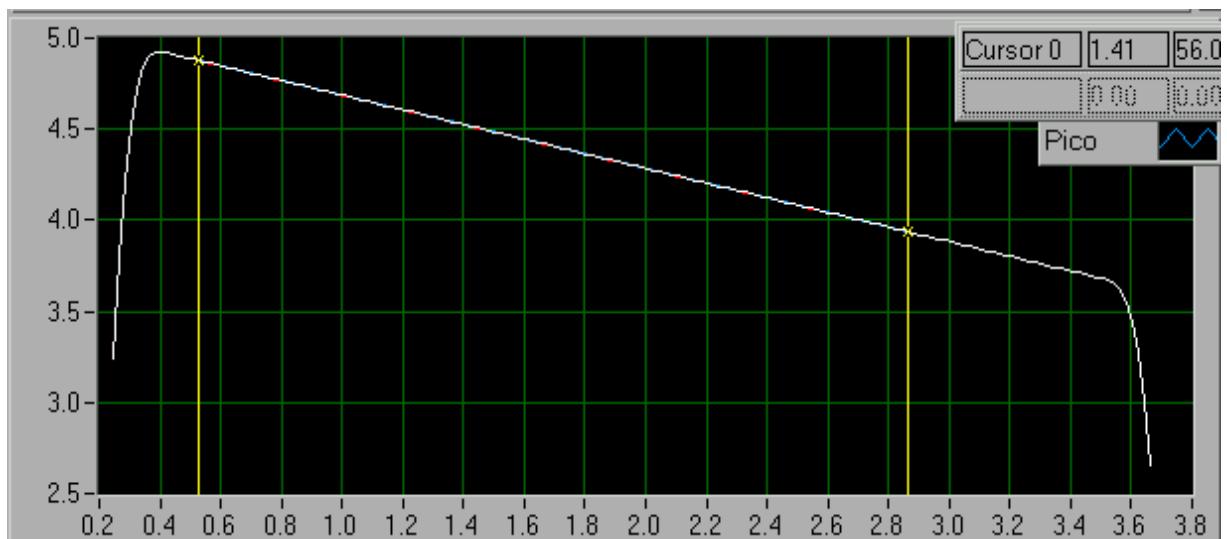
**Figura 34** Detalhe da seleção do intervalo de tempo a ser analisado



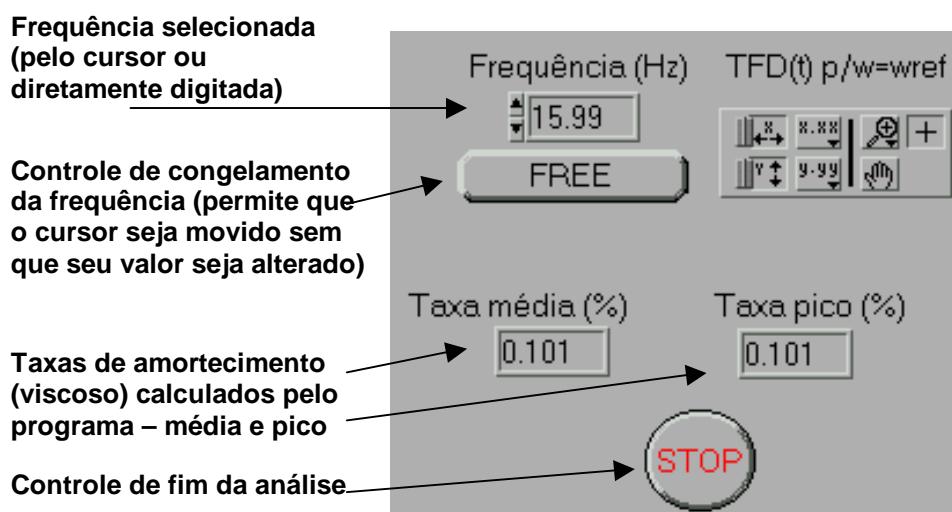
**Figura 35** Detalhe dos controles da análise



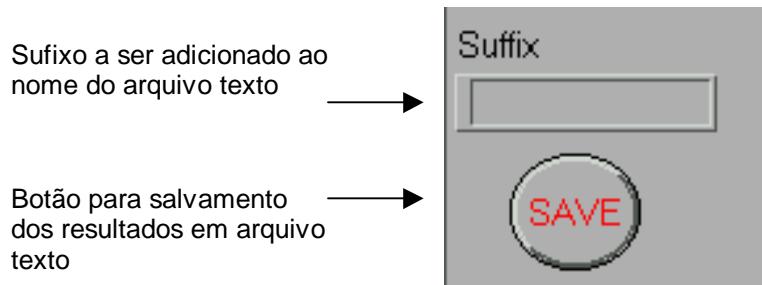
**Figura 36** Detalhe da seleção da frequência de análise - cursor horizontal sobre a TFD



**Figura 37 Curva de amortecimento  $[\log E(t)/\omega]$  - detalhe da seleção do trecho para regressão linear**



**Figura 38 Seleção da frequência e resultados finais obtidos (taxas de amortecimento)**



**Figura 39 Detalhe dos controles para salvamento dos resultados**

### III.4 Simulação numérica

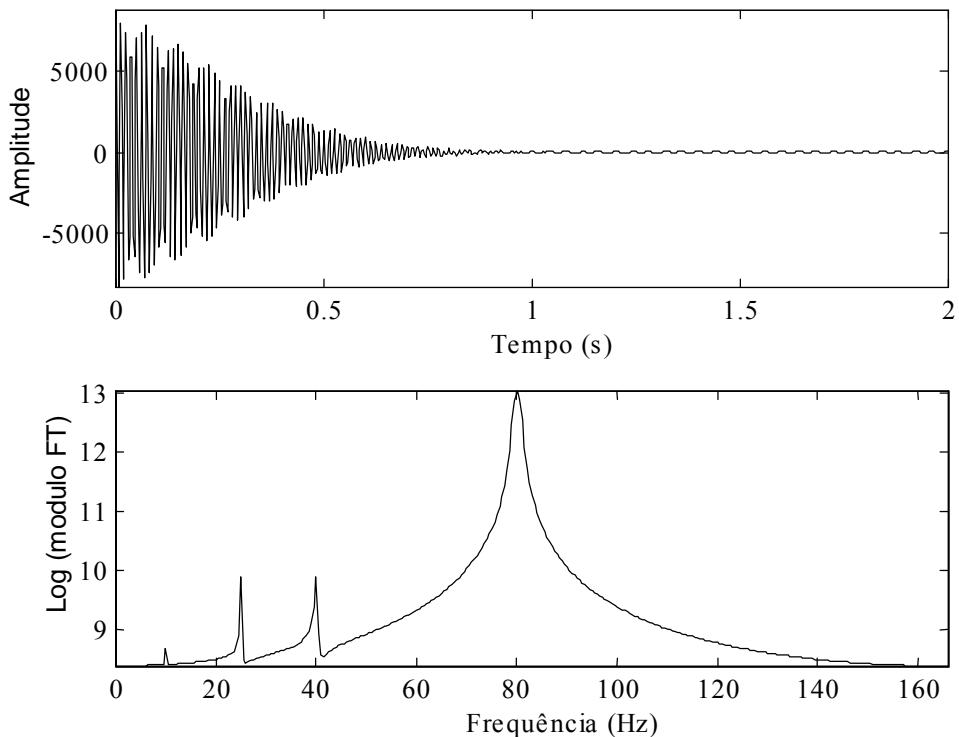
Uma simulação foi elaborada com o intuito de se verificar a exatidão do método apresentado. Um sinal  $v(t)$  foi construído usando-se quatro senóides ( $j = 1$  até 4) amortecidas de frequências  $\Omega_j/2\pi$  iguais a 10, 25, 40 e 80Hz, tais que

$$\xi_j(t) = \begin{cases} 0.1\% & j=1 \\ 0.2\% & j=2 \\ 0.3\% & j=3 \\ 0.1\% + 0.95\% \cdot t & j=4 \end{cases} \quad (75)$$

e

$$(\text{amplitude}) A_j = \begin{cases} 5 & j=1 \\ 81 & j=2 \\ 140 & j=3 \\ 8500 & j=4 \end{cases} \quad (76)$$

Estes valores foram arranjados de tal forma que os gráficos finais tivessem a aparência o mais didática possível. Isto não ocorrerá, contudo, para os ensaios experimentais, nos quais o método continuará a fornecer resultados claros e confiáveis, conforme será visto mais tarde.

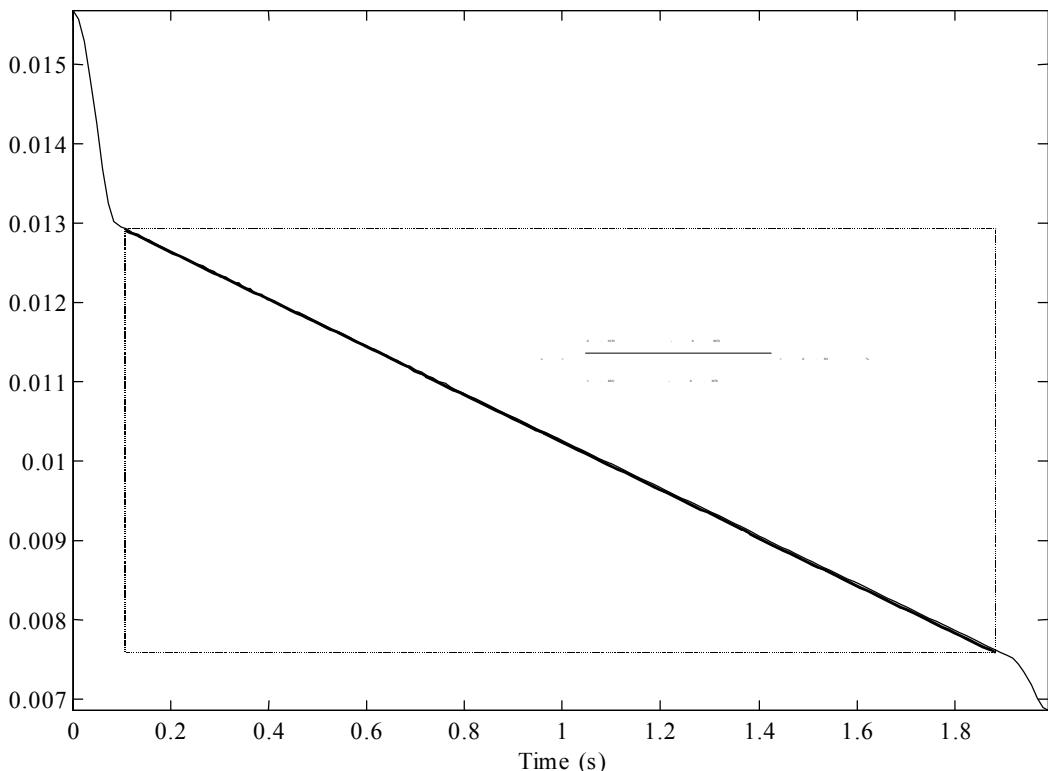


**Figura 40** Sinal objeto da simulação e respectiva transformada de Fourier

As expressões anteriores definem um sistema harmônico cujo amortecimento é constante para as três primeiras parcelas (0.1%, 0.2% e 0.3%) e proporcional ao tempo ( $0.1\% + 0.95t$ ) para a última parcela. Este amortecimento não-linear produzirá uma função com taxas de amortecimento variando linearmente com o tempo e iguais a 0.1%, 0.9%, 1.5% e 2.0% respectivamente para os instantes 0.0s, 0.8s, 1.4s e 2.0s. A Figura 40 mostra o sinal obtido através destas equações e seu respectivo espectro.

Pode-se observar na Figura 40 os picos referentes aos harmônicos nas frequências 10, 25, 40 e 80Hz. Observa-se também o grande abaulamento da transformada de Fourier nas regiões vizinhas aos picos, fato resultante do amortecimento imposto e também pelo fato de que a amplitude do espectro está plotada sobre uma escala logarítmica.

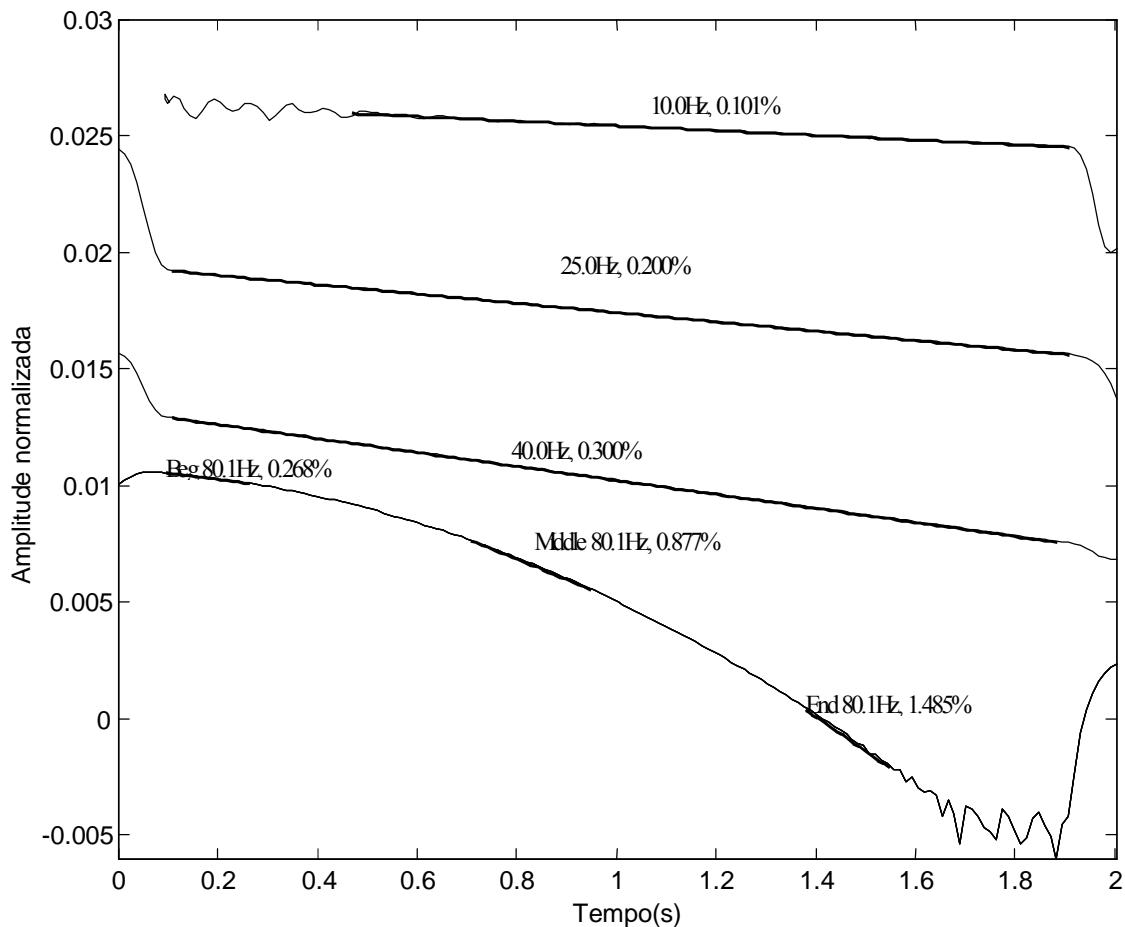
Com este sinal, calcula-se a sua respectiva distribuição tempo-frequência e retira-se desta distribuição bidimensional somente os valores referentes às frequências dos modos impostos ( $\Omega_j = 10, 25, 40$  e  $80\text{Hz}$ ). Isto fornecerá quatro funções  $C_f(t, \Omega_j)$  para  $j = 1, 2, 3$  e  $4$ . Cada função é então usada para calcular a respectiva amplitude normalizada  $\hat{v}_j(t)$  através da equação (41). Estas amplitudes normalizadas, quando plotadas versus tempo ( $t$ ) dão origem a gráficos denominados *curvas de amortecimento*. A curva de amortecimento para a frequência 40Hz é mostrada na Figura 41.



**Figura 41** Curva de amortecimento para a simulação, frequência 40Hz

A partir da curva de amortecimento é selecionado então um trecho que é então submetido à regressão linear. A reta obtida da regressão é mostrada na Figura 41 como uma linha escura e seus limites são ressaltados por uma linha pontilhada. Observa-se que o valor obtido da regressão linear concorda com o parâmetro da simulação até a terceira casa decimal, a menos de uma diferença devida a arredondamento, o que evidencia a precisão do método.

A Figura 42 mostra todas as curvas de amortecimento resultantes da simulação realizada, juntas. Esta figura mostra uma convenção que será adotada neste ponto em diante: quando a mesma curva de amortecimento for dividida em trechos para análise, o valor da frequência será precedida por uma indicação do posicionamento do trecho. Por exemplo: *Beg 80.1Hz, 0.268%* indica que a curva de amortecimento relativa à frequência de 80.1Hz foi dividida em trechos para análise e que este trecho corresponde ao início do sinal (*begin*).



**Figura 42 Curvas de amortecimento resultantes da simulação**

Percebe-se na Figura 42 que, como previsto, os amortecimentos referentes às frequências 10, 25 e 40Hz são constantes, o que é indicado por uma linha reta. No entanto, o mesmo não ocorre para a frequência de 80Hz cuja função se assemelha com uma parábola descendente, como previsto.

Os valores obtidos para as três frequências menores (10, 25 e 40Hz) novamente concordam com os parâmetros da simulação até a terceira casa decimal. Para a frequência 80Hz foram analisados três trechos: 0.1-0.4s, 0.7-0.9s e 1.35 a 1.55s. Os valores obtidos são, respectivamente, 0.268%, 0.877% e 1.485%, bastante próximos dos valores esperados de 0.300%, 0.900% e 1.500%, obtidos teóricamente para os mesmos intervalos de tempo.

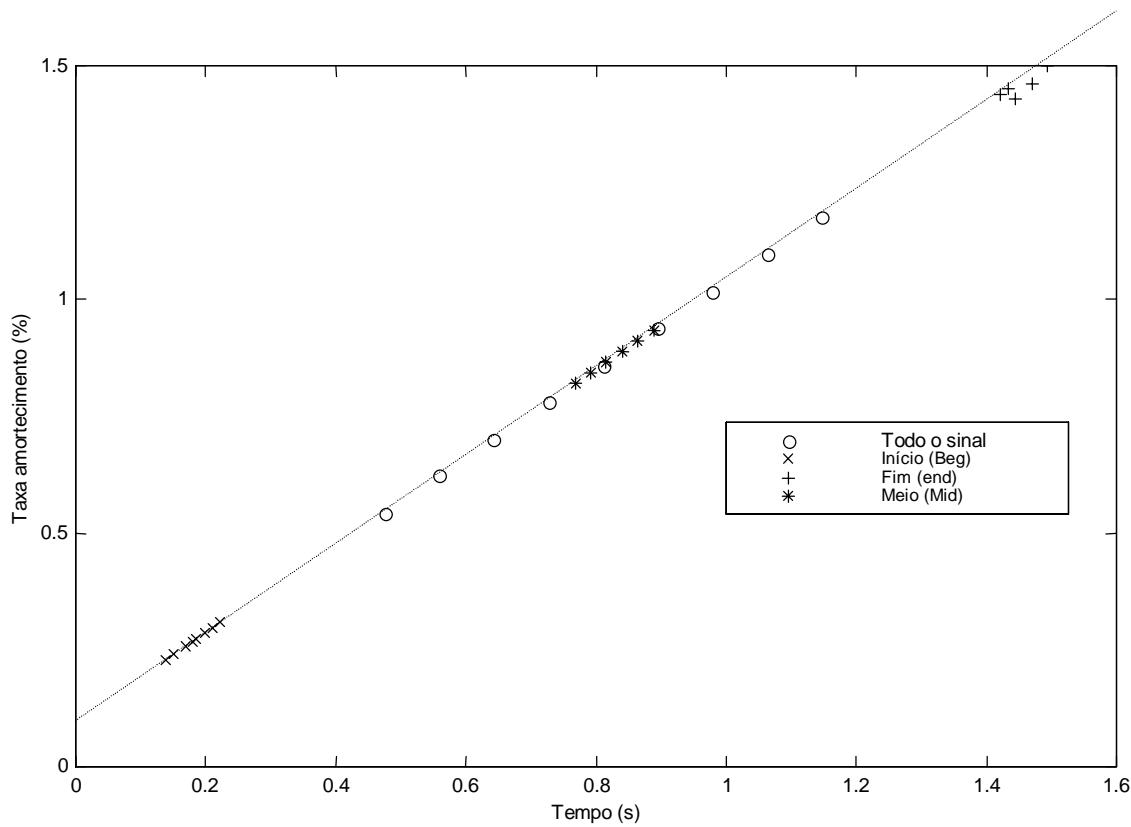
De forma a perceber-se a não-lineariedade desta última frequência (80Hz), é possível construir um segundo tipo de gráfico mostrando a taxa de amortecimento como função do tempo. Pode-se fazer isto dividindo-se a curva da Figura 42 em pequenos trechos e então calculando-se sua inclinação – que vem a ser a taxa de amortecimento por definição – através de uma regressão linear por mínimos quadrados.

O resultado deste procedimento pode ser visto na Figura 43. Adicionalmente está plotada na figura uma reta que representa o resultado esperado teoricamente, ou seja, a reta  $\xi_j(t) = 0.1\% + 0.95\% \cdot t$  correspondente à frequência de 80Hz conforme definido em (75). Percebe-se que os resultados obtidos estão muito bem relacionados aos resultados teóricos esperados, o que demonstra a utilidade deste tipo de gráfico, denominado por ora de *gráfico de (não-)lineariedade*, por possibilitar um estudo mais aprofundado de casos onde o amortecimento é não-linear.

Deve-se atentar para o fato de que os bons resultados mostrados na Figura 42 foram obtidos apesar de a amplitude do primeiro modo ter um valor da ordem de 0,05% da amplitude do quarto modo. Esta é uma séria limitação para outros métodos. Desde que o modo tenha amplitude suficiente para ser diferenciado do ruído de fundo, é possível obter-se bons resultados, independente da amplitude dos demais modos.

Este mesmo exemplo foi testado com um método clássico, o *Least Square Complex Exponential LSCE* (ALLEMANG, BROWN, 1987, ARRUDA, BERTOLINI, 1994) mas os resultados mostraram que este método não foi capaz de identificar corretamente algumas frequências e tampouco as taxas de amortecimento. Sua principal dificuldade, como um método de minimização de erro, é relativo à dificuldade em perceber modos

de baixa intensidade ou não significativos por todo o intervalo analisado pois estes são eclipsados pelas variações dos modos mais significativos. Este método também apresentou dificuldade em identificar modos cujas frequências estejam muito próximas no espectro.



**Figura 43** Gráfico de lineariedade calculado para o modo 4 (80Hz) mostrando a dependência do amortecimento com o tempo

### III.5 Testes experimentais com pórtico

Para comprovar a aplicabilidade do método, foi realizada uma série de quatro ensaios experimentais, indicados pelos números 0 a 3, num modelo experimental constituído de material metálico em forma de pórtico, ilustrado na Figura 44.

Esta estrutura é na verdade um modelo reduzido em escala 1/3 e foi utilizado primeiramente como parte de uma tese de mestrado da COPPE (DUARTE, 1990). Na construção deste modelo reduzido foi necessário acrescentar-se massas nos nós para que a semelhança física com a estrutura original fosse preservada. Essas massas, no entanto, foram incorporadas na presente análise ao peso próprio dos membros cujo valor resultante é indicado Figura 44.

A estrutura foi instrumentada no topo com quatro acelerômetros, denominados AC1Z, AC2X, AC3X e AC4Z, cujos sentidos positivos também estão indicados na Figura 44. Em cada teste a estrutura foi excitada por meio de um impacto lateral na direção X (testes 0X e 1X e 4T) ou na direção Z (testes 2Z e 3Z), obtendo-se um total de vinte sinais, quatro para cada acelerômetro. Para auxiliar o entendimento dos resultados experimentais foi realizada uma análise utilizando-se o software ANSYS, cujos resultados são mostrados na Figura 45 e Figura 46. No intervalo escolhido (0 a 100Hz) foram detectados seis modos, notados por X1, Z1, T1, X2, Z2 e T2.

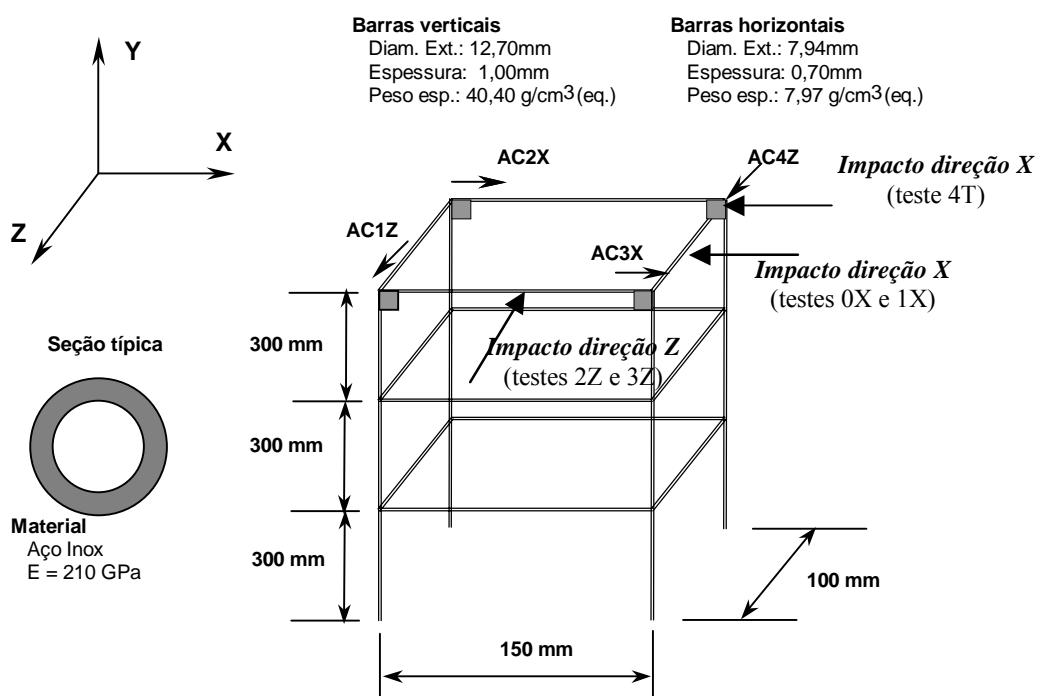
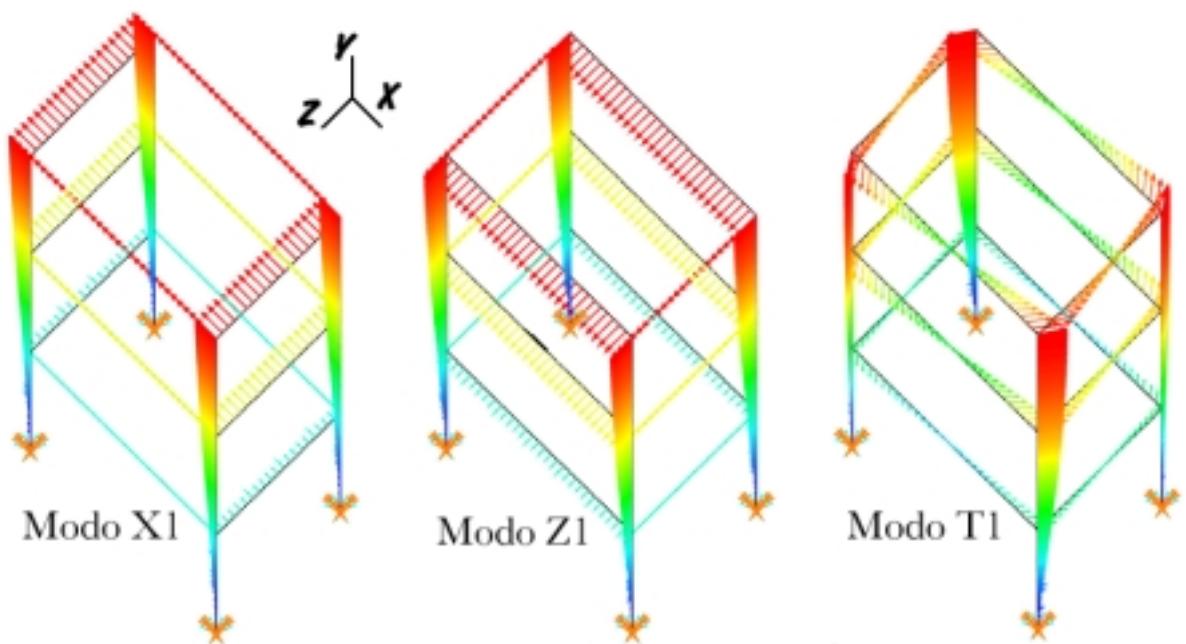
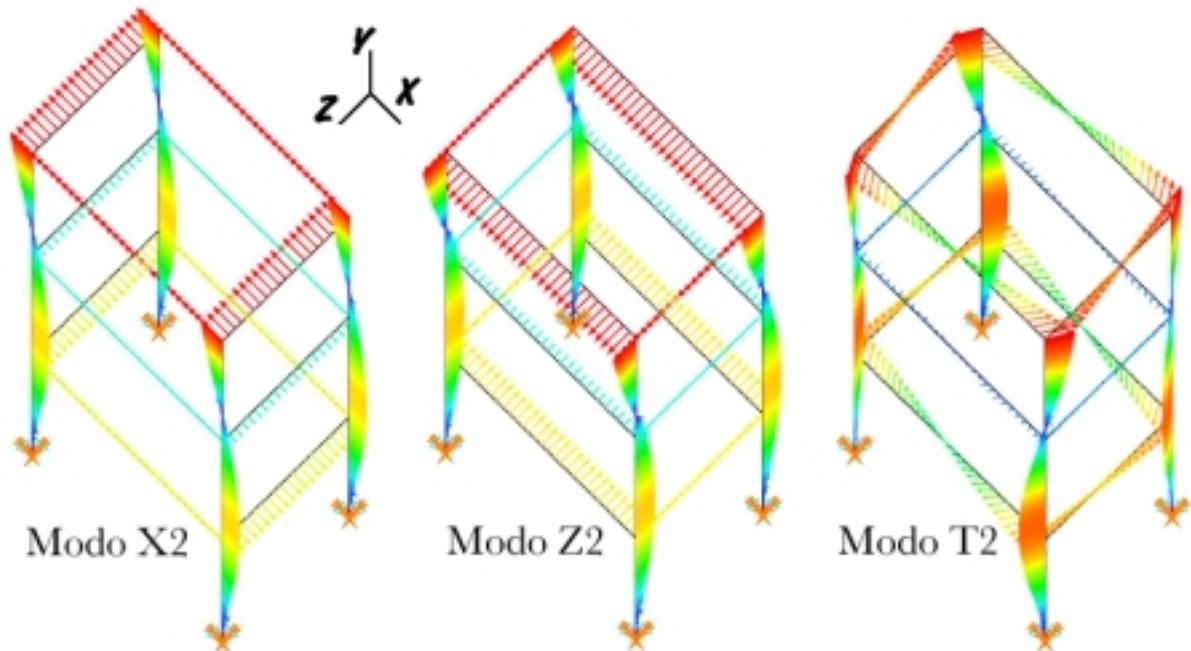


Figura 44 Desenho esquemático do modelo ensaiado



**Figura 45** Três primeiros modos X1 (flexão no plano X-Y), Z1 (flexão no plano Y-Z) e T1 (torção em torno do eixo Y)



**Figura 46** Três últimos modos X2 (flexão no plano X-Y), Z2 (flexão no plano Y-Z) e T2 (torção em torno do eixo Y)

Os modos X1 e X2 referem-se à flexão no plano X-Y. Os modos Z1 e Z2, flexão no plano Y-Z. Os modos T1 e T2 são modos de torção em torno do eixo Y (vertical). A Tabela 2 mostra um resumo das frequências naturais obtidas da análise por elementos finitos e dos experimentos. Como a análise por elementos finitos teve o único objetivo de aumentar a compreensão dos modos de vibração da estrutura, nenhum esforço foi

realizado no sentido de se ajustar as frequências naturais do modelo numérico aos resultados experimentais. No entanto, as frequências, numéricas e experimentais, mostram-se suficientemente bem relacionadas como pode-se perceber na Tabela 2.

**Tabela 2 Resumo dos modos detectados teórica e experimentalmente**

Modo	ID	Frequência teórica (MEF)	Frequência experimental	Melhor captado pelos acelerômetros	Melhor(es) teste(s)
1º modo flexão na direção X	X1	14.8 Hz	15.3 Hz	AC2X e AC3X	0X e 1X
1º modo flexão na direção Z	Z1	16.3 Hz	16.4 Hz	AC1Z e AC4Z	2Z e 3Z
1º modo torção	T1	29.9 Hz	25.7 Hz	Todos	0X e 3Z
2º modo flexão na direção X	X2	55.5 Hz	53.9 Hz	AC2X e AC3X	0X e 1X
2º modo flexão na direção Z	Z2	59.1 Hz	56.4 Hz	AC1Z e AC4Z	2Z e 3Z
2º modo torção	T2	88.2 Hz	80.2 Hz	Todos	4T

### Metodologia de cálculo e apresentação das taxas de amortecimento

Os sinais obtidos experimentalmente foram submetidos aos mesmos procedimentos de cálculo utilizados para a simulação anterior. Evidentemente uma grande parte da informação foi descartada como por exemplo os casos onde o modo estudado é de flexão na direção X e o acelerômetro está alinhado na direção transversal Z. Apesar de, surpreendentemente, em vários desses casos adversos ser ainda possível detectar os parâmetros modais com boa precisão, a massa de dados a serem apresentados se tornaria muito grande para ser apresentada com concisão.

Os resultados individuais constam de dois tipos de gráficos: o gráfico de amortecimento e o gráfico de lineariedade. O gráfico de amortecimento mostra o desenvolvimento da função de amplitude normalizada  $\hat{v}(t)$  em função do tempo. O gráfico de (não)lineariedade mostra o desenvolvimento da taxa de amortecimento  $\xi(t)$  como função do tempo.

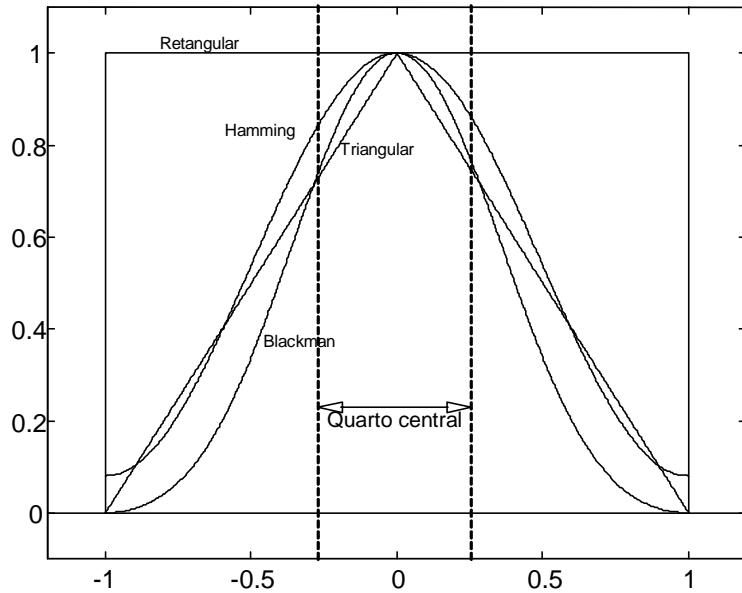
Poderia-se, de maneira a diminuir a quantidade de gráficos apresentados, mostrar apenas os gráficos de lineariedade que já mostram explicitamente o resultado final dos cálculos, ou seja, a taxa da amortecimento. No entanto a taxa de amortecimento é definida como a derivada da amplitude normalizada conforme (40) e esta derivada é calculada através de uma regressão linear. Se o trecho submetido à regressão for muito

curto os resultados poderão mostrar-se bastante oscilatórios, o que vem contra o comportamento físico esperado de um amortecimento constante ou, no pior dos casos, levemente não-linear. Portanto é interessante que se escolha um trecho o maior possível mas não tão grande de forma que ainda se possa detectar uma variação da taxa de amortecimento, se houver.

Em resumo o que se quer deixar claro é que os resultados finais – as taxas de amortecimento – podem ter uma certa variabilidade de acordo com a escolha do tamanho dos trechos submetidos à regressão *nos casos não-lineares*. Isto acontece, por exemplo, com os modos X1, T1 e T2. Nos casos onde o amortecimento mostra-se claramente linear este problema não existe pois toma-se todo o trecho do sinal para a regressão, com resultados únicos. Desta forma optou-se por apresentar, para os casos lineares, somente os gráficos de amortecimento, ou seja, os que mostram a amplitude normalizada  $\hat{v}(t)$  como função do tempo. O amortecimento obtido por regressão é, neste caso, mostrado diretamente no gráfico ao lado da respectiva curva. Nos casos não-lineares são mostrados os dois gráficos, o de amortecimento e o de lineariedade, onde pode-se acompanhar as variações da taxa de amortecimento  $\xi(t)$  com o tempo. Para estes gráficos foi usada uma largura do trecho da regressão igual a 30% da largura do sinal total.

Como este processo depende das transformadas tempo-frequência e estas não podem ser obtidas com precisão indefinida e arbitrariamente grande nos dois domínios, se quiser-se uma maior precisão em frequência precisa-se abrir mão de alguma precisão no tempo. Isto foi efetivamente feito nesta análise: para obter alta precisão em frequência e assim garantir uma melhor estimativa dos parâmetros modais, foi adotada uma janela relativamente larga (10%) o que, para o tempo total de 30s, fornece uma janela de largura total 3s. A técnica de modulação contorna este problema e torna possível a utilização de janelas mais curtas pela translação do sinal no eixo da frequência.

Isto seria totalmente verdade para a janela retangular, por exemplo. Para as outras janelas, contudo, esta largura não é a precisão total da transformada visto que a maior parte da energia destas janelas concentra-se em seu lobo central. Isto pode ser compreendido melhor através da Figura 47 onde são mostradas algumas janelas comuns neste tipo de análise.



**Figura 47 Algumas janelas comumente usadas com o espectrograma**

Percebe-se que a janela retangular é a única a ocupar efetivamente toda a extensão do intervalo. As demais ocupam, efetivamente, apenas uma parte da região central. A quantidade de energia retida no quarto central do intervalo (-0,25 a +0,25) para algumas janelas é mostrado na Tabela 3.

**Tabela 3 Percentagem de energia retida no quarto central do intervalo para algumas janelas mais utilizadas com o espectrograma**

Janela	Percentagem de energia retida no quarto central
Retangular	25,0%
Hamming	58,3%
Blackman	71,0%
Triangular	58,2%

Da Tabela 3 apreende-se que, afora a retangular, todas as demais janelas concentram mais de 50% de sua energia no quarto central. Isto é especialmente verdade para a janela Blackman, que efetivamente foi usada na presente análise. Pode-se adotar portanto, empiricamente, para as janelas – exceto a retangular – um quarto do valor nominal como valor a ser utilizado para o intervalo de incerteza.

O intervalo de incerteza corresponde ao intervalo de tempo que a transformada necessita para responder plenamente a um sinal subitamente inserido. Imagine-se um sinal constante de valor zero ao qual é inserida subitamente uma senóide no instante 0s. Se a janela utilizada for, por exemplo, a blackman de largura nominal 1s, é (empiricamente) esperado que a transformada parta do valor zero no instante 0s até um valor constante

no instante 0.25s que deverá continuar constante enquanto a senóide existir no sinal. Dentro deste intervalo a transformada apresentará uma curva suave como uma spline, isto é, tipicamente concordando com a primeira derivada da função nos extremos. A este intervalo de transição chama-se *intervalo de incerteza da transformada* por estar diretamente relacionado ao *princípio da incerteza*. Dentro deste intervalo, se houverem transições bruscas do espectro em frequência como o caso do exemplo, os valores não serão confiáveis.

Deve-se notar que o valor empírico adotado de  $\frac{1}{4}$  do valor nominal da largura da janela refere-se somente ao spectrograma. Sendo o spectrograma a distribuição tempo-frequência com menor qualidade (no sentido do princípio da incerteza), este valor torna-se um limite superior. Outras transformadas como a SAF (BUCHER, 1998), por exemplo, fornecem intervalos bastante menores. Não fosse o preço computacional tão alto certamente estas distribuições poderiam tomar o lugar do spectrograma como transformada padrão. Para a atual análise, quando um número relativamente grande de transformadas precisam ser calculadas, o spectrograma ainda é a escolha ideal.

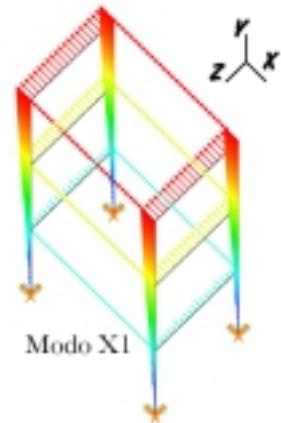
Dado que a janela utilizada correntemente nesta análise tem 3s de largura nominal (10% de 30s), o intervalo de confiabilidade é de  $\frac{1}{4}$  3s, ou 0.75s. Por este motivo descartam-se sempre os primeiros 0.75s do sinal que, pode-se notar nos gráficos, é geralmente uma curva. Deve-se atentar para o fato de que os sinais foram transladados de forma que iniciassem exatamente no instante 0s.

Desta forma, para fazer a regressão linear somente no trecho (supostamente) linear do sinal deve-se descartar os primeiros 0,75s a partir do instante do impacto que pode ser obtido sem dificuldade do gráfico da força imposta também gravado mas não mostrado nas figuras.

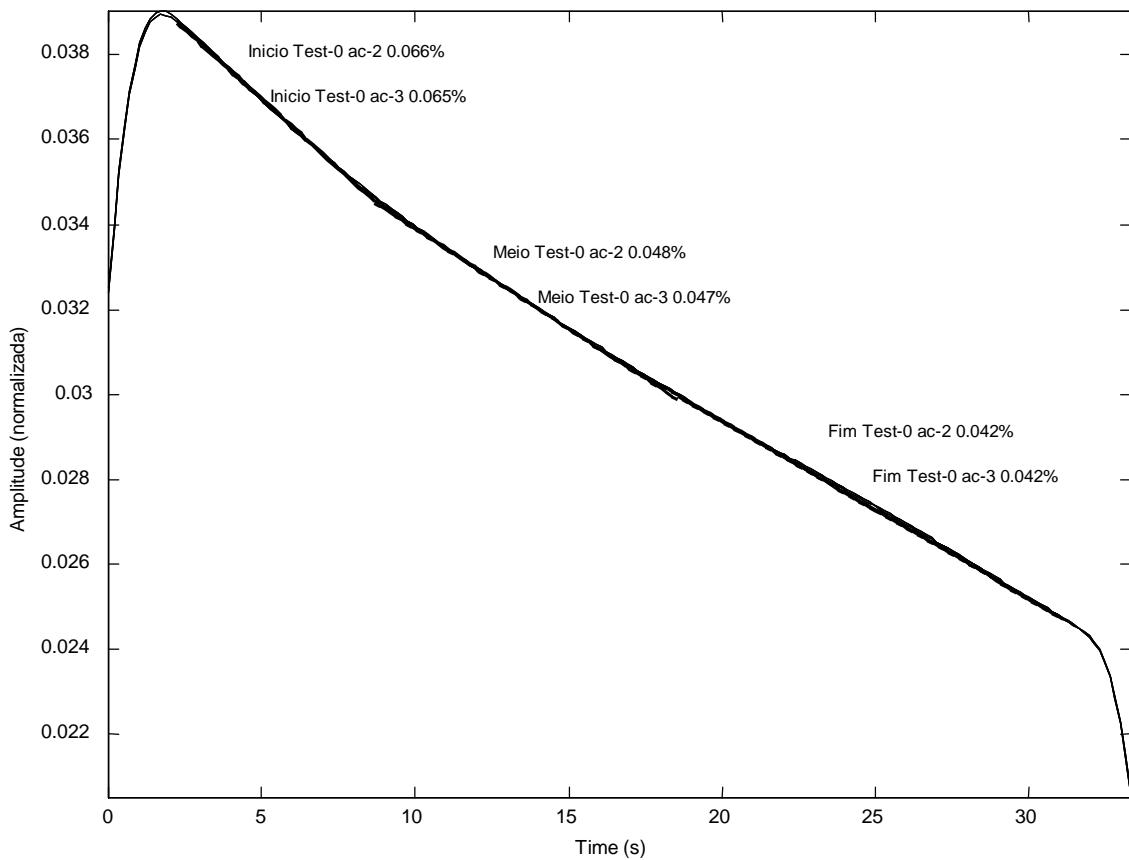
Nos itens seguintes passa-se a apresentar em detalhes os resultados das análises de cada modo.

### a) Modo X1, freqüência 15,3Hz

O primeiro modo de flexão na direção X é o que apresenta a menor taxa de amortecimento. Devido à proximidade em frequência com o modo Z1 (primeiro modo de flexão na direção Z), ocorre frequentemente acoplamento entre os modos, o que prejudica a qualidade de qualquer análise experimental.

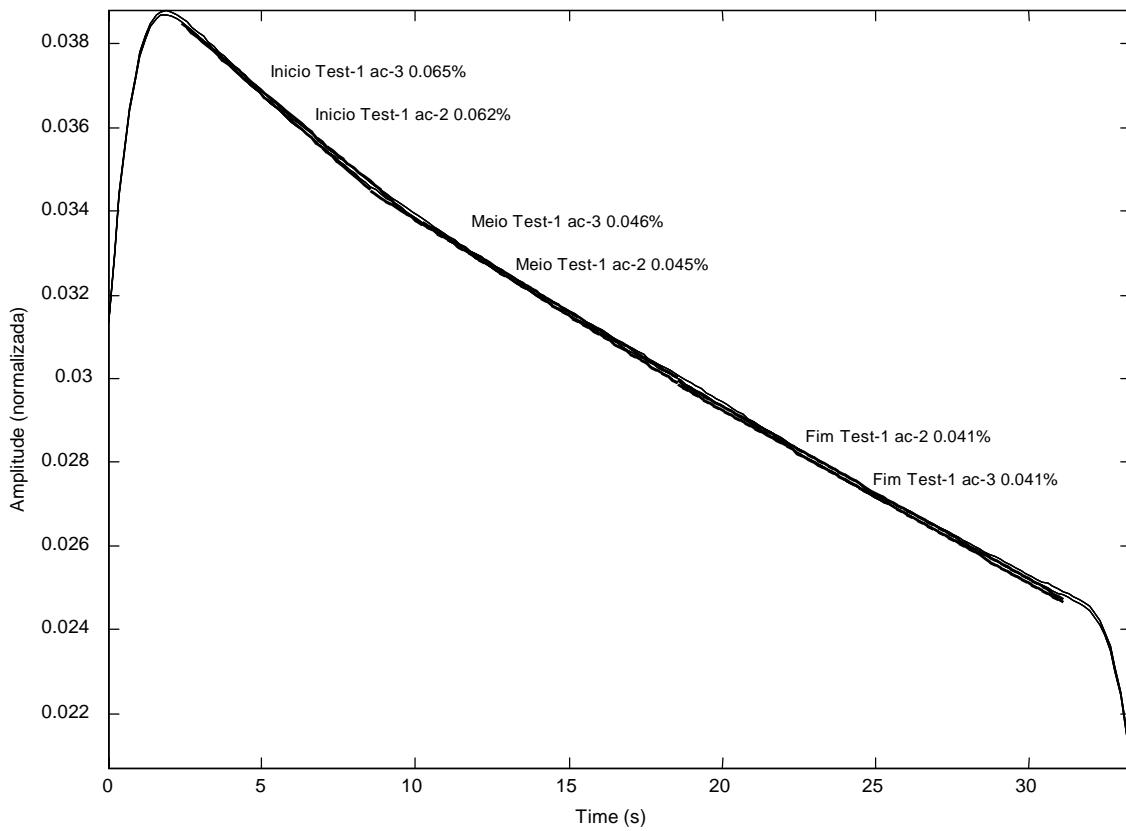


Os melhores testes para extração de parâmetros deste modo são os testes 0X e 1X, que são impactos na direção X. Os melhores acelerômetros são os AC2X e AC3X, alinhados na direção X.

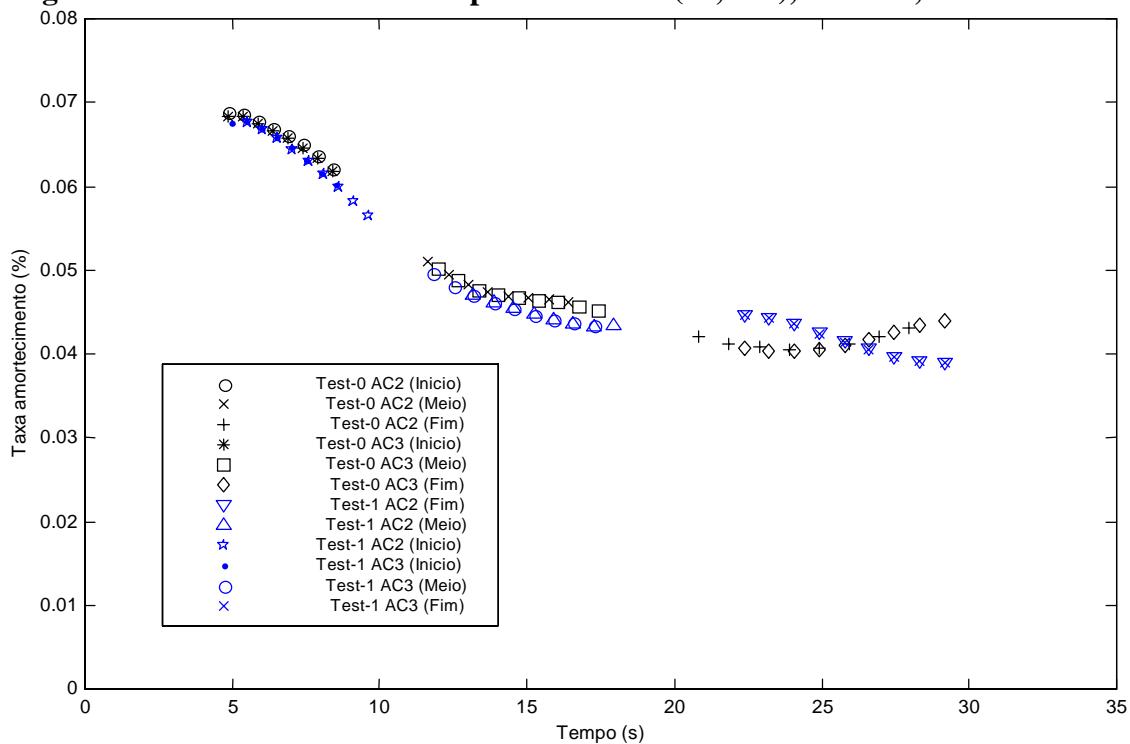


**Figura 48 Curva amortecimento para modo X1 (15,3Hz), teste 0X, AC2X e AC3X**

Este modo apresenta um forte comportamento não-linear, conforme pode-se observar na Figura 48 (resultados para o teste 0X) e na Figura 49 (resultados para o teste 1X). De forma a caracterizar bem este comportamento visualmente, optou-se por dividir as curvas de amortecimento em três trechos aproximadamente lineares denominados nos gráficos por “Início”, “Meio” e “Fim” cujos amortecimentos obtidos são, respectivamente, 0.066% (0 a 12s), 0.048% (12 a 20s) e 0.042% (20 a 34s).



**Figura 49 Curva amortecimento para modo X1 (15,3Hz), teste 1X, AC2X e AC3X**

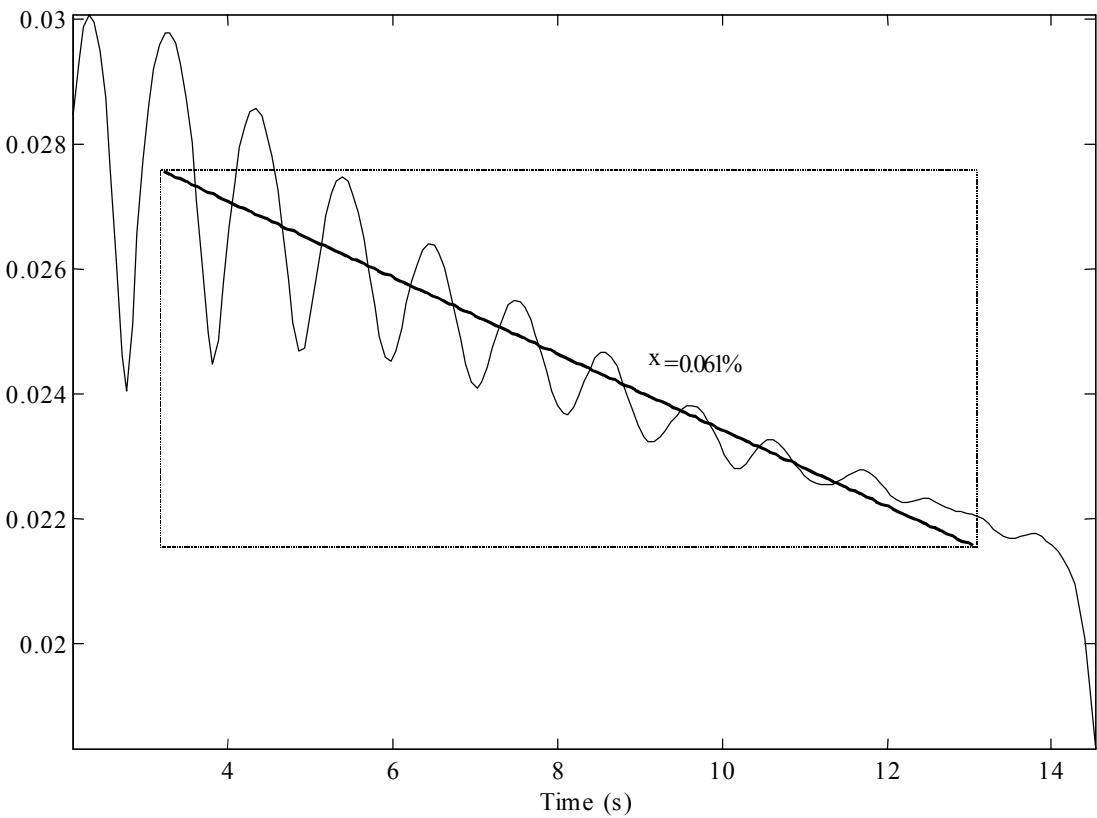


**Figura 50 Gráfico de lineariedade calculado para o modo X1 (15.3Hz)**

Um ponto importante a se notar é a grande correlação entre os resultados experimentais para este modo. Isto pode ser verificado pela sobreposição das curvas a pontos de não poder-se distingui-las individualmente. Isto também se reflete na proximidade dos

valores de amortecimento obtidos para cada trecho, que assemelham-se até a terceira casa decimal.

Do gráfico de lineariedade da Figura 50 pode-se notar mais claramente que a taxa de amortecimento inicia-se por volta de 0.07% e rapidamente decai para um valor próximo de 0.045% em 15s e então tem um decaimento menos acentuado até 0.042% em 30s. Este comportamento parece indicar uma dependência do amortecimento com a amplitude do movimento. Nota-se novamente a grande proximidade dos resultados obtidos tanto entre testes como, principalmente, entre as taxas obtidas de sinais de diferentes acelerômetros de um mesmo teste.



**Figura 51 Curva amortecimento para modo X1 (15,3Hz), teste 0X, AC1Z**

Para demonstrar a robustez do método, mostra-se na Figura 51 a curva de amortecimento obtida utilizando-se um acelerômetro na pior posição possível para obter-se resultados do modo X1, ou seja, na direção transversal Z. Teoricamente nenhum resultado deveria ser obtido pois a forma modal do modo X1 na direção Z é nula e, portanto, um acelerômetro nesta direção não deveria captar qualquer sinal relativo a este modo. No entanto, pelas imperfeições intrínsecas na montagem do acelerômetro bem como de sua capacidade de medição (sensibilidade a acelerações transversais), uma pequena parcela da energia do modo X1 é captado pelo acelerômetro

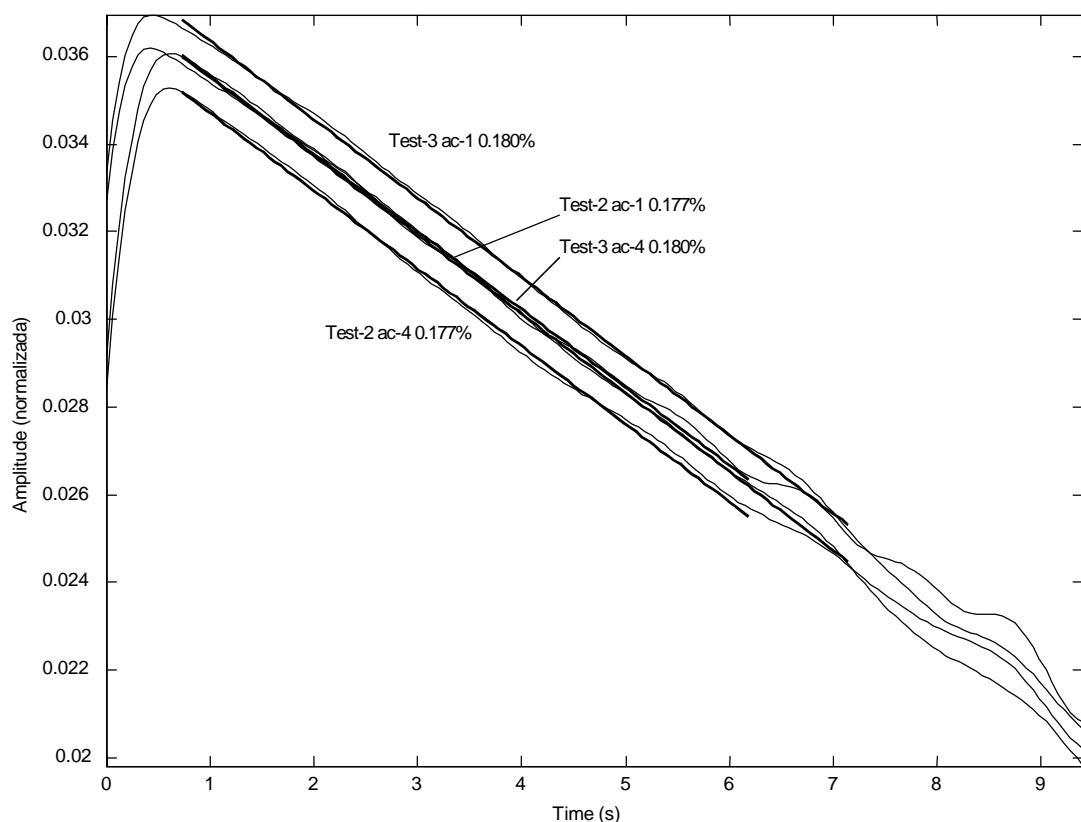
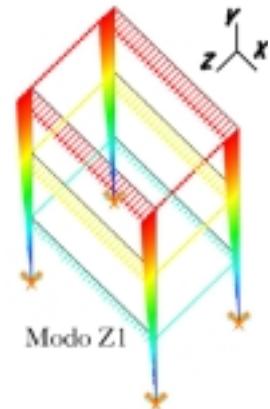
positionado transversalmente. Como agravante, a presença de um modo bastante próximo (modo Z1, distante menos de 1Hz) cuja direção de vibração está alinhada com o acelerômetro faz supor que o método possa confundir-se, não tendo precisão suficiente para detectar aquela pequena parcela de energia.

Pode-se notar na Figura 51 um aspecto oscilatório ao invés da forma linear - ou levemente curva - esperada. Isto provavelmente se deve à proximidade deste modo com o modo Z1. No entanto, selecionando-se um trecho longo para regressão linear, observa-se que o amortecimento médio obtido (0.061%) aproxima-se bastante dos resultados obtidos nas melhores condições com os demais sensores (0.04 a 0.07%). Desta forma verifica-se que o método é bastante robusto no sentido de ignorar influências de modos próximos, mesmo em condições extremamente desfavoráveis, ainda assim obtendo resultados satisfatórios.

## b) Modo Z1, frequência 16.4 Hz

Este é o primeiro modo de flexão na direção Z. Os melhores testes para observá-lo são os testes 2Z e 3Z e os melhores acelerômetros, AC1Z e AC4Z.

Este modo apresentou amortecimento totalmente linear de valor invariavelmente dentro da faixa ( $0.180 \pm 0.005\%$ ) tanto para os testes 2Z como 3Z. Mesmo nos testes 0X e 1X nos quais os impactos foram aplicados na direção transversal, ainda assim pode-se obter estes valores, embora com uma dispersão maior.

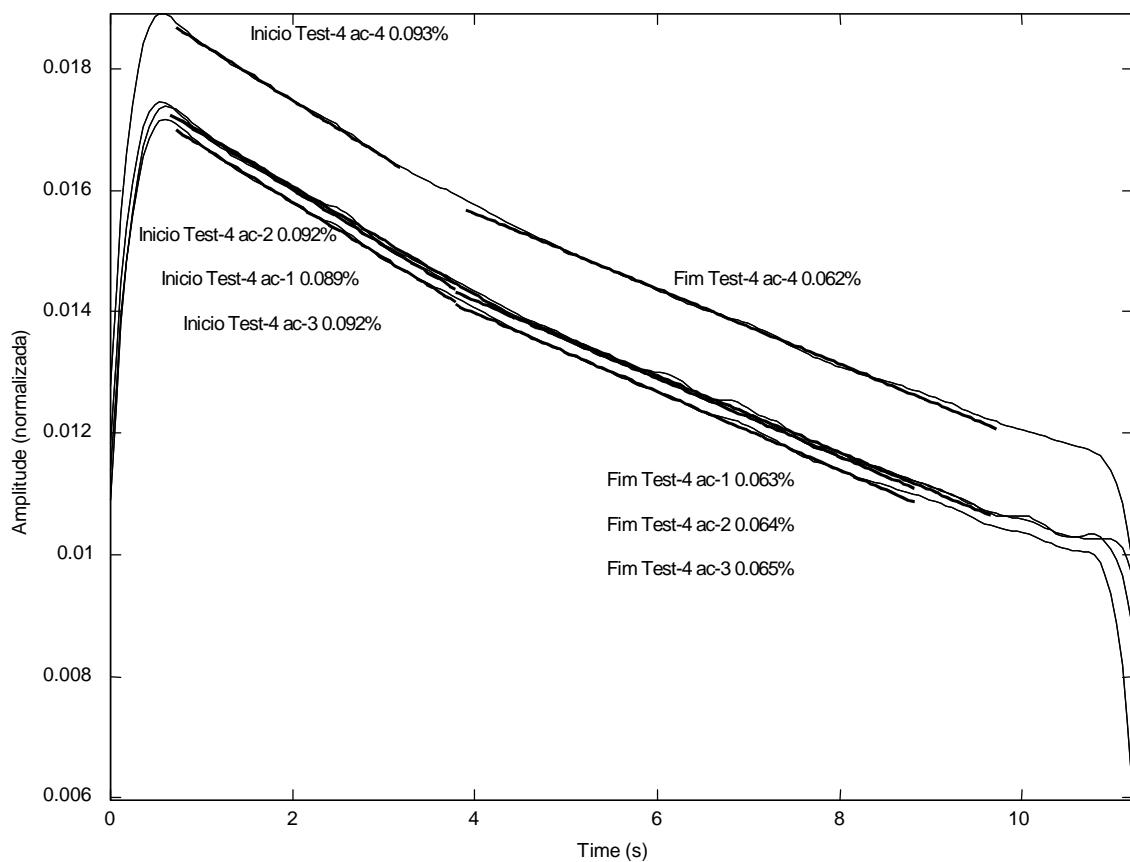
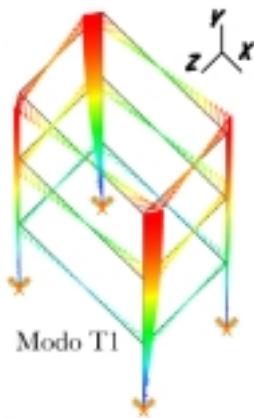


**Figura 52 Curvas amortecimento para modo Z1 (16,3Hz), teste 2Z e 3Z, AC1Z e AC4Z**

O gráfico de amortecimento da Figura 52 mostra claramente que as curvas de amplitude permanecem retas e paralelas, indicando um amortecimento constante até o instante 7s onde uma oscilação mais constante indica baixas amplitudes de movimento e, portanto, maior contaminação pelo ruído.

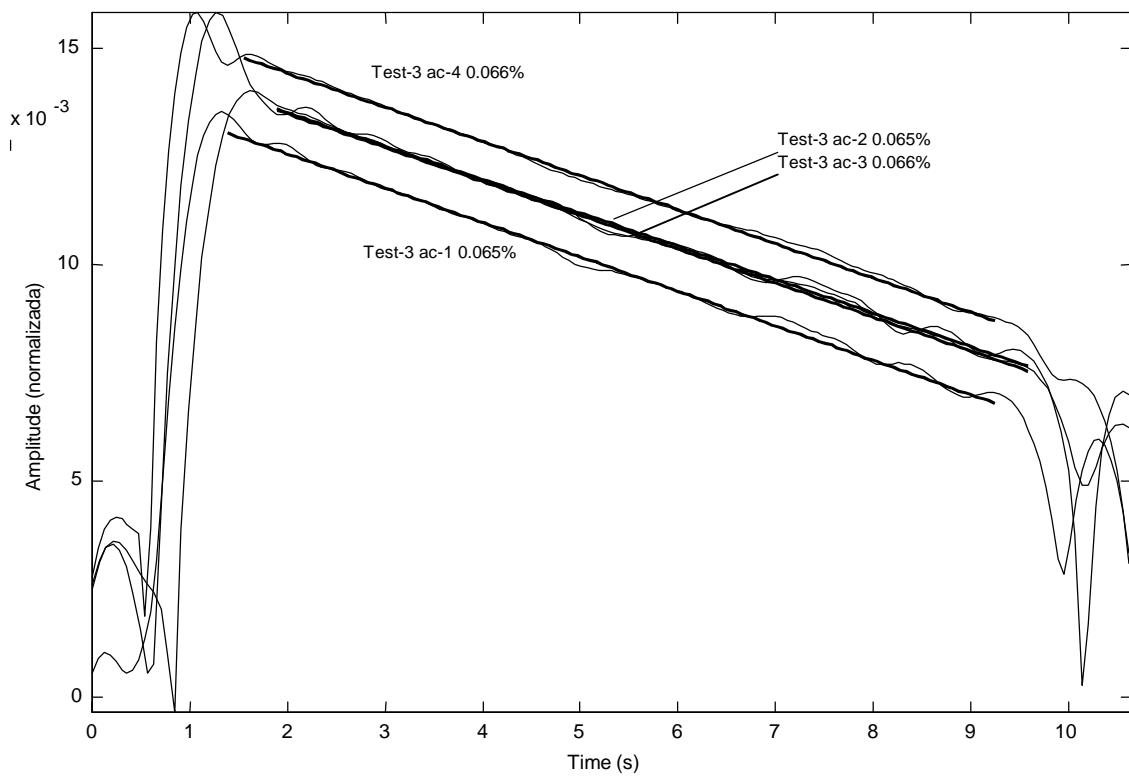
### c) Modo T1, frequência 25.7 Hz

Este é o primeiro modo de torção em torno do eixo Y. O melhor teste para se observar este modo é o teste 4T, na qual foi aplicado um impacto não-centrado na direção X, que teve o objetivo de aplicar um momento torsor na estrutura, aumentando assim a excitação deste modo. No entanto bons resultados foram igualmente obtidos por outros testes. Todos os acelerômetros podem ser usados para analisar este modo desde que o modo induz movimentos nas duas direções X e Z.

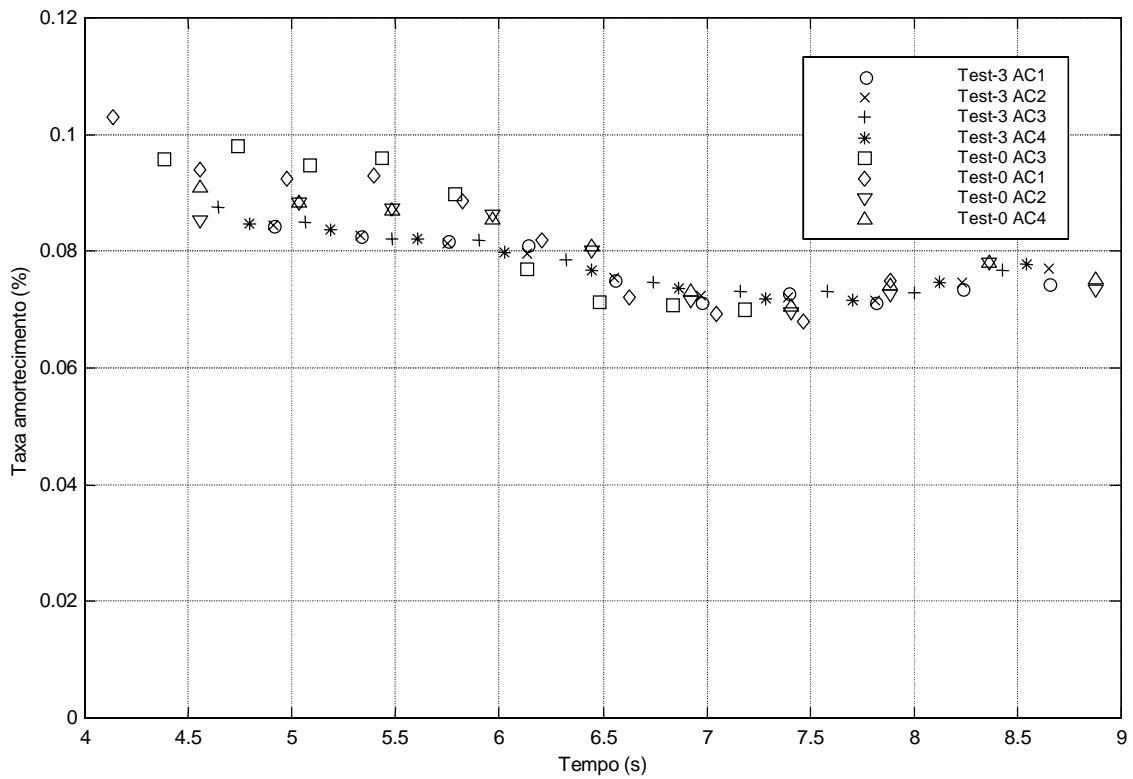


**Figura 53 Curva amortecimento para modo T1, teste 4T, AC1Z e AC4Z**

Foram obtidos comportamentos aparentemente distintos deste modo quando compararam-se os resultados obtidos com o teste 4T (impacto não centrado na direção X) e teste 3Z (impacto centrado na direção Z). No caso do teste 4T foi observado um amortecimento claramente não-linear (Figura 53) enquanto para o teste 3Z foi observado amortecimento linear (Figura 54).



**Figura 54 Curva de amortecimento para o modo T1, teste 3Z, AC2X e AC3X**



**Figura 55 Gráfico de lineariedade do modo T1 (25.7 Hz)**

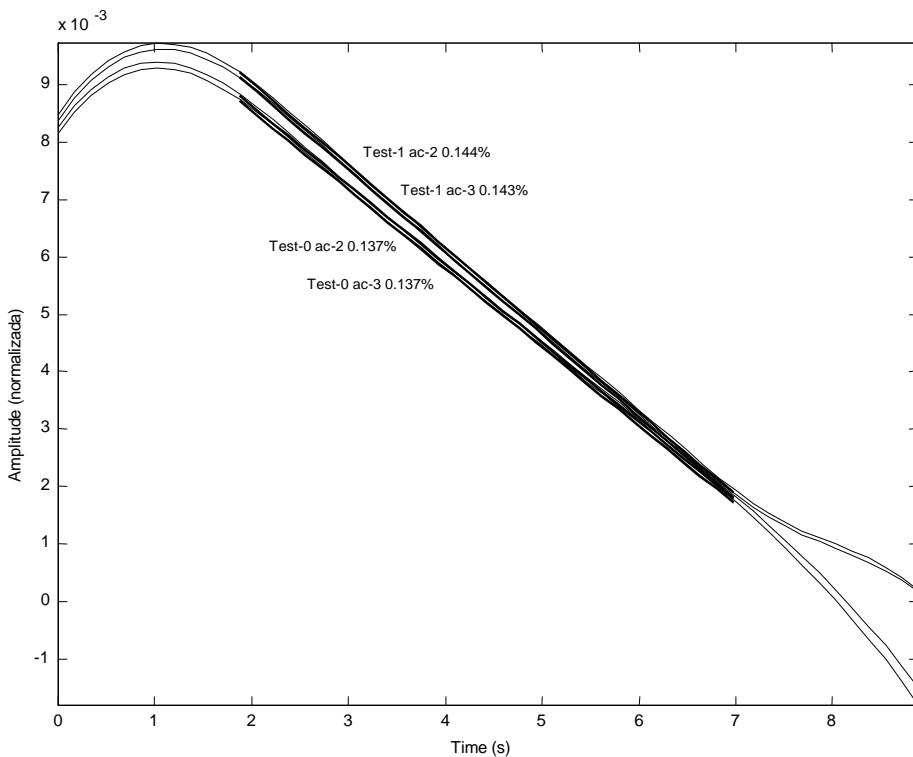
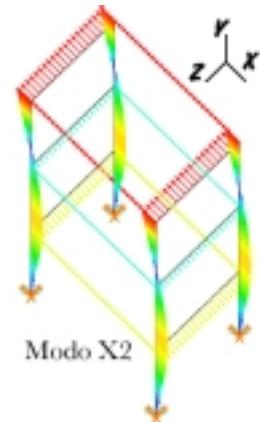
Esta distinção pode ser explicada se for observado que a amplitude do movimento (normalizada) do teste 4T inicia maior, por volta de 0.018, decaindo até 0.015 enquanto a amplitude do teste 3Z inicia menor, por volta de 0.015 e decai até 0.005. Isto indica que o teste 4T excitou com mais energia este modo por causa do impacto não centralizado, impelindo o modo para dentro de uma região de altas amplitudes onde o comportamento do amortecimento é não-linear.

Esta suposição é corroborada pela observação do gráfico de lineariedade (Figura 55) onde pode-se notar que o amortecimento, inicialmente igual a 0.100% cai para um valor por volta de 0.070% em 7s e então estabiliza-se neste valor que também é aproximadamente obtido do teste 3Z mostrado na Figura 54.

#### d) Modo X2, frequência 53,4Hz

Este é o segundo modo de flexão na direção X. Os melhores teste para se observar este modo são os teste 0X e 1X. Os melhores acelerômetros, AC2X e AC3X, alinhados na direção X.

Este modo apresenta comportamento perfeitamente linear como pode-se observar na Figura 56. Possivelmente este modo não foi levado a regiões de não-lineariedades como o modo X1 por ser um modo de maior rigidez dinâmica, requerendo portanto mais energia para realizar deslocamentos de mesma ordem de grandeza.



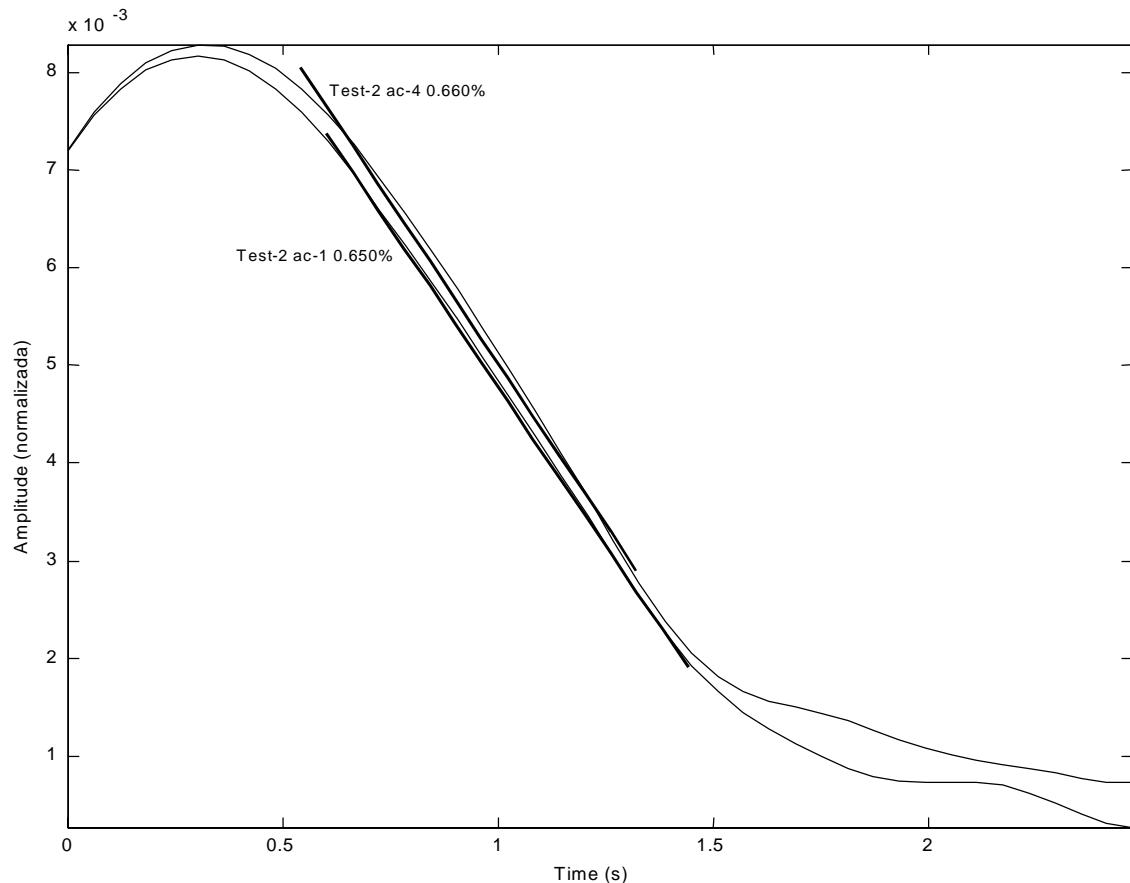
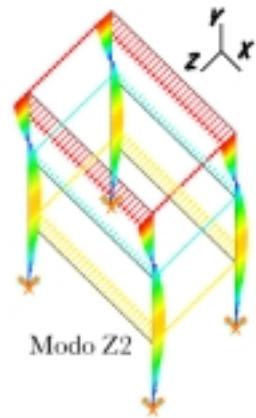
**Figura 56 Curva amortecimento, modo X2 (53,4Hz) testes 0X e 1X, AC2X e AC3X**

Observa-se que os resultados apresentados pelos acelerômetros AC2X e AC3X indicam uma correlação muito forte – até a terceira casa decimal – em cada teste, isto é, 0.144% para o teste 0X e 0.137% para o teste 1X. Estes resultados atestam a precisão do método para casos puramente lineares pois pode-se observar que uma leve alteração da inclinação das retas, 0.007% de diferença entre os testes 0X e 1X, pode ser facilmente notada através da simples inspeção visual da curva de amortecimento. Esta pequena diferença entre os resultados dos testes talvez possa ser devida a pequenas alterações de rigidez da estrutura durante os intervalos entre os testes, fato que certamente passaria despercebido não fosse o gráfico acima.

### e) Modo Z2, frequência 56,4Hz

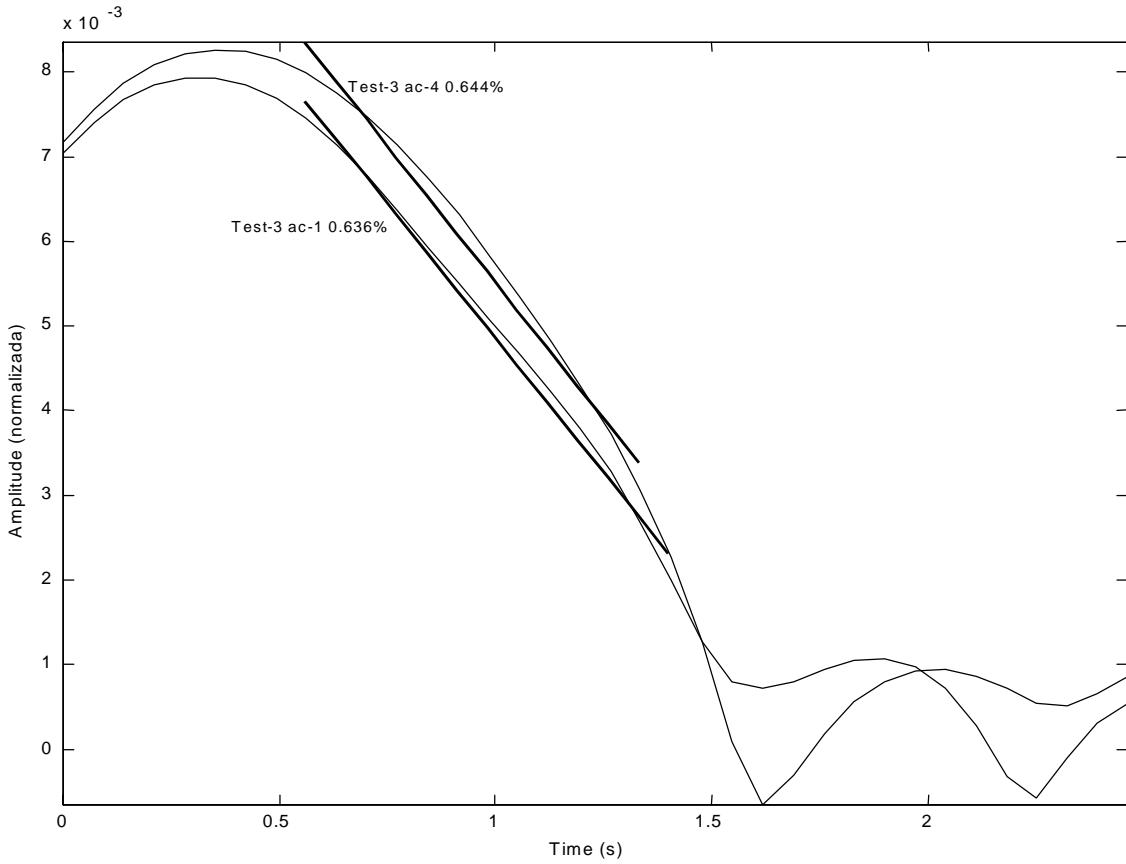
Este é o segundo modo de flexão na direção Z. Os melhores testes para observá-lo são os testes 2Z e 3Z e os melhores acelerômetros, AC1Z e AC4Z.

Este modo apresenta uma alta taxa de amortecimento (0.650%) mais elevada que os seis primeiros modos de vibração desta estrutura (até 0.180%).



**Figura 57 Curvas amortecimento para modo Z2 (56,4Hz), teste 2Z, AC1Z e AC4Z**

Este possivelmente é o modo mais difícil de ser captado por outros métodos pois seu período útil (até sua amplitude ser igual ao do ruído) é muito curto (0 a 1s) se comparado aos outros modos (até 30s). Como o presente método tem a capacidade de separar os modos em problemas distintos, pode-se plotar a curva de amortecimento somente para o período útil do modo, como mostrado na Figura 57 e Figura 58.

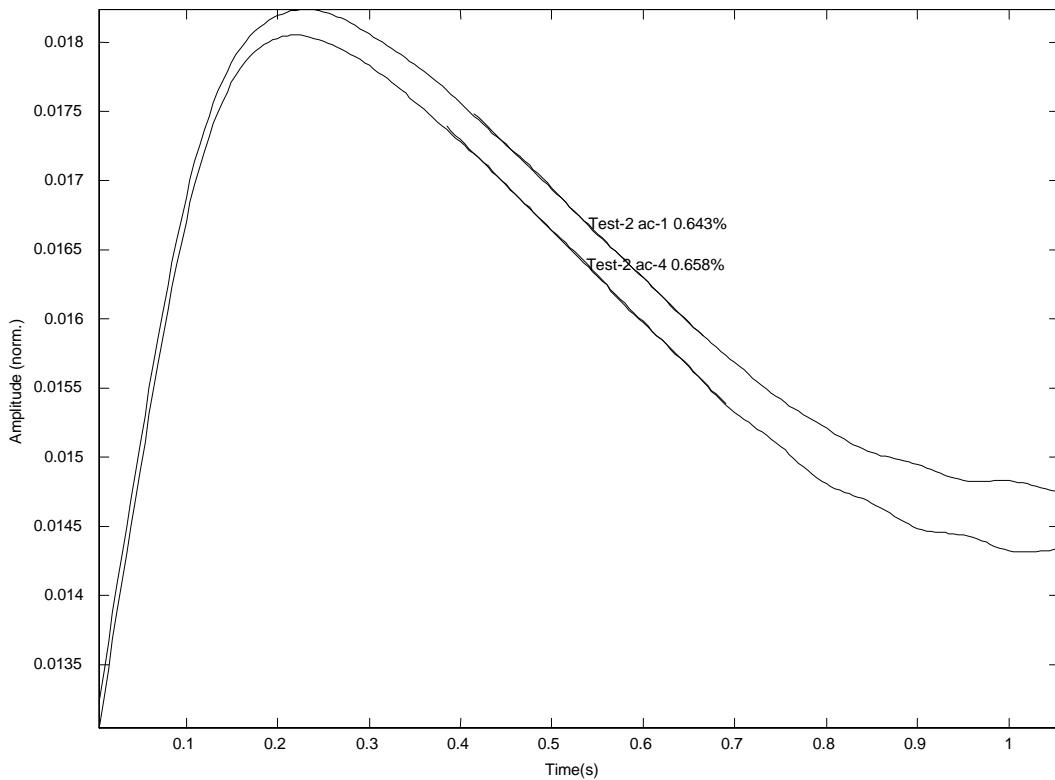


**Figura 58 Curvas amortecimento para modo Z2 (56,4Hz), teste 3Z, AC1Z e AC4Z**

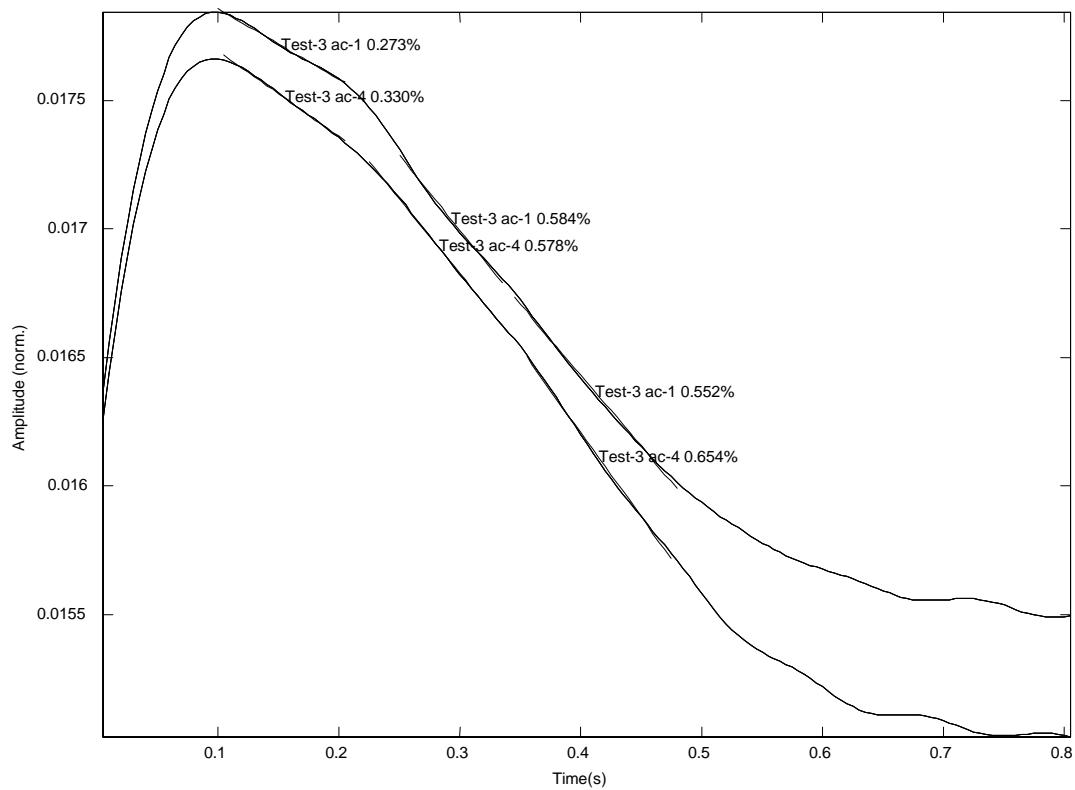
Nota-se pela Figura 58 que o trecho disponível para regressão linear é muito curto para este modo visto que precisa-se descartar os primeiros e últimos 0,75s do sinal (intervalo de incerteza). Uma solução, neste caso, seria aplicar-se excitações em lugares diferentes ou de maneiras diferentes de forma que este modo particular seja excitado com maior amplitude, aumentando-se assim o intervalo confiável para a regressão. A melhor solução neste caso é utilizar-se a técnica da modulação.

Alternativamente, pode-se recorrer a uma transformada mais exigente (SAF por exemplo) para obter-se uma curva de amortecimento mais precisa. A Figura 59 e a Figura 60 mostram as curvas de amortecimento obtidas com o método SAF (Simplified Alias-Free). Nota-se que o teste 2 (Figura 59) é basicamente linear mas o teste 3 (Figura 60) apresenta trechos descontínuos que, na Figura 58, aparecem como uma curva muito suave. A precisão do método SAF é mais percebido quando se nota que o intervalo de tempo da transformada é de apenas 1 segundo enquanto o trecho total da curva de amortecimento obtida pelo espectrograma é de 2,5 segundos. Seu cálculo, no

entanto, é mais caro computacionalmente, o que torna inviável utilizá-lo em todos os casos desta análise.



**Figura 59 Curvas calculadas utilizando o método SAF, teste 2Z, AC1Z e AC4Z**

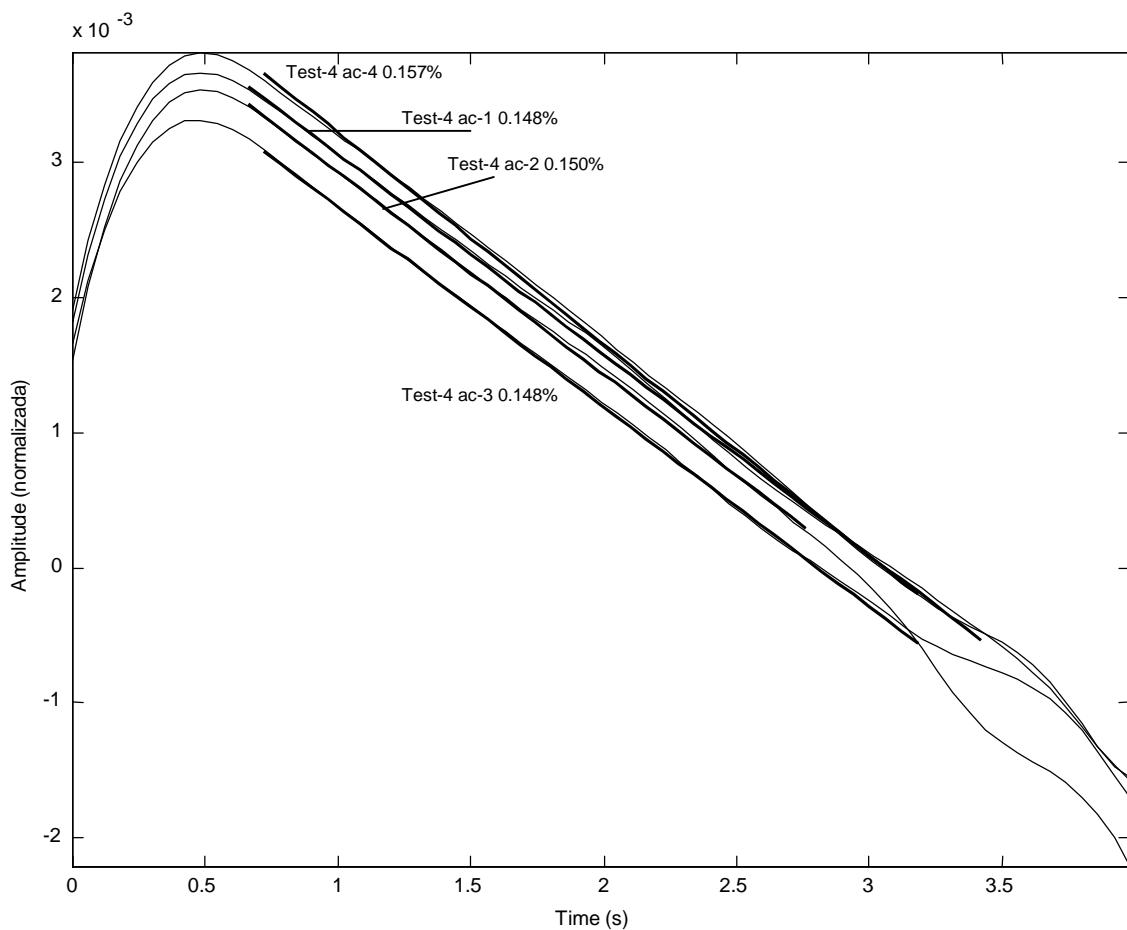
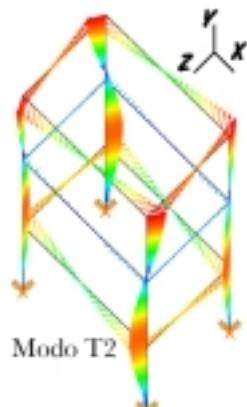


**Figura 60 Curvas calculadas utilizando o método SAF, teste 3Z, AC1Z e AC4Z**

## f) Modo T2, frequência 80,2Hz

Este é o segundo modo de torção em torno do eixo Y. O melhor teste para observá-lo é o teste 4T. Todos os acelerômetros podem ser usados para analisar este modo desde que sua forma modal possui coordenadas não-nulas nas duas direções (X e Z).

Ao contrário do modo anterior (Z2), este modo possui taxa de amortecimento pequena, por volta de 0.150%, o que fornece um trecho linear bastante longo para a regressão, como pode-se notar na Figura 61. Também neste modo pode-se notar a precisão do método, onde o desvio padrão dos resultados não passou de 0,004%.



**Figura 61 Curvas amortecimento para modo T2 (80,2Hz), teste 4T, AC1Z , AC2X, AC3X e AC4Z**

## Resumo das taxas de amortecimento obtidas

De forma a modularizar a apresentação dos dados selecionados, que ainda assim é extensa, resolveu-se agrupar os resultados pelos modos, da menor para a maior frequência, seguindo a itemização corrente. Os resultados finais, resumidos, estão condensados na Tabela 4, que fornece os resultados condensados – taxas de amortecimento – obtidos através do método apresentado neste capítulo.

**Tabela 4 Taxas de amortecimento obtidas pelo método da TFD**

Modo	ID (freq.)	Teste	Acele- rômetro	Taxa média (%)		
1º modo flexão na direção X (apresentou não-lineariedades)	X1 15.3Hz	0X		0-12s	12-20s	20-34s
			AC2X	0.066%	0.048%	0.042%
		1X	AC3X	0.065%	0.047%	0.042%
			AC2X	0.062%	0.045%	0.041%
	Z1 16.4Hz	2Z	AC3X	0.065%	0.046%	0.041%
			AC1Z		0.177%	
		3Z	AC4Z		0.178%	
			AC1Z		0.181%	
			AC4Z		0.180%	
1º modo torção (apresentou não-lineariedades)	T1 25.7Hz	4T		1-4s	4-10s	
			AC1Z	0.089%	0.063%	
			AC2X	0.092%	0.064%	
			AC3X	0.092%	0.065%	
			AC4Z	0.092%	0.065%	
2º modo flexão na direção X	X2 53.9Hz	0X	AC2X		0.137%	
			AC3X		0.139%	
		1X	AC2X		0.144%	
			AC3X		0.143%	
2º modo flexão na direção Z	Z2 56.4Hz	2Z	AC1Z		0.650%	
			AC4Z		0.660%	
		3Z	AC1Z		0.636%	
			AC4Z		0.644%	
2º modo torção	T2 80.2Hz	4T	AC1Z		0.148%	
			AC2X		0.150%	
			AC3X		0.148%	
			AC4Z		0.157%	

Para os modos que apresentaram taxas de amortecimento não constantes (X1 e T1), subdividiu-se o intervalo total do teste em subtrechos onde o amortecimento pôde ser considerado aproximadamente constante.

Para comparação com o presente método, os sinais dos testes 0X e 2Z foram analisados utilizando-se o método das exponenciais amortecidas (LSCE) como definido por ALLEMANG, BROWN (1987). A maior dificuldade encontrada pelo LSCE foi devido às não lineariedades presentes nos sinais. Sendo um método que supõe taxas de amortecimento constantes por toda a vida útil do sinal, o mesmo teve dificuldade de lidar com os sinais experimentais aqui obtidos, o que causou uma grande dispersão nos resultados que, por este motivo, não foram mostrados. Ao tentar-se repetir a análise numérica variando-se parâmetros verifica-se que existe uma instabilidade muito grande no valor dos resultados, que variam em uma larga faixa.

### **Testes experimentais com um trecho de riser flexível**

Este exemplo final constitui-se da análise de sinais resultantes de ensaios experimentais realizados com o intuito de estimar a taxa de amortecimento de um riser flexível. O riser flexível é um tubo formado por várias camadas metálicas e de resinas utilizado para conduzir óleo ou gás nas plataformas de petróleo. Estes ensaios fazem parte de um projeto contratado pela Petrobrás e está sendo realizado no Laboratório de Estruturas sob a coordenação dos professores Ney Roitman e Carlos Magluta.

Este ensaio foi montado engastando uma das extremidades de um trecho de riser de 4 polegadas com comprimento de 4m e mantendo-se a outra extremidade livre. Nesta extremidade livre foram adicionadas massas através de chapas de aço. Os ensaios consistiram de impor um deslocamento inicial à extremidade livre e posteriormente liberá-la, permitindo assim que o elemento estrutural vibre livremente.

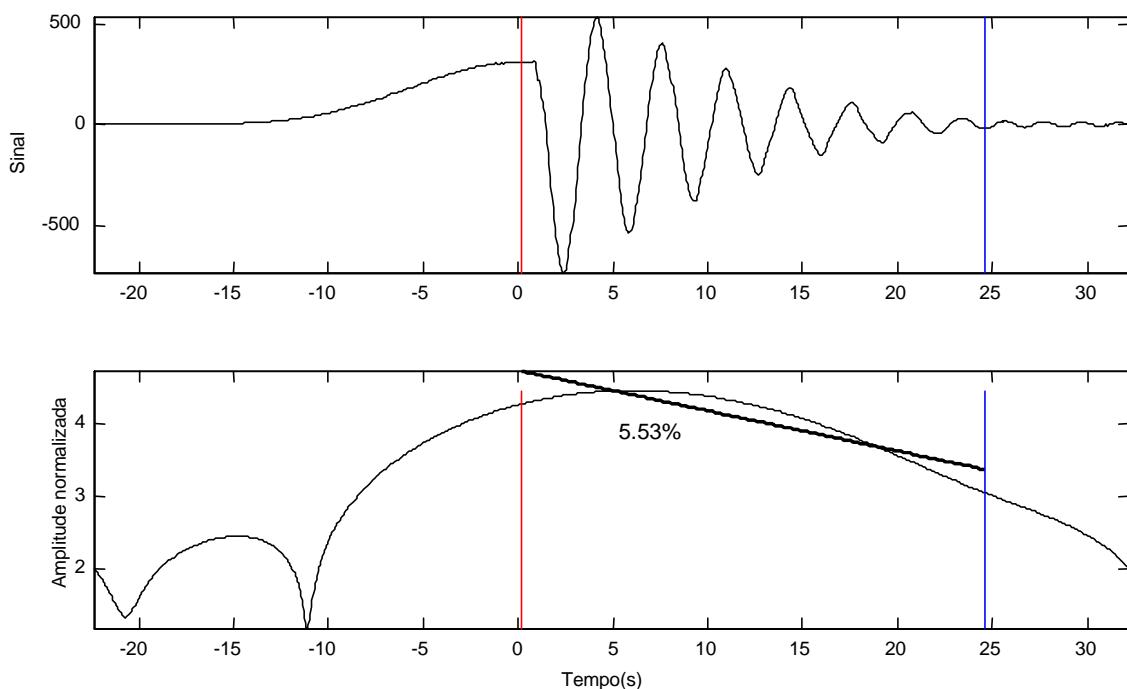
A resposta da estrutura foi medida através de três extensômetros elétricos de resistência colados diretamente à camada externa do riser e um acelerômetro preso próximo à extremidade livre. Medições realizadas mostram que o comportamento desta sub-estrutura é dominado por taxas de amortecimento elevadas, o que resulta em poucos ciclos úteis para análise, impedindo o uso de técnicas tradicionais de forma confiável, sendo portanto um bom desafio para a técnica aqui desenvolvida.

Este exemplo tem a finalidade de tão somente aplicar a técnica recém introduzida da pré-modulação do sinal em casos de alto amortecimento e baixas frequências, onde a técnica desenvolvida neste capítulo tem dificuldades em obter resultados satisfatórios. Portanto será conscientemente suprimida a apresentação dos detalhes do modelo

experimental o que, ao final, iria pouco ou nada acrescentar à apreensão dos conceitos ora introduzidos, objetivo último deste item.

A Figura 62 mostra o sinal obtido com o riser, completamente carregado com placas de aço, levado a uma posição inicial e então liberado. Pode-se notar que, devido ao carregamento, a frequência principal de vibração é de apenas 0,29 Hz. Nota-se que o primeiro pico tem amplitude maior que a posição no instante da liberação mas isto é devido a um enjanelamento das extremidades do sinal, necessário pela periodicidade da transformada de Fourier. Esta particularidade, no entanto, não afeta – ou afeta muito pouco - a presente análise.

A duração do modo (adotada) é de 25s e, para este valor vê-se que a taxa de comprometimento mínimo do sinal é, segundo (51), de 37%, que é um valor excessivamente alto. A largura (útil) da janela referente a este fator de comprometimento mínimo é 37% de  $Dm$ , ou seja, 9,25s. A curva de amortecimento obtida utilizando-se uma janela com esta largura é mostrada também na Figura 62.

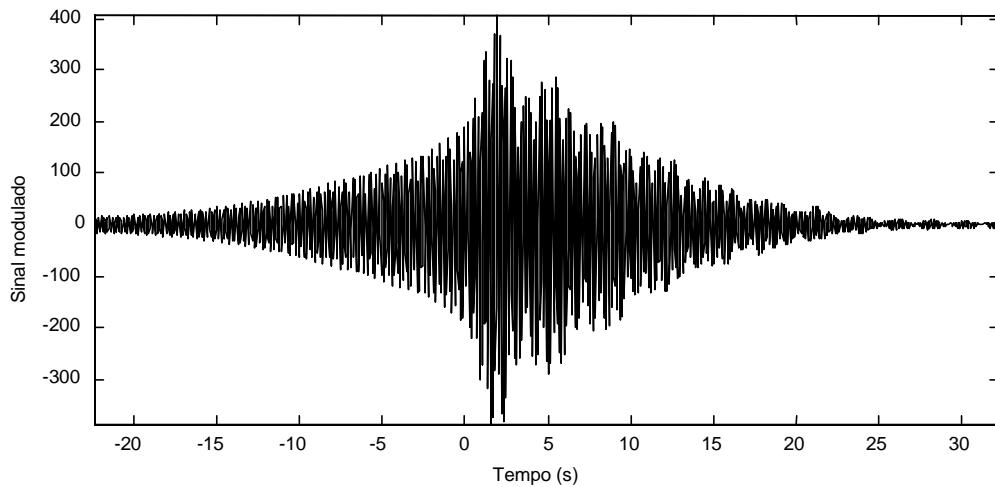


**Figura 62 Riser carregado: sinal e respectiva curva de amortecimento obtida sem pré-modulação**

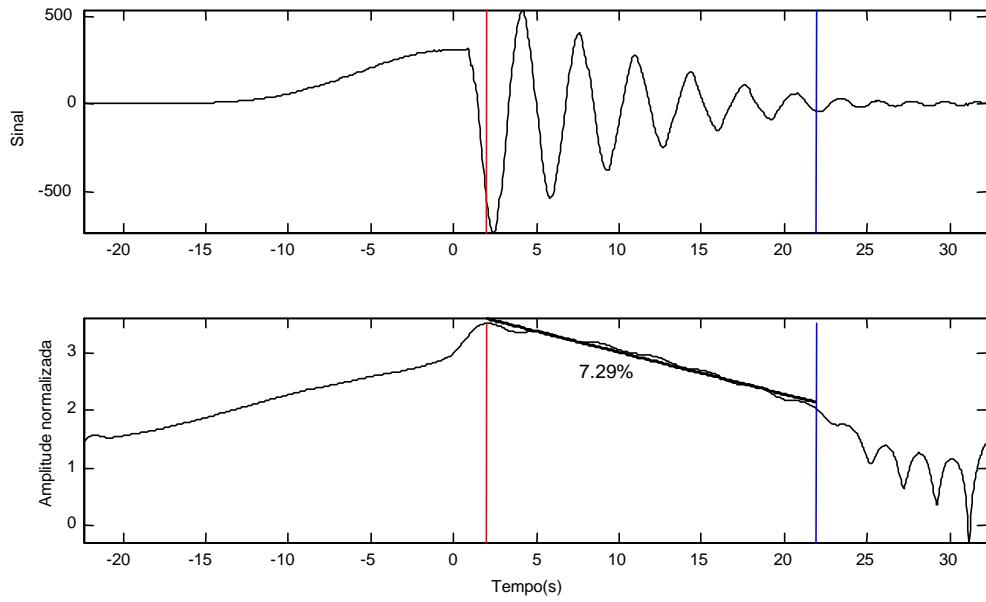
Pode-se observar na Figura 62 que a curva de amortecimento é suave mas está bastante comprometida. Desde que o sinal está digitalizado a uma taxa de 20 pontos por segundo, pode-se realizar uma pré-modulação do sinal da ordem de 6 Hz com segurança.

O sinal já modulado é mostrado na Figura 63. Vê-se que o efeito da modulação é inserir mais ciclos no sinal de forma que a TFD tenha sensibilidade suficiente para perceber o decaimento da energia do sinal. Com esta pré-modulação o fator de comprometimento mínimo cai para apenas  $1/\sqrt{(6,29\text{Hz} \times 25\text{s})} = 8\%$ , um valor bem mais razoável.

A partir deste sinal pré-modulado é calculada a respectiva TFD utilizando uma janela de largura (útil) igual a 8% da duração do modo  $Dm$ , ou seja, 2s. A partir da TFD calculada é obtida a respectiva curva de amortecimento, mostrada na Figura 64.



**Figura 63 Sinal modulado a 6 Hz**



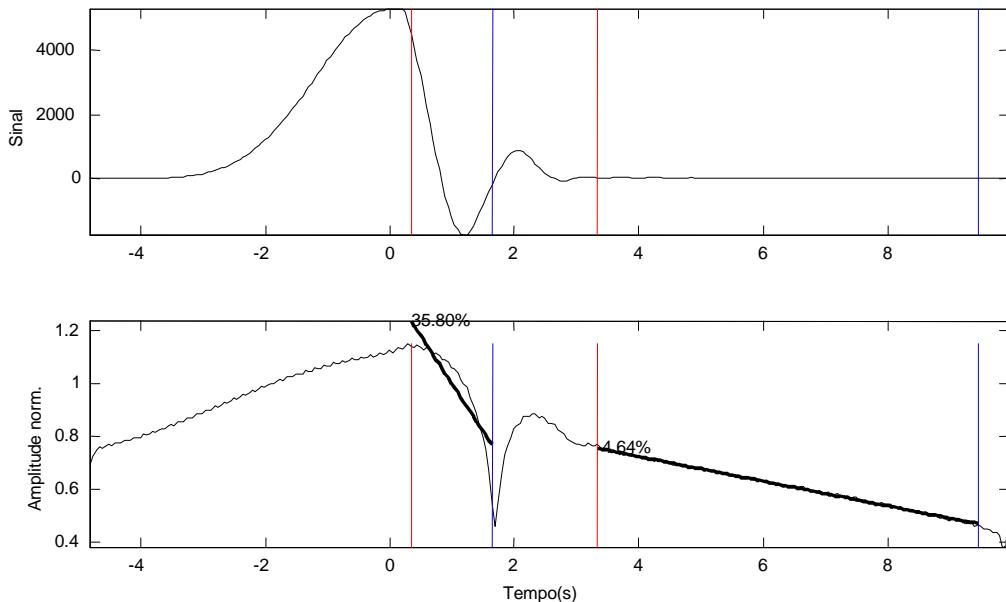
**Figura 64 Sinal experimental e respectiva curva de amortecimento;  
pré-modulação a 6 Hz**

Apreende-se da Figura 64 que a curva de amortecimento agora está bem mais precisa, isto é, reflete com maior segurança as variações na energia do sinal. Próximo ao

instante 0s, por exemplo, nota-se que a energia é subitamente retirada, fato que se reflete pela imediata mudança na declividade da curva.

A Figura 65 mostra o sinal obtido com o riser, sem carga, levado a uma posição inicial e então liberado. Pode-se notar que após o instante de liberação (primeira linha vertical) o sinal entra em uma frequência de oscilação da ordem de 0,50Hz, quase duas vezes maior que a anterior pela retirada das placas de aço.

O cálculo da curva de amortecimento foi feita utilizando-se a mesma modulação do caso anterior (6Hz) e uma largura de janela de 0,50s.



**Figura 65 Riser sem carga: sinal e respectiva curva de amortecimento**

Nota-se que existe, a partir da liberação, uma crescente perda de energia até um nítido ponto mínimo, indicado pela segunda linha vertical; a média deste trecho fornece um valor para a taxa de amortecimento da ordem de 35,80%. A partir deste ponto de mínimo, que corresponde ao instante onde o sinal passa pelo zero, existe uma inesperada recuperação da energia até um segundo ponto de máximo, indicando que a grande perda de energia do trecho anterior foi apenas devida à acumulação temporária de energia elástica, posteriormente liberada.

No trecho seguinte, do instante 2s a aproximadamente 3.8s existe a repetição do mesmo fenômeno não-linear, embora com menor intensidade. Após o instante 4s o modelo parece definitivamente assumir um comportamento linear, quando a amplitude normalizada assume a forma de uma reta cuja taxa de amortecimento estabiliza-se em 4,64%. Isto parece indicar que as não-linearidades estão intimamente ligadas às altas amplitudes. Para baixas amplitudes, o comportamento torna-se linear.

# IV COMPRESSÃO DE SINAIS

Como parte do estudo das técnicas tempo-frequência foi inevitável a introdução do estudo das wavelets pois esta teoria está atualmente tornando-se bastante popular. Entretanto as primeiras formas de aplicação das wavelets neste trabalho tiveram êxito mais no sentido de como não usá-las, como por exemplo na geração de gráficos tempo-frequência, isto porque nestes casos as distribuições tempo-frequência são muito mais eficientes e também de melhor qualidade.

Percebeu-se ao decorrer dos trabalhos que as principais características das wavelets que deveriam ser realmente exploradas estavam relacionadas às suas propriedades de filtragem e compressão que ocorrem, por sinal, simultaneamente. Este capítulo, portanto, dedica-se a explorar exatamente estas propriedades inerentes da transformada, adotando-se um enfoque menos matemático – o usual – e mais voltado às necessidades da engenharia.

Inicialmente será mostrada superficialmente as formas de compressão usuais e então será mostrado como proceder para obter filtragem e compressão simultâneas com o uso da transformada wavelet. A seguir vários sinais resultantes de monitoração de tráfego serão analisados e as propriedades das wavelets comparadas às respectivas propriedades da transformada de Fourier. Ao final do capítulo será mostrado como a compressão de sinais, unidimensional, pode ser extendida para o domínio bidimensional.

## *IV.1 Compressão de dados*

Apesar de, atualmente, a tecnologia ter avançado tanto que grandes quantidades de dados podem ser transferidas entre continentes a taxas muito altas, a indústria de comunicação tem investido muito tempo e dinheiro em compressão de dados. Isso porque os recursos ainda são caros e o potencial da demanda é grande. A engenharia civil, agora valendo-se cada vez mais dos recursos da eletrônica e da informática, também passa a ver-se às voltas com problemas quanto à quantidade de dados aquisitados.

Quando a massa de dados passa a ser aquisitada além de uma velocidade crítica existem dois problemas a resolver: (1) como armazená-los e (2) como transmiti-los. Para restringir um pouco o escopo do problema, faz-se um exemplo: imagine-se um sistema

de instrumentação aquisitando 100 canais a uma taxa modesta de 300 pontos por segundo. Estes canais irão produzir dados a uma taxa de 300 kbytes/s, velocidade que exigirá certamente uma infraestrutura cara para transmissão destes dados. Um esquema de compressão simples, como o que será desenvolvido aqui, poderia ser diretamente utilizado neste caso, reduzindo a taxa de transmissão para 6 kbytes/s, capacidade facilmente atingida por um modem doméstico.

Além disso, existe o problema de armazenagem. A aplicação do exemplo exigiria um esquema de armazenagem extremamente caro, com dispositivos SCSI atuando em paralelo e backups diários, além de uma equipe de plantão para eventuais problemas no hardware. Por outro lado, a compressão de dados permite novamente a reutilização de discos rígidos comumente encontrados no mercado, implicando em uma redução de custo.

O exemplo serve para ilustrar que, nos dias atuais, o uso de compressão não é mais facultativo em aplicações críticas. Isso não é novidade, argumenta-se. Contudo a informação que parece ser ignorada é que os esquemas de compressão são altamente dependentes do tipo de dados aos quais serão aplicados.

Uma alternativa ao desenvolvimento de um esquema particular de compressão é a utilização de algum compressor já pronto disponível no mercado, talvez derivados dos padrões MPEG-3 para sons ou JPEG para imagens (comentados mais adiante). O risco da utilização destes esquemas está em que informações importantes sejam irremediavelmente eliminadas pelo compressor. Isto acontece porque o padrão MPEG-3, por exemplo, foi projetado para eliminar estruturas sonoras que, de qualquer forma, seriam ignoradas pelo sistema auditivo humano mas que poderiam conter informações relevantes para uma análise estrutural. O formato JPEG igualmente foi construído para comprimir os arquivos eliminando estruturas visuais que de qualquer forma serão imperceptíveis ao olho humano.

Ao analisar o código-fonte de tais algoritmos (disponível gratuitamente na internet no caso do formato JPEG) percebe-se que ambos possuem tabelas que armazenam valores binários estudados à exaustão que armazenam combinações conhecidas de som/cor que podem ser desprezadas sem maior prejuízo à percepção humana. Em resumo, estas tabelas definem parâmetros de cor e intensidade de luz obtidos experimentalmente de forma que as taxas de compressão sejam as mais altas possíveis sem afetar significativamente a qualidade visual da imagem.

Portanto, um esquema eficiente de compressão é antes de tudo um algoritmo especificamente desenvolvido e dirigido para uma determinada aplicação que traz embutido em seu código conhecimento prévio da massa de dados ao qual será aplicado. Desta maneira, antes de se comprimir algum dado, é necessário um estudo completo a fim de definir, através de uma descrição matemática possível de ser implementada como um algoritmo, critérios de qualidade que determinarão o que tem relevância ou não nos sinais.

Neste contexto, as wavelets facilitam o projeto de tais esquemas de compressão ao fornecer um substrato teórico, matemático e computacional bastante maleável, capaz de adequar-se para atender a uma boa gama de aplicações. Um exemplo disto é a utilização pela polícia federal dos EUA (FBI) da compressão com wavelets para armazenar em meio digital as digitais de seus arquivos (BRADLEY, BRISLAWN, HOPPER, 1992). Em FROMMEN, MALLAT (1992) anuncia-se como a segunda geração de compactadores de imagens aqueles que utilizam-se das wavelets.

Os esquemas de compressão de dados digitais podem ser classificados em dois tipos:

1. Sem perda. A compressão e a descompressão são transformações exatamente reversíveis, isto é, os dados restaurados após a descompressão são exatamente iguais aos dados originais. São os únicos esquemas possíveis de serem aplicados a arquivos texto, por exemplo. Permitem taxas modestas de compressão, tipicamente 1.5:1 a 3:1. Ex.: LZW, ZIV, (LEMPEL, 1977), RLE (MICROSOFT, 1991), arithmetic coding (WITTEN, NEAL, CLEARY, 1987).
2. Com perda. Os dados descomprimidos não são exatamente iguais mas assemelham-se bastante aos originais. São geralmente aplicados a imagens ou sons, onde pequenas alterações nos dados não são perceptíveis. Permitem altíssimas taxas de compressão, dependendo do sinal, tipicamente 20:1 a 30:1 podendo chegar a taxas de até 1000:1. Ex.: EZW (SHAPIRO, 1993), MPEG (LE GALL, 1991), Indeo (MACROMEDIA, 1993), JPEG (WALLACE, 1991), AVI (MICROSOFT, 1991), QuickTime (HORNE, 1993).

Um excelente guia para iniciar a exploração do universo dos formatos de arquivos gráficos está em (CARPENTER, MUMFORD , 1992).

A questão relevante neste ponto é: os esquemas de compressão precisam necessariamente eliminar algum tipo de informação? A resposta é afirmativa, pois entre

os compressores *sem perda* (*lossless*) e os *com perda* (*lossy*), os últimos têm desempenho médio muito superior. Existem casos, porém, que não é possível eliminar-se informação alguma, como é o caso de textos ou programas, quando esquemas sem perda devem obrigatoriamente ser utilizados. Para os casos onde pode-se aceitar a eliminação de algum tipo de informação, os esquemas com perda invariavelmente têm desempenho superior.

O que se deseja, no caso, é que as informações descartadas sejam irrelevantes, como o ruído ambiente. Em certos casos os sinais relativos à passagem de um caminhão podem ser relevantes mas a passagem de carros pequenos pode ser irrelevante. E esta distinção pode ser programada no esquema de compressão desde que a teoria seja maleável suficientemente para contemplá-la.

Os esquemas de compressão com perda invariavelmente podem ser encaixados num tipo de metodologia denominada *codificadores baseados em transformadas* (ou *transform coders*, em inglês). Esta metodologia consiste basicamente de três etapas:

- (1) transformação de domínio;
- (2) quantização;
- (3) codificação entrópica (compressão sem perda).

Na primeira etapa tradicionalmente utiliza-se a transformada cosseno de Fourier discreta (DCT ou *Discrete Cosine Transform*). A teoria tempo-frequência será utilizada nesta etapa, substituindo a DCT pela transformada wavelet rápida (FWT ou *Fast Wavelet Transform*). O novo formato JPEG 2000 (CHRISTOPOULOS *et al*, 2000) efetivamente utiliza as wavelets como transformada básica para a compressão de imagens.

## ***IV.2 Exemplo de compressão via transformada wavelet***

Antes de apresentar uma aplicação concreta, será feito um exemplo a fim de ilustrar como a teoria das wavelets pode ser usada comprimir um sinal. Os coeficientes calculados no exemplo do capítulo II são:

$$C = \{155.0 \quad -80.0 \quad -3.2 \quad 6.7\}$$

Pode-se observar que os dois primeiros coeficientes 155 e -80.0 são pelo menos uma ordem de grandeza superiores aos dois outros coeficientes. Se for consenso que

variações no sinal dentro da faixa de  $\pm 10$  unidades podem ser desprezadas, então pode-se desprezar também os dois últimos coeficientes. Geralmente, no caso de sinais maiores, muitos coeficientes situam-se muito próximos ao zero e podem ser desprezados sem perda significativa de qualidade. Este processo simples é a base da compressão wavelet.

Desta forma um novo vetor de coeficientes  $C^*$  pode então ser obtido:

$$C^* = \{155.0 \quad -80.0 \quad \text{CNR} \quad \text{CNR}\}$$

onde a sigla CNR indica *Coeficiente Não-Relevante*. Poderia-se ter simplesmente substituído o coeficiente por zero, o que efetivamente é feito, mas preferiu-se dar ênfase ao aspecto da relevância do coeficiente.

A compressão com wavelets, em sua forma mais simples, termina neste ponto. O restante do trabalho consiste em empacotar da melhor forma possível este vetor  $C^*$  em um arquivo, isto é, definir um formato de compressão.

Existem, no entanto, duas dificuldades essenciais que podem não ter sido notadas:

1. Determinar qual família de wavelet a se utilizar (foi utilizada a wavelet de Haar neste exemplo). Este é um assunto especialmente estudado por RIOUL (1993) e MANDAL, PANCHANATHAN, ABOULNASR (1993).
2. Determinar quais coeficientes serão eliminados ou alterados, tarefa que depende de conhecimento prévio do tipo de dados que serão comprimidos. Neste exemplo foram eliminados os dois menores coeficientes - em módulo.

Nestas duas tarefas concentram-se basicamente a grande parte das pesquisas realizadas em compressão com wavelets, inclusive com a criação de novas famílias de wavelets especializadas em determinados tipos de sinais.

## **Thresholding**

À operação de corte dos menores coeficientes, exemplificada a pouco, dá-se o nome de *thresholding* e o valor 10,0 é denominado *threshold value* ou simplesmente *threshold*.

A escolha dos valores de threshold tem razões empíricas e depende exclusivamente do conteúdo do sinal. Deve ser entendido que cada nível de coeficientes wavelet cobre

uma faixa específica de frequência. O primeiro nível (1) cobre aproximadamente a faixa de  $fd/2$  até  $fd/4$  onde  $fd$  é a taxa de digitalização. O segundo nível, de  $fd/4$  a  $fd/8$  e assim por diante, sempre em intervalos de oitavas. No caso deste exemplo, se a taxa de digitalização fosse de 100Hz (ou 100 pontos por segundo), então os coeficientes do nível 1 são responsáveis pelo conteúdo das frequências entre 50 e 25Hz; os do nível 2, pelas frequências entre 25 e 12.5Hz. Se houvesse um nível 3, este seria responsável pelas frequências entre 12.5 e 6.25Hz. Portanto, adotar um valor de 10 para o threshold do nível 2 significa, na faixa de frequências de 12.5 a 25Hz, bloquear todos os eventos cuja amplitude seja menor que 10 unidades. Desta forma, a escolha do threshold a ser utilizado depende da noção do que é significativo em cada faixa de frequência coberta por cada nível de coeficientes.

O “thresholding” pode ser classificado como global ou local e também em *hard* ou *soft*. Estes termos significam:

*Global* – indica que o mesmo valor é utilizado para fazer o corte em todos os níveis;

*Local* – indica que cada nível (ou escala) possui um valor de threshold diferente;

*Hard* – indica que o corte afeta somente os coeficientes que estejam dentro do intervalo especificado. No caso, [-10,+10]. Os coeficientes dentro da faixa são eliminados e os fora da faixa permanecem intactos.

*Soft* – idêntico ao *Hard*, *exceto* que os coeficientes fora do intervalo também são afetados. Os coeficientes dentro da faixa, portanto, são eliminados e os fora da faixa são subtraídos – em módulo – do valor do threshold.

O *soft thresholding* é mais coerente com a suposição da contaminação com ruído (DONOHO, 1993) pois o ruído afeta a todos os níveis de coeficientes (leia-se faixas de frequências) com a mesma intensidade. Desta maneira, retirando-se do módulo de todos os coeficientes sua suposta intensidade média, e não somente os de menor módulo, está-se corrigindo o ruído de uma forma mais efetiva. No entanto este tipo de discussão é polêmica e a prática tem consagrado o uso do *hard threshold*, mais fácil de ser implementado.

Se fosse aplicado um *local soft thresholding* ao exemplo, com valores de threshold de  $\pm 5$  para o nível 1 e  $\pm 10$  para o nível 2, teria-se o seguinte vetor resultante:

$$C^* = \{150.0 \quad -75.0 \quad CNR \quad CNR\}$$

pois os dois últimos elementos são menores que 10 (em módulo) e os dois primeiros, do nível 1, são maiores que 5 e, portanto, são subtraídos (em módulo) deste valor.

A escolha entre *local* ou *global threshold* depende do tipo de ruído que suspeita-se contaminar o sinal. Se a suspeita da contaminação for por um ruído do tipo branco então um valor constante de corte (*threshold*) para todos os níveis (leia-se frequências) seria uma boa estratégia. Isto, na prática, indica o uso do *global threshold*. No entanto, se a suspeita for de um ruído do tipo colorido (*colored noise*) como, por exemplo, o ruído vermelho (*red noise*) cuja distribuição concentra-se nas altas frequências então um valor distinto para cada nível (leia-se faixa de frequências) é a metodologia mais adequada.

## Quantização

Após escolhido o tipo de corte é necessário quantizar os coeficientes. O termo *quantização* refere-se à substituição do valor em ponto-flutuante (geralmente um *double* de 64 bits ou *float* de 32 bits) por um valor inteiro com menos bits. No caso atual serão utilizados 3 bits para o módulo mais 1 bit para o sinal, o que dará 4 bits (que cobrem valores de  $-2^3+1$  a  $2^3-1$ ).

Deve ser observado que o efeito da quantização sobre os coeficientes wavelet é menos danoso que o efeito da quantização diretamente sobre o sinal no tempo, que pode ser desastroso. No tempo, o número de bits para quantização deve ser, no mínimo, de 12 bits sob pena de má representação do sinal. Já nos coeficientes wavelet um grande erro local de quantização é redistribuído sobre uma faixa no tempo, reduzindo o impacto. Desta forma, 8 ou até menos bits são suficientes para representar um coeficiente.

Utilizando o vetor inicial ao qual foi aplicado o “hard global threshold” ( $C^*$ ), aplica-se a quantização obtendo-se

$$C^{**} = \text{Arredonda}((2^{\text{nbits}}-1) \cdot C^*/\max(C^*)) = \{7 \quad -4 \quad CNR \quad CNR\}$$

Esta forma de quantização é denominada *uniforme* por definir uma malha uniformemente distribuída sobre o eixo real. Uma forma mais sofisticada é a quantização vetorial (GERSHO, 1995) que efetivamente é utilizada nos formatos de arquivos gráficos mais eficientes.

## Codificação entrópica

A terceira etapa, a codificação entrópica, será parcialmente suprimida dado que os codificadores entrópicos são invariavelmente muito mais lentos que os algoritmos das etapas (1) e (2). Adicionalmente, para uma grande classe de sinais, os coeficientes transformados com wavelet e quantizados já têm uma entropia bem próxima à mínima sendo esta etapa, portanto, de pouca relevância para esta aplicação. Esta conclusão tem base em uma campanha de testes realizada especificamente para este trabalho utilizando-se vários sinais obtidos no Laboratório de Estruturas. Portanto, esta etapa consistirá apenas em empacotar os coeficientes mais relevantes.

Este empacotamento é feito utilizando-se um *bitstream*, ou seja, uma interface de software para arquivos que aceita operações bit-a-bit do tipo

```
void putbit(bool bit);

bool getbit();

void putbits(void *bits, int count);

void getbits(void *bits, int count);
```

O formato de arquivo conterá as seguintes informações:

- (1) Todos os coeficientes relevantes;
- (2) Os respectivos índices destes coeficientes relevantes;

O código em C++ que utiliza o bitstream (`bit_stream`) para gravar os `nc` maiores coeficientes (já ordenados no vetor `coef`) e os respectivos índices (também ordenados no vetor `idx`) é mostrado a seguir, onde `nbits` é o número de bits de cada coeficiente quantizado e `nidxbits` é o número de bits efetivamente utilizados pelos índices.

```
for (i=0; i<nc; i++) bit_stream.putbits(&coef[i], nbits);

for (i=0; i<nc; i++) bit_stream.putbits(&idx[i], nidxbits);
```

Seguindo esta orientação, o vetor de coeficientes final – comprimido – será:

```
C*** = { 7 -4 0 1 }

-----
coeficientes índices

(2x4 bits) (2x2 bits) = 12 bits total
```

Pode parecer que este vetor comprimido seja igual ao vetor  $x$  original pois ambos têm 4 números, mas é engano. O vetor original ocupa  $4 \times 64 = 256$  bits de memória enquanto o vetor comprimido ocupa 12 bits, pois cada coeficiente quantizado possui 4 bits e cada posição pode ser representada por um número de 2 bits (0 a 3). Isto corresponde a uma compressão de  $(256-12)/256 = 95,3\%$  ou seja, 21:1. Para sinais de maior tamanho as taxas serão naturalmente maiores pois o número de coeficientes não-relevantes tenderá a ser relativamente maior, o que compensaria de folga o acréscimo no número de bits dos índices.

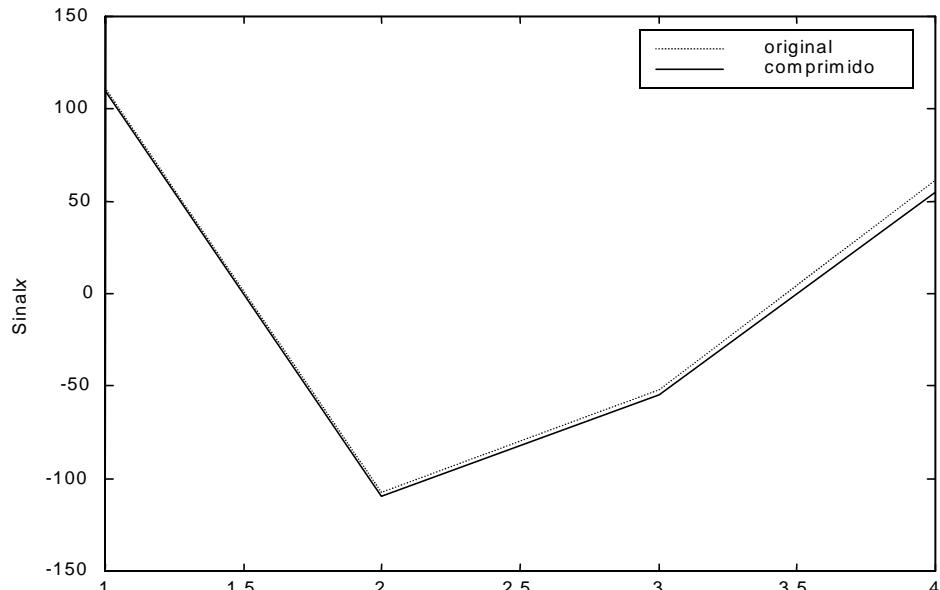
A não inclusão do valor máximo para quantização 155,0 no arquivo resultante é uma prática correta desde que, conhecidos os sinais que serão comprimidos, pode-se adotar um valor máximo fixo que será inserido na programação do algoritmo como uma constante.

### Erro introduzido pela compressão

Para verificar-se o efeito das perdas por *threshold* e por quantização, são feitas todas as operações inversas usando-se este vetor de coeficientes, quando obtém-se o sinal descomprimido:

$$x^* = \{109.6 \quad -109.6 \quad -54.8 \quad 54.8\}$$

A Figura 67 mostra graficamente a comparação entre os sinais original ( $x$ ) e comprimido-descomprimido ( $x^*$ ).



**Figura 67 Sinais original e obtido após compressão-descompressão com perda**

Vê-se pela Figura 67 que a diferença entre os sinais é quase imperceptível visualmente. O resíduo resultante da compressão-descompressão com perda e o erro RMS (*Root Mean Square*) são, respectivamente:

$$\text{Resíduo} = \mathbf{x} - \mathbf{x}^* = \{1.8 \quad 1.8 \quad 3.2 \quad 6.7\}$$

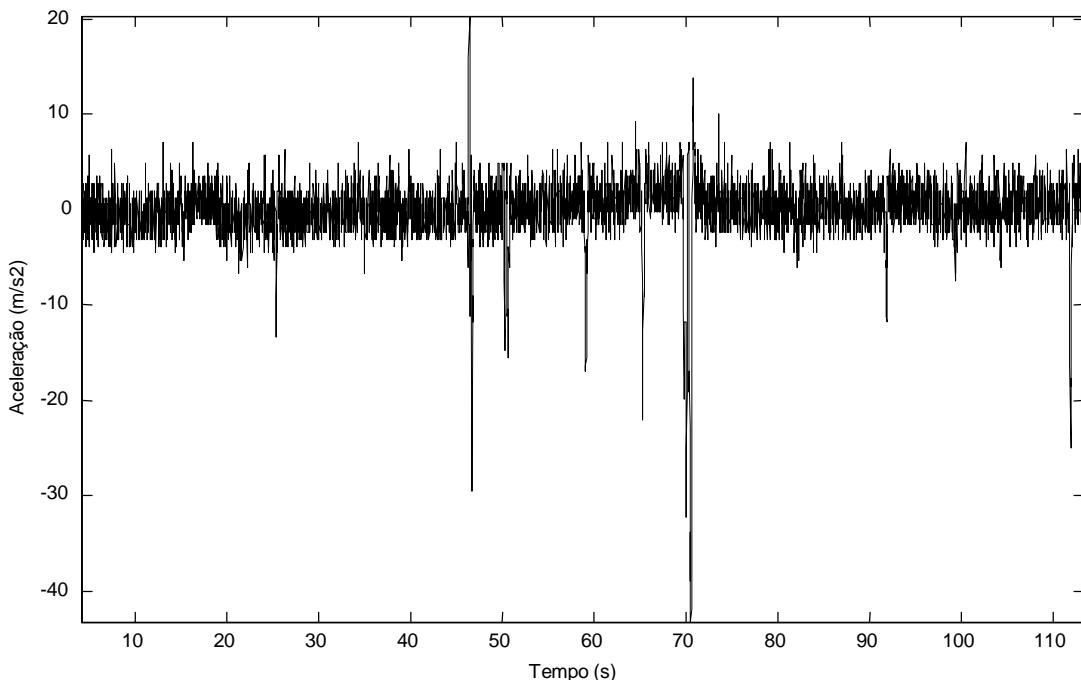
$$\text{RMSE} = \sqrt{\sum \frac{1}{4}(x_i - x_i^*)^2} = 3.92$$

que estão dentro da faixa utilizada para corte de 10 unidades.

### **IV.3 Compressão de sinais de tráfego**

Na seção anterior foi visto como as técnicas tempo-frequência podem ser usadas para comprimir um sinal através de um exemplo simples e didático. Para verificar o comportamento da técnica utilizada naquele exemplo frente a uma aplicação prática, o mesmo método de compressão será utilizado para comprimir sinais obtidos da instrumentação de uma via com tráfego automotor.

Um sinal típico foi selecionado entre os sinais obtidos por um sistema de monitoramento de tráfego conforme mostrado na Figura 68, correspondente a 110s de monitoração a uma taxa de digitalização de 140 S/s, tendo portanto 15400 pontos.

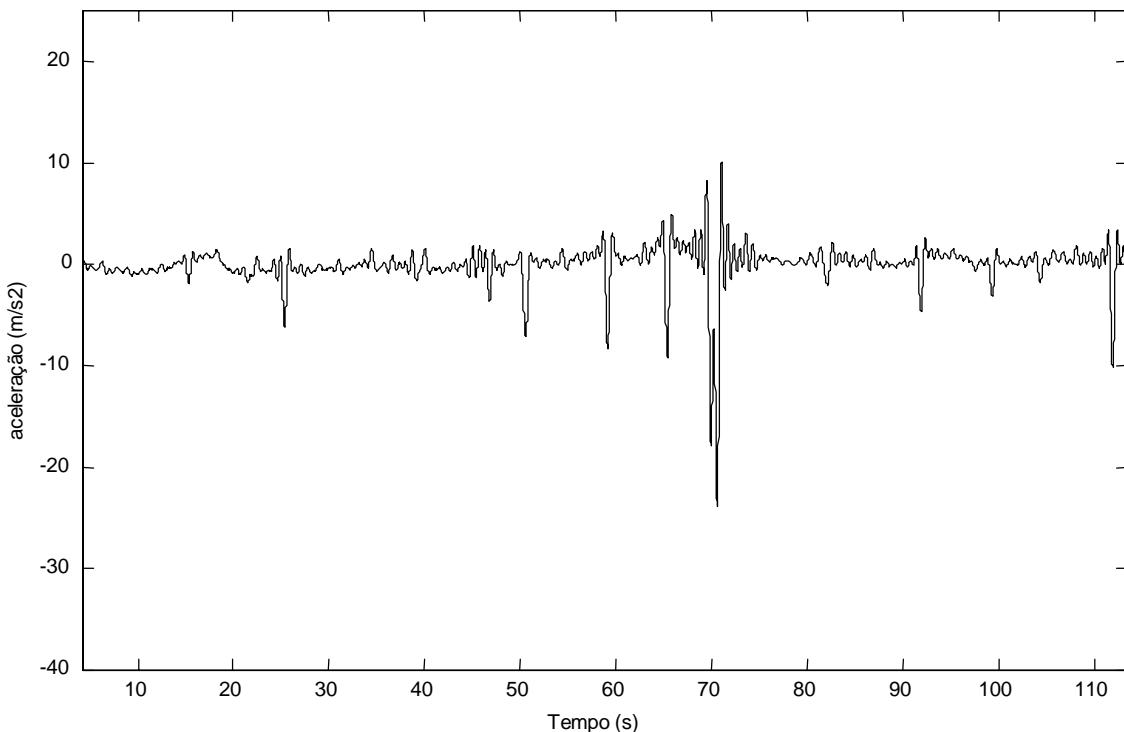


**Figura 68 Sinal típico de tráfego**

No sinal mostrado na Figura 68 pode-se perceber que há bastante ruído no sinal. Percebe-se também que existem grandes descontinuidades, aproximadamente nos

instantes 48s, 51s, 60s, 65s, 70s (a maior) e 112s. Estas descontinuidades correspondem à passagem de veículos de grande porte como ônibus e caminhões. Adicionalmente existem outras descontinuidades de menor dimensão relativas a veículos menores e aparentemente, uma forma de onda de baixa frequência não identificada por volta de 20s.

A primeira providência ao se constatar tamanho nível de ruído seria filtrar o sinal utilizando-se um filtro passa-baixa. Com efeito este filtro elimina grande parte do ruído mas às custas da eliminação relevante pois o sinal está digitalizado a uma taxa-limite, isto é, não existe folga no espectro para uma filtragem. O efeito negativo notado imediatamente após a filtragem é o abaulamento dos picos com perda de sua amplitude, conforme é ilustrado na Figura 69.

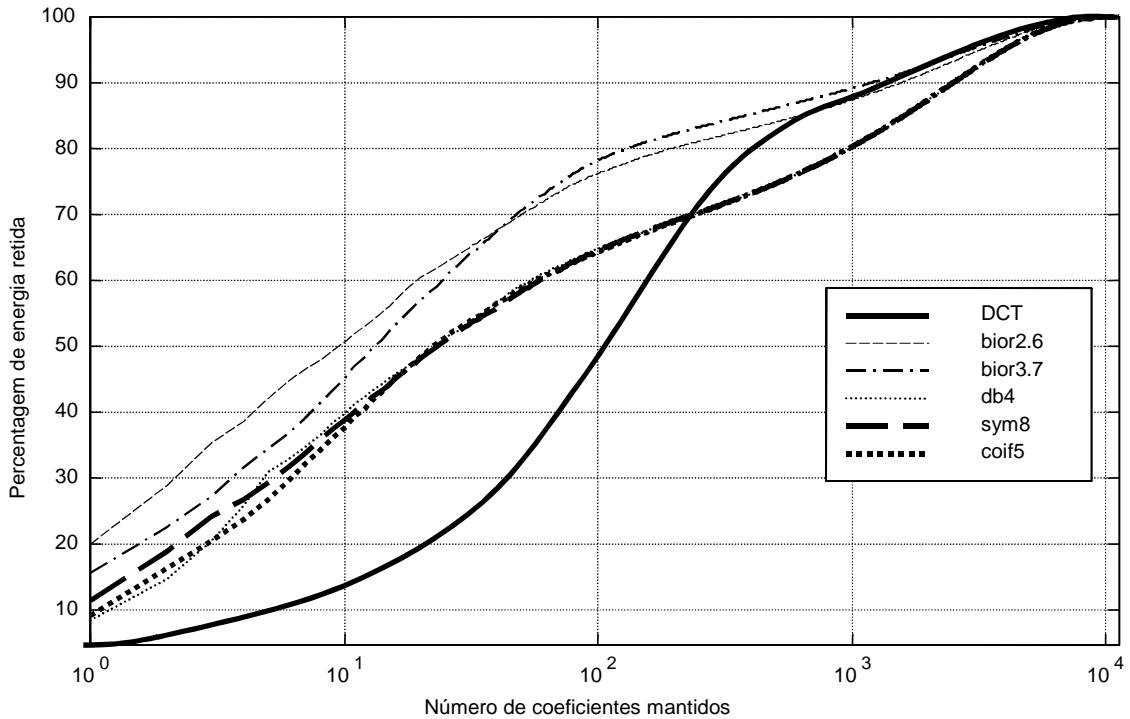


**Figura 69 Sinal de tráfego filtrado com filtro elíptico ordem 8, ripples 0,5dB/40dB, frequência de corte 2Hz**

Percebe-se na Figura 69 que o pico principal, por volta de 70s foi modificado de 40m/s<sup>2</sup> para 23m/s<sup>2</sup>, uma redução de 45%. O segundo maior pico, por volta de 48s, sofreu uma redução de 30m/s<sup>2</sup> para 4m/s<sup>2</sup>, ou seja, 87%! É uma informação valiosa que está sendo eliminada caso a estrutura estudada contenha modos superiores a 2Hz. A atenuação do filtro com o aumento da frequência de corte ou mesmo a redução de sua ordem resultaria no rápido retorno do ruído ao sinal, o que é indesejável. Será mostrado adiante que a wavelet, além de comprimir, elimina o ruído de forma eficiente, caso ele

seja do tipo ruído branco, como pode-se constatar na Figura 72. Nesta figura é mostrado o resultado final do processo implementado com as wavelets. O primeiro passo para realizar a compactação através da FWT é a escolha da família de wavelets mais adequada ao problema. Para tanto, ao sinal foi aplicada a transformada wavelet utilizando-se cinco diferentes famílias: B-Spline ordem 2/6, B-Spline ordem 3/7, Daubechies 4, Symmlet 8 e Coiflet 5 (DAUBECHIES, 1992). Também aplicou-se a transformada cosseno discreta (DCT).

No sentido de estudar a qualidade da transformação proporcionada por estas seis transformadas (cinco wavelets mais a DCT), apresenta-se na Figura 70 um gráfico relacionando a percentagem da energia do sinal retida *versus* o número de coeficientes. A energia de um coeficiente, por convenção, é definida como o quadrado do módulo de seu valor (norma euclidiana  $L^2$ ).



**Figura 70** Percentagem de energia retida para cada transformação

O primeiro fato a ser notado na Figura 70 é que todas as transformadas, tanto as wavelets como a DCT, preservam toda a energia do sinal desde que todos os coeficientes sejam também preservados. Isto pode ser verificado pelo fato de que todas as curvas alcançam o valor 100% de energia preservada para o número total de coeficientes retidos. Isto evidencia o fato de que estas transformadas são *isometrias*, isto é, são representações completas do sinal em um domínio diverso cuja base é ortogonal. Esta ortogonalidade assegura que as transformadas são *reversíveis*, isto é, o

sinal pode ser perfeitamente recuperado através das coordenadas (coeficientes) neste outro domínio.

A única diferença entre os domínios é que algumas bases têm maior capacidade de representar o sinal no tempo com um menor número coeficientes. Pode-se perceber na Figura 70 que as wavelets B-Spline 3/7 e principalmente a 2/6 comportam-se muito melhor que todas as outras pois com o mesmo número de coeficientes conseguem reter mais energia do sinal. A DCT, por sua vez, apresenta uma performance modesta, pois necessita de um grande número de coeficientes para reter uma parcela considerável de energia.

Pode ser observado também que a partir do centésimo coeficiente existe uma mudança de inclinação das curvas, indicando que existe uma alteração na natureza dos coeficientes a partir deste ponto, isto é, existe uma grande possibilidade de que os coeficientes até o centésimo representem eventos fisicamente relevantes e os coeficientes além do centésimo representem apenas ruído.

A partir de cerca de 200 coeficientes retidos a DCT parece melhorar seu desempenho pelo aumento considerável da energia retida. Entretanto deve-se notar que é por volta deste número que começam a aparecer os coeficientes relativos ao ruído. Isto significa que a DCT tem grande capacidade para representar eventos não-localizados, isto é, espalhados no tempo, como o ruído. A wavelet, cujos coeficientes são localizados no tempo, necessita de um grande número de coeficientes para representar o ruído. Por este motivo a DCT tem um melhor desempenho a partir deste ponto.

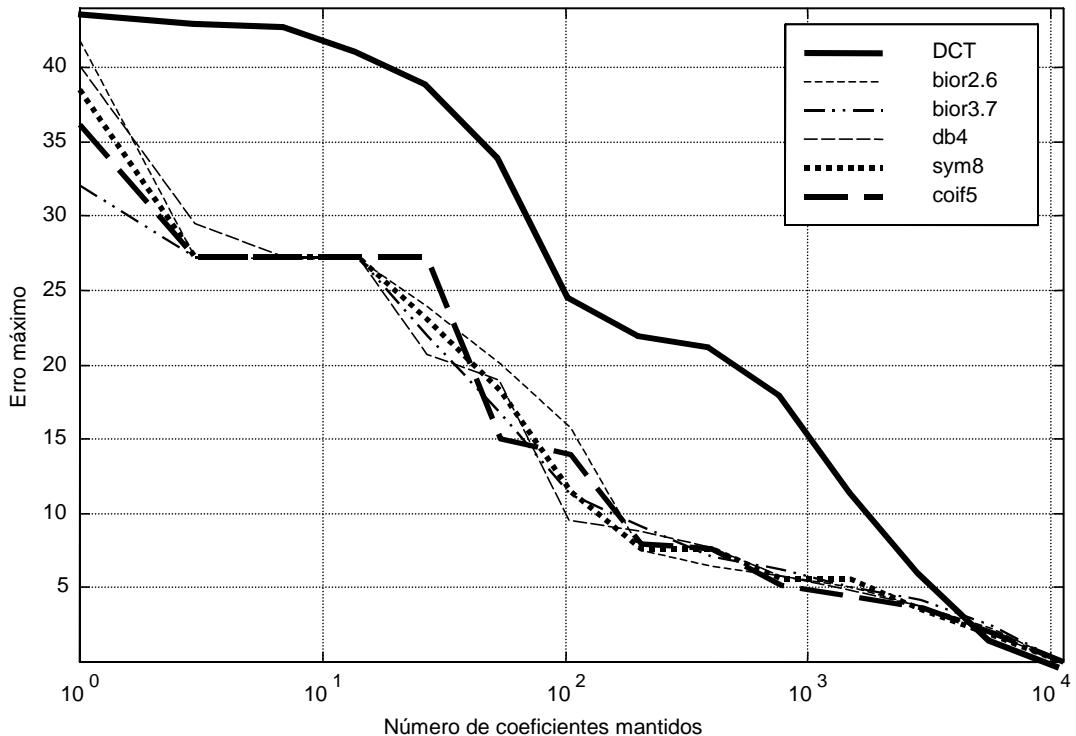
Desta consideração pode-se tirar uma conclusão importante: se o sinal é composto por componentes harmônicos bem espalhados no tempo, possivelmente a DCT (leia-se Fourier) seja mais eficiente para comprimir o sinal no domínio do tempo que a FWT (wavelet), isto é, a wavelet não compete com Fourier. De fato, se o sinal é bastante harmônico, sem perturbações e com taxas de amortecimento mínimas, é indicado que a compressão com wavelet seja feita no domínio da frequência e não do tempo. Para este esquema de compressão deve ser feita inicialmente uma transformada de Fourier sobre o sinal e então a compressão wavelet sobre os coeficientes calculados. Para restaurar o sinal, faz-se a descompressão seguida da transformada inversa de Fourier.

Por outro lado, se o sinal é composto por fortes transientes localizados em curtos espaços de tempo então a FWT definitivamente substitui a DCT.

Entretanto a percentagem de energia retida não é sempre a estimativa de qualidade mais adequada. Em certas aplicações é preciso garantir que os picos do sinal - os valores extremos – sejam bem representados. Na análise de colapso de uma estrutura, por exemplo, é preciso manter bem representados os valores extremos de aceleração à qual a estrutura foi exposta. Na Figura 71 é mostrado o erro máximo absoluto calculado para o mesmo sinal de tráfego, definido como

$$EA_{\max} = \max_{i=1}^N |y_i - x_i| \quad (77)$$

onde  $x$  é o sinal original e  $y$  é o sinal modificado pela compressão posterior à transformação de domínio.



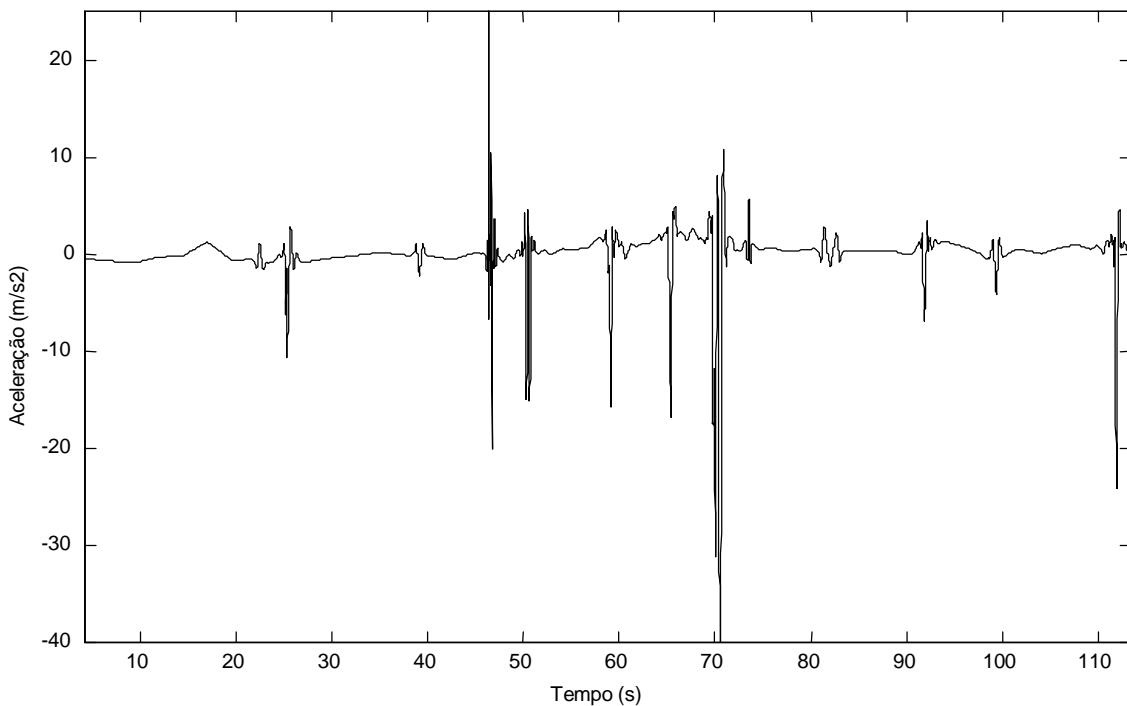
**Figura 71 Erro máximo absoluto para cada transformação**

Pode-se apreender da Figura 71 que se o critério de qualidade utilizado for o erro máximo absoluto, as wavelets, neste exemplo, são sempre bastante superiores à DCT. Isto vem confirmar o fato de que as wavelets têm uma grande capacidade para representar bem picos e descontinuidades. Deve-se observar que abaixo da amplitude 10 a wavelet tem uma maior dificuldade de baixar o valor do erro máximo absoluto, como esperado, pois estará atuando sobre regiões dominadas pelo ruído.

Seguindo o roteiro usado no exemplo da seção anterior, o sinal foi transformado utilizando-se a wavelet de melhor rendimento em relação ao critério de energia - a wavelet B-Spline ordem 2/6 - e codificado utilizando-se uma quantização uniforme de 8 bits. Retendo-se somente os 200 maiores coeficientes, a quantidade de energia retida, de acordo com a Figura 70, alcança mais de 80% da energia do sinal original. Este valor indica uma relação sinal/ruído (SNR) baixa de valor 5, que é razoável tendo em vista o gráfico da Figura 68.

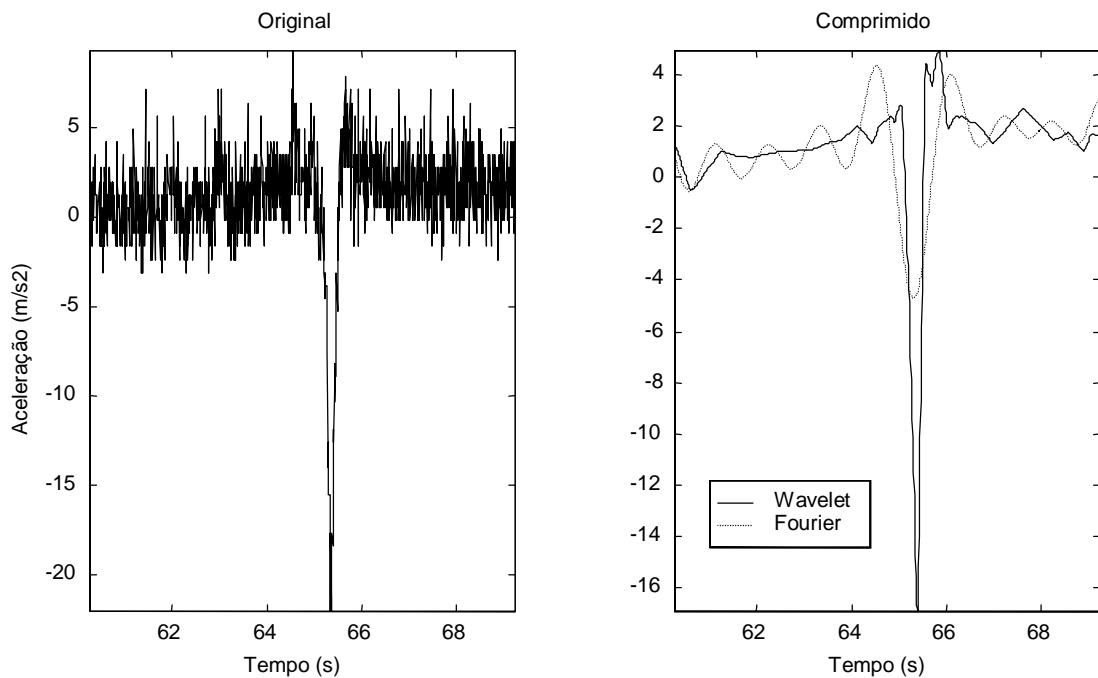
O número de bits do vetor comprimido resultou em  $200 \times (8+14) + 64 = 4464$  bits dos  $15400 \times 64 = 985600$  bits originais, o que representa uma compressão de 99,54%, ou seja, uma excelente taxa de 220:1. Mesmo assim, se fosse utilizado um codificador mais eficiente poderia-se pelo menos dobrar este resultado.

O sinal foi então restaurado realizando-se as operações inversas e o resultado é mostrado na Figura 72. Comparando-se a Figura 68 com a Figura 72 percebe-se que a compressão, além de diminuir o volume dos dados em 220 vezes, também removeu o ruído do sinal deixando intatos somente os eventos significativos. Isto pode ser visto mais detalhadamente na Figura 73 e na Figura 74.

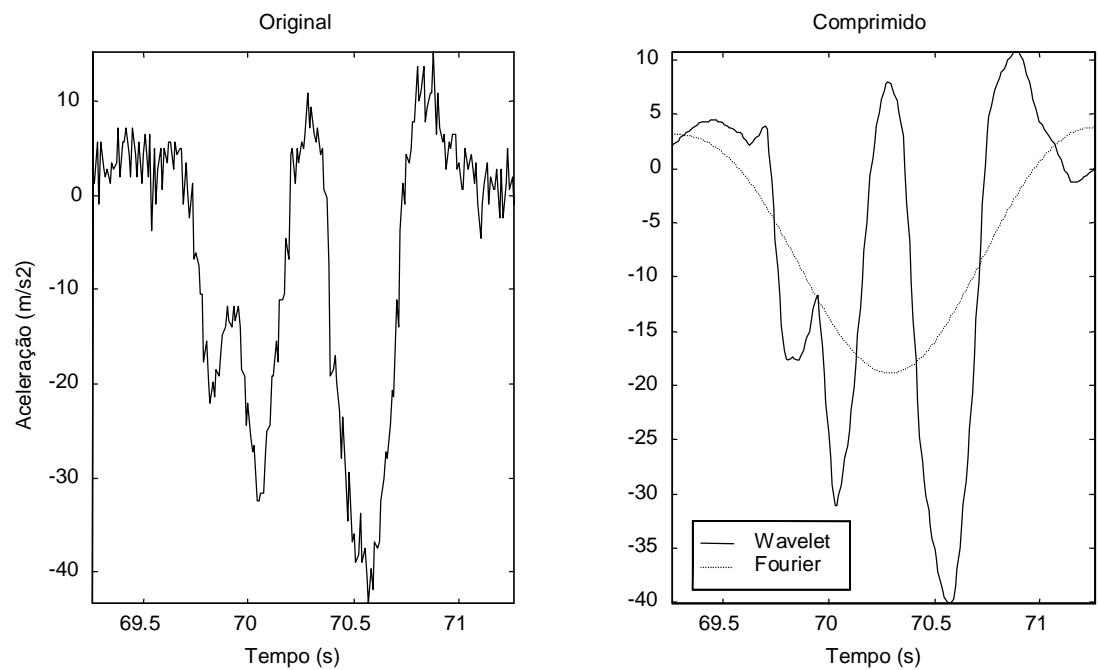


**Figura 72 Reconstituição do sinal original através dos coeficientes comprimidos**

Nestas figuras foram plotados trechos do sinal original e do sinal comprimido com wavelets e, alternativamente, com a transformada de Fourier (DCT). Para compressão com Fourier, somente os 200 primeiros coeficientes foram mantidos.



**Figura 73 Detalhe (1) do efeito da compressão sobre o sinal de tráfego**



**Figura 74 Detalhe (2) do efeito da compressão sobre o sinal de tráfego**

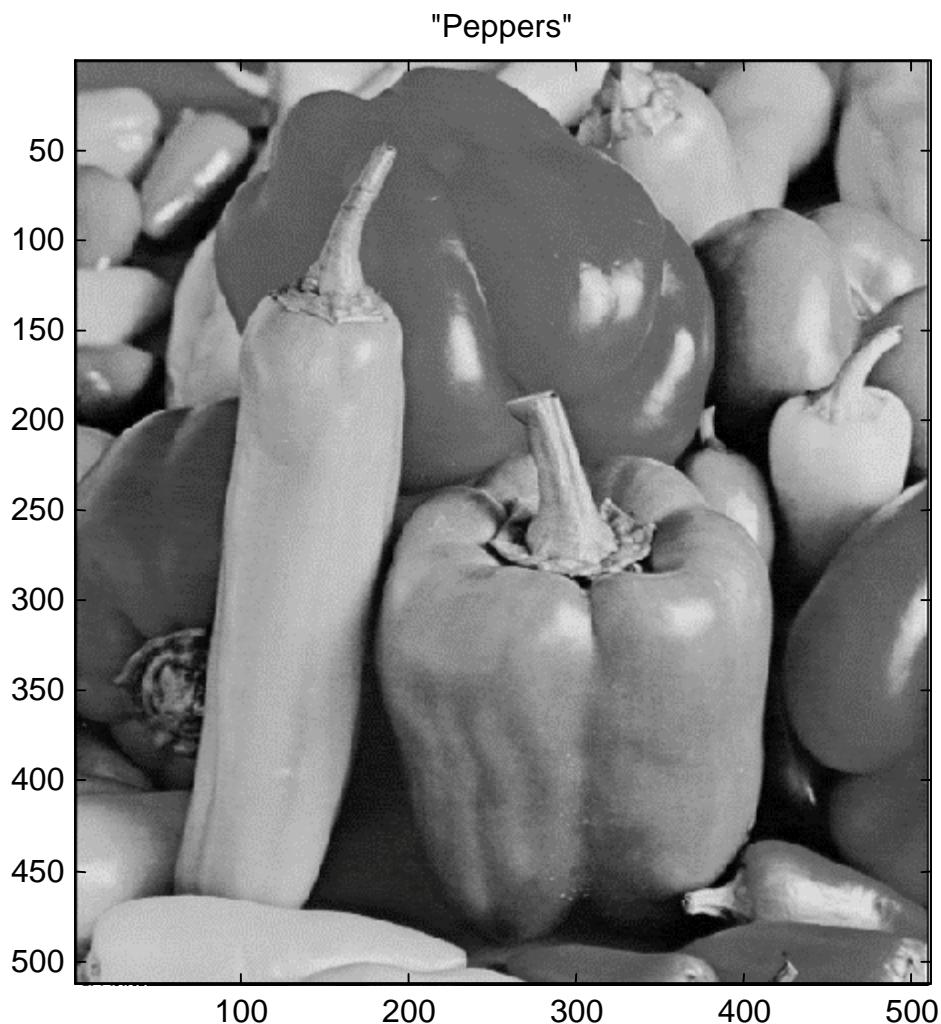
Da comparação com Fourier, notam-se duas características principais das wavelets: as descontinuidades são melhor representadas e o valor dos picos não é significativamente alterado.

#### ***IV.4 Transformadas multidimensionais***

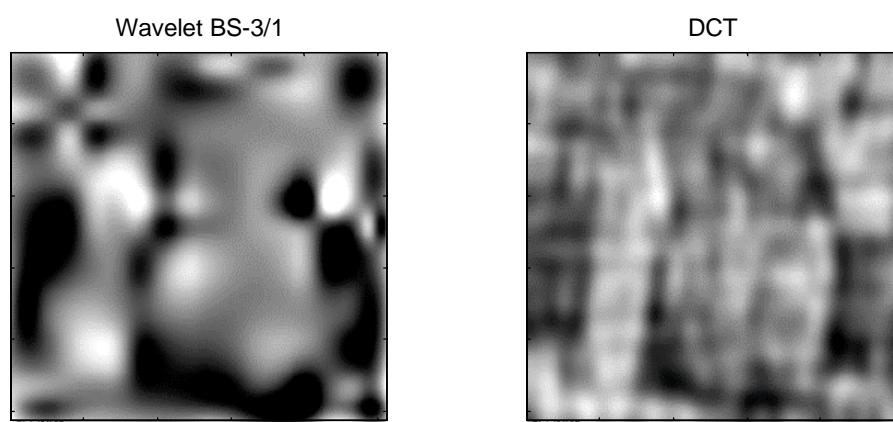
A transformada wavelet pode ser facilmente adaptada para domínios 2D ou n-D através de um produto tensorial ou de uma forma não padrão como pode ser visto em (BEYLIN, 1993), (BEYLIN et al, 1991) ou (WICKERHAUSER, 1994). Computacionalmente o efeito do primeiro procedimento é tão somente fazer-se uma transformada cruzada, ou seja, particularizando para o caso 2D, realizar-se primeiramente a transformada wavelet 1D em cada coluna e, sobre a matriz resultante, realizar-se a transformada 1D por linhas.

Tradicionalmente em compressão de imagens é utilizada a transformada cosseno de Fourier (DCT). Para que uma comparação seja estabelecida, será feita a compressão de uma imagem padrão de tamanho 512x512 pontos, conhecida como “peppers” ou “pimentões” (Figura 75), utilizada como benchmark de algoritmos de compressão de imagens por ter uma boa variedade de planos suaves, contrastes e detalhes. As duas transformadas, wavelet e DCT, serão utilizadas. Deve-se frisar que nenhum artifício adicional será realizado visando-se aumentar a qualidade de qualquer método pois o objetivo desta comparação é estabelecer-se tão somente as facilidades “naturais” oferecidas pelas duas transformadas.

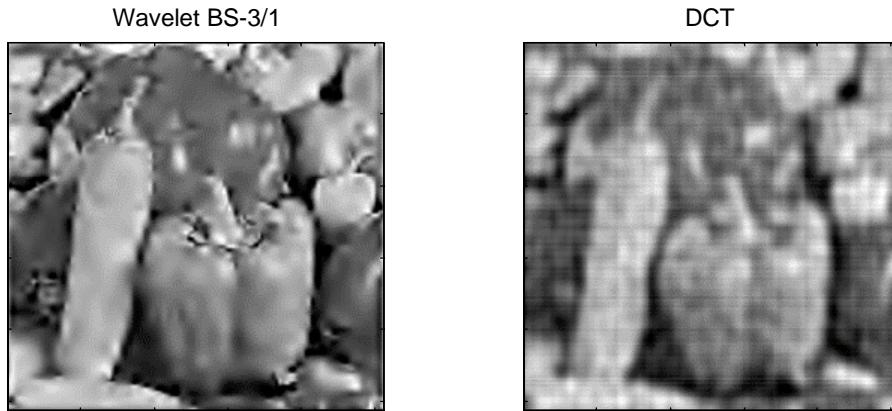
A wavelet utilizada foi a B-Spline 3/1 (DAUBECHIES, 1988) sem motivo especial para sua escolha a não ser a suavidade, ou seja, não foram realizados testes com outras famílias para verificar-se qual a de melhor rendimento, embora existam outras famílias de wavelets especialmente criadas para compressão de imagens (MANDAL *et al*, 1993). A transformada DCT bidimensional é realizada por um produto tensorial, isto é, inicialmente a DCT unidimensional por colunas seguida pela DCT das linhas resultantes. Tendo sido feitas as transformadas, foram retidos somente os coeficientes de maior módulo de cada conjunto, sendo os demais coeficientes zerados. As transformadas inversas de cada imagem finalmente foram calculadas sendo obtidas as imagens, em sequência, da Figura 76 à Figura 79.



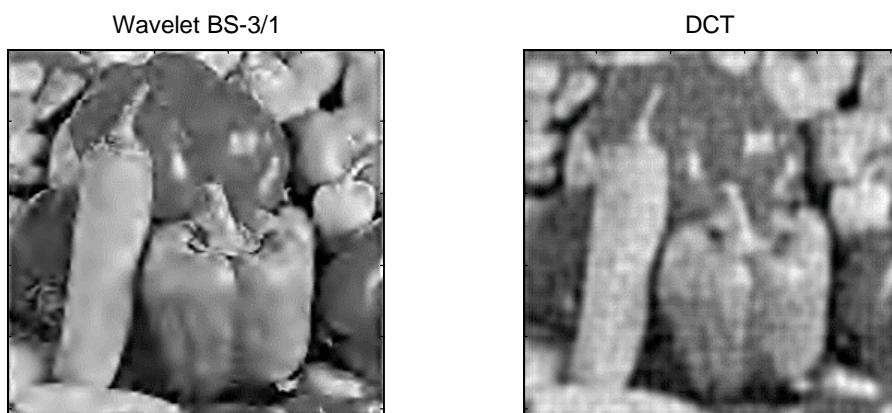
**Figura 75** Imagem padrão “peppers”, grayscale, 512x512 pontos



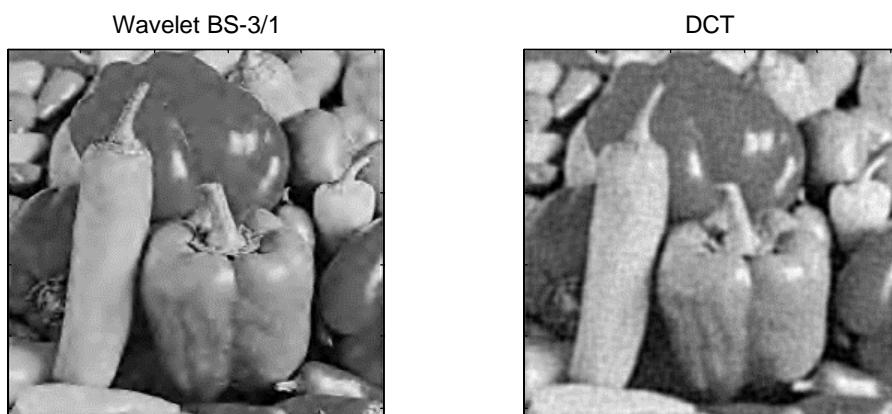
**Figura 76** Compressão realizada retendo-se os 125 coeficientes de maior módulo



**Figura 77 Compressão realizada retendo-se os 500 coeficientes de maior módulo**



**Figura 78 Compressão realizada retendo-se os 1000 coeficientes de maior módulo**

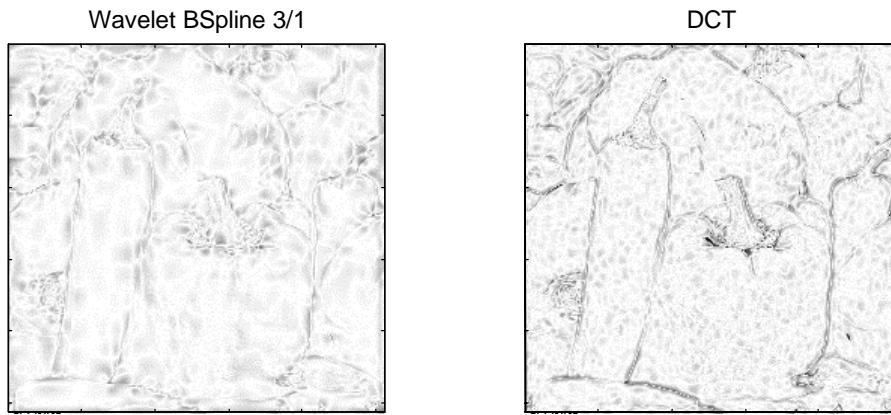


**Figura 79 Compressão realizada retendo-se os 3000 coeficientes de maior módulo**

Pode-se observar que a DCT consegue inicialmente (Figura 76) transmitir melhor o conteúdo da fotografia mas parece ter grande dificuldade em acrescentar detalhes de acordo com que mais coeficientes são retidos. Isto pode ser notado principalmente pelos contornos dos pimentões, especialmente na representação dos talos.

A Figura 77 mostra bem a organização multi-níveis das wavelets. Pode-se reparar que, em comparação com a respectiva imagem da DCT, o interior dos pimentões estão suaves como realmente é na figura original. Na DCT este interior apresenta-se

texturizado como se tivesse sido marcado por um tecido rústico, grosso. Isto se deve ao fato de que as wavelets representam estas áreas suaves apenas com coeficientes dos níveis mais altos, isto é, de baixa frequência. Já o talo do pimentão mais central apresenta uma boa definição, ao contrário da imagem DCT correspondente, pois a transformada wavelet concentra nestes locais maior quantidade de coeficientes de baixa escala, ou seja, de alta frequência. Esta propriedade de poder-se concentrar coeficientes de baixa ou alta frequência em diferentes locais de acordo com a necessidade é a principal responsável pela qualidade da transformada wavelet. A transformada de Fourier (DCT) possui apenas a propriedade de escolha em frequência, não a de localização. Isto pode ser melhor compreendido com o cálculo do resíduo entre a imagem comprimida e a original, mostrada na Figura 80.



**Figura 80 Resíduo da compressão após retenção de 1500 coeficientes**

Na Figura 80 nota-se que a maior parte da informação visual foi retida pela transformada wavelet restando apenas algumas manchas em áreas esparsas. Na transformada DCT, no entanto, vê-se que informação visual importante ainda se encontra no resíduo. Esta informação consta principalmente dos contornos que podem ainda ser claramente identificados. Isto é devido ao fato de que nos contornos os coeficientes de alta frequência adquirem valores altos (em módulo) e são, portanto, retidos pelo algoritmo de corte. Já a DCT precisa de um grande número de coeficientes para representar uma súbita variação cromática, dispersando a energia contida na descontinuidade entre vários coeficientes que têm, por consequência, valores mais baixos (em módulo). Por este motivo estes coeficientes não são retidos pelo algoritmo de compressão e são descartados, carreando a informação do contorno para o resíduo.

A Figura 78 mostra bem uma consequência secundária da localização dos coeficientes: a imagem wavelet possui mais contraste, isto é, os picos da imagem são melhor

representados. Olhando-se de longe as duas figuras nota-se que a imagem DCT correspondente é mais acinzentada que a imagem wavelet pois os picos tendem a ser pior representados, transformando tons claros e escuros em tons médios, cinzas.

Última da sequência, a Figura 79 mostra como a DCT possui dificuldade em livrar-se do efeito de texturização enquanto a imagem wavelet já está em qualidade bem próxima à imagem original. Deve-se notar, principalmente, a suavidade da representação da graduação dos tons do pimentão que está na posição vertical. Enquanto na imagem wavelet este apresenta-se suave, os tons médios variando suavemente conforme a imagem original, na DCT este apresenta-se com um aspecto encaroçado.

Deve-se recordar que uma transformação isométrica deve preservar duas propriedades do espaço original, dimensionalidade e energia. As transformadas wavelet (quando a família é ortogonal) e DCT preservam ambas as propriedades. Preservar energia significa que o somatório dos quadrados dos módulos dos coeficientes deve ser igual em ambos os espaços. Preservar dimensão implica que o número de coeficientes existentes em ambos os espaços deve ser exatamente igual. A imagem original deste exemplo possui 512 por 512 pontos, o que resulta em um total de 262144 pontos. Os espaços transformados, por serem isometrias do espaço original, deverão também possuir 262144 coeficientes, o que efetivamente se verifica. Portanto as taxas de compressão iniciais (sem levar-se em conta a posterior quantização e codificação dos coeficientes) para as imagens transformadas/reconstruídas são, respectivamente, 2097:1, 525:1, 262:1 e 87:1.

Este efeito de compressão das imagens pode ser aproveitado em outras áreas como análise numérica. Sendo imagens em verdade, para efeito de armazenamento computacional, matrizes, a analogia se aplica com facilidade. Neste caso, existem famílias especialmente criadas com o intuito de manipular-se matrizes resultantes de análises numéricas (OJANEN, 1998, OJANEN, 1991).

No método dos elementos de contorno, por exemplo, as matrizes de rigidez são cheias, ao contrário do que acontece com o método dos elementos finitos onde as matrizes são naturalmente esparsas. Sendo o tamanho destas proporcional ao quadrado do número de graus de liberdade do modelo, sua compressão tem um grande impacto sobre o tamanho do maior modelo possível de ser analisado por limitações de memória. Este é o assunto discutido no capítulo V.

# V COMPRESSÃO DE MATRIZES RESULTANTES DA DISCRETIZAÇÃO DE OPERADORES INTEGRAIS

## V.1 *Introdução*

Neste capítulo será apresentada a teoria necessária para a compressão de matrizes resultantes do método dos elementos de contorno utilizando wavelets. A idéia básica é que as matrizes resultantes do Método dos Elementos de Contorno (*Boundary Element Method* ou *BEM*) possuem geralmente muitas regiões suaves que podem ser comprimidas utilizando a transformada wavelet, inserindo um nível de erro limitado – arbitrariamente escolhido – no resultado final.

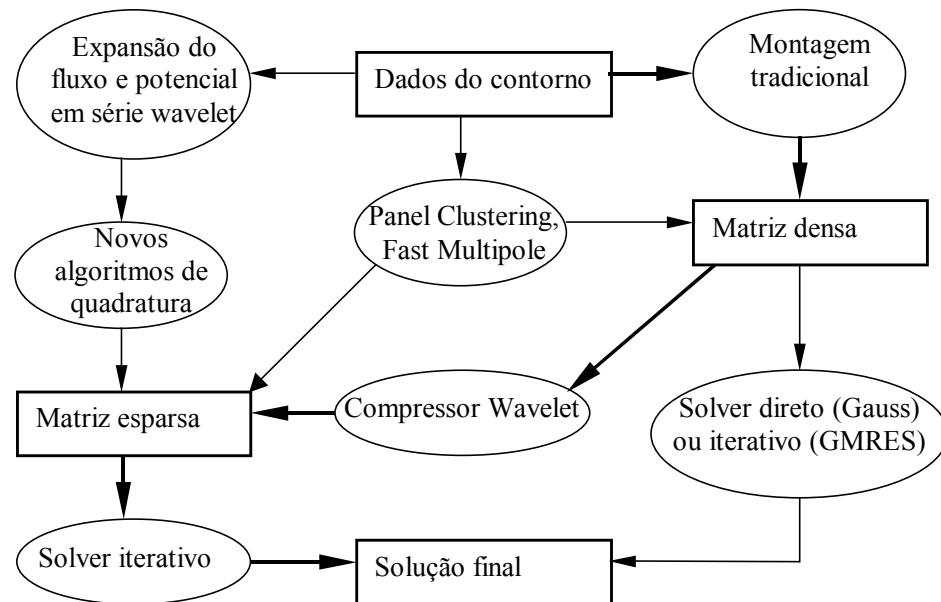
O método dos elementos de contorno, quando aplicado à solução de equações integrais, produz matrizes não-simétricas, densas, que requerem grande esforço computacional quando são utilizados métodos diretos como o método da eliminação de Gauss – de ordem  $O(N^3)$  para sistemas com  $N$  graus de liberdade. Para reduzir este esforço computacional envolvido na obtenção da solução, vários trabalhos têm sido publicados sobre métodos iterativos baseados nos subespaços de Krylov como o GMRES (SAAD, SCHULTZ, 1986, BARRA et al., 1994, KANE et al, 1991, MANSUR et al, 1992) e respectivos pré-condicionadores (PRASAD et al, 1994). A despeito do sucesso destes esforços, o tempo necessário para obter soluções acuradas permanece ainda da ordem  $O(N^2)$ .

Analisando-se estas matrizes verifica-se que existem grandes áreas suaves, resultantes da conhecida super-representação da geometria nos elementos distantes. Existe um grande esforço, no presente momento, no sentido do desenvolvimento de *solvers* rápidos para o método, de ordem  $O(N \log N)$ , baseados na idéia da redução de tal redundância através da utilização de técnicas como *panel clustering* (HACKBUSH, 1989) ou *fast multipole expansions* (ROKHLIN, 1983, NABORS et al, 1994). Apesar de muito eficientes, estas técnicas requerem que seja desenvolvida uma formulação matemática completamente nova e também os respectivos programas. Uma outra forma de reduzir tal redundância é a utilização da transformada wavelet, cuja maior vantagem

é que ela pode ser diretamente usada como uma “caixa-preta” a ser simplesmente inserida nas formulações pré-existentes de elementos de contorno.

A aplicação de técnicas de compressão baseadas na transformada wavelet para a solução de sistemas de equações oriundas da discretização de operadores integrais está ainda em sua tenra infância e somente problemas simples da teoria do potencial têm sido resolvidos com resultados satisfatoriamente documentados (BEYLKIN et al, 1991, BOND, VAVASIS, 1994, GONZALEZ et al, 2000). Algoritmos robustos são necessários para que a técnica de compressão com wavelets seja utilizada em problemas na prática da engenharia.

As duas formas básicas de inserção da transformada wavelet ao BEM estão esquematicamente ilustradas na Figura 81 na qual são mostrados a expansão em série wavelet do potencial e fluxo ( $1^{\text{a}}$  forma) e compressão direta da matriz ( $2^{\text{a}}$  forma).



**Figura 81** Esquema dos principais *solvers* rápidos utilizados em conjunto com o método dos elementos de contorno

A primeira forma é teoricamente implementada nos estágios mais recentes de desenvolvimento, antes da introdução das aproximações numéricas. Se a técnica de colocação pontual for utilizado então é produzida uma compressão unidimensional equivalente e elegantes resultados numéricos são obtidos (BOND, VAVASIS, 1994). Por outro lado, se o método de Galerkin é utilizado (DAHMEN, MICCHELLI, 1993, LATTO et al, 1992, PETERSDORFF, SCHWAB, 1997) então uma compressão bidimensional – muito mais eficiente – é produzida, aumentando a taxa de compressão várias vezes devida à dupla eliminação da redundância.

A expansão em série das variáveis da solução (potencial e fluxo) permite evitar o cálculo da matriz – de ordem  $O(N^2)$  – reduzindo o tempo de computação para ordem  $O(N \log N)$ . A desvantagem de tal forma é que esta requer o desenvolvimento de uma formulação numérica totalmente nova desde que as matrizes dos elementos são calculadas agora no espaço wavelet, exigindo que técnicas de quadratura especializadas na transformada wavelet para representá-las com relativa acurácia. Desta forma os códigos atualmente existentes de elementos de contorno não podem ser adaptados a esta técnica.

A segunda forma, a qual é o principal objeto de estudo deste trabalho, pode ser aplicada como uma “caixa-preta” – isto é, sem substancial modificação, somente inserção de código – a programas pré-existentes. A principal razão para isto é que as técnicas de compressão wavelet são aplicadas às matrizes do sistema logo antes da efetiva solução do sistema de equações. Resumidamente, o algoritmo transforma as matrizes densas produzindo novas matrizes esparsas, equivalentes à original. Desta forma toda a teoria previamente utilizada para elementos de contorno pode ainda ser utilizada desde que a compressão wavelet seja aplicada diretamente às matrizes resultantes da discretização original. A principal desvantagem é que as matrizes densas devem ainda ser montadas.

A originalidade da compressão desenvolvida neste trabalho baseia-se principalmente na não-compressão da matriz do sistema, como feito em trabalhos anteriores, mas a compressão separada das matrizes  $\mathbf{G}$  e  $\mathbf{H}$ . A principal vantagem dessa metodologia é que estas matrizes dependem unicamente da geometria do problema mas a matriz do sistema depende das condições de contorno aplicadas ao modelo. Ao evitar-se a montagem da matriz do sistema através de um método denominado “montagem virtual”, tais matrizes não precisam ser recalculadas ao mudarem-se as condições de contorno, o que fornece uma grande flexibilidade para lidar-se com algoritmos de otimização que alterem várias vezes as condições de contorno utilizando a mesma geometria. Esta técnica também torna quase trivial lidar com condições de contorno não-lineares que aparecem, por exemplo, em problemas como proteção catódica.

A implementação destes algoritmos de compressão requer o cálculo completo, embora em uma única vez, das matrizes  $\mathbf{H}$  e  $\mathbf{G}$  antes da compressão ser aplicada, o que significa que a disponibilidade de memória é um fator crítico. A quantidade de memória requerida por estas matrizes aumenta numa relação quadrática com o aumento do número de graus de liberdade do modelo. GONZALEZ et al, 2000, apresentaram uma

solução para este problema onde uma versão paralela da transformada wavelet é introduzida. A melhor propriedade desta abordagem é que, como a forma em *lifting* é usada, a transformada wavelet não precisa necessariamente trabalhar em blocos de tamanho potência de dois como a FWT padrão. Apesar de que paralelização pode salvar tempo – e um tratamento paralelo será apresentado adiante neste trabalho – nesta abordagem particular o tamanho do problema ainda está limitado pela soma total da memória contida no cluster, isto é, não é possível aumentar-se arbitrariamente o tamanho do problema.

Uma solução efetiva para lidar-se com problemas de qualquer tamanho independentemente da quantidade de memória disponível no computador (ou no cluster), deste ponto em diante denominada a *Transformada Wavelet em Blocos – Block Wavelet Transform* ou BWT – foi desenvolvida para esta proposta pela compressão particionada da matriz cheia. A despeito de compartilhar algumas idéias básicas com outros algoritmos wavelet de mesmo nome utilizados em compressão de imagens como em CETIN et al, 1993 e HUH et al, 1995, a BWT é baseada num desenvolvimento teórico completamente diverso e original. A idéia principal por trás da BWT é comprimir a matriz por blocos de tipicamente 4096 elementos, um por vez, exigindo menos que 300 Mb de memória total disponível para calcular problemas de tamanho arbitrário.

Neste sentido foi desenvolvida uma formulação teórica baseada na criação de matrizes denominadas *Matrizes Localizadoras-Extratoras* (LE), similar às matrizes de adjacência em elementos finitos, que permite a criação do *solver* por blocos. Esta formulação foi implementada numericamente fornecendo excelentes resultados.

## **V.2 Introdução da transformada wavelet ao método dos elementos de contorno**

O Método dos Elementos de Contorno (BEM), em sua formulação direta, gera o seguinte sistema de equações quando utiliza-se a técnica padrão de colocação pontual:

$$\mathbf{H}\mathbf{u} = \mathbf{G}\mathbf{q} \quad (78)$$

Nesta equação os coeficientes das matrizes  $\mathbf{G}$  e  $\mathbf{H}$  resultam da integração da solução fundamental e sua derivada normal, respectivamente, multiplicadas pela função de interpolação ao longo do elemento de contorno.  $\mathbf{u}$  e  $\mathbf{q}$  são vetores contendo valores

nodais da grandeza estudada e sua derivada na direção normal. Em problemas de transferência de calor, estas grandezas são a temperatura e fluxo de calor, respectivamente. Para elasticidade, deslocamentos e tensões e para eletricidade, potencial e fluxo de corrente. Tradicionalmente, quando tratando-se do método em geral e não de um problema em particular, adota-se a nomenclatura correspondente ao problema de transferência de calor, isto é,  $u$  é denominado potencial e  $q$ , fluxo.

Para um melhor entendimento do significado da redução da redundância das matrizes geradas pelo método dos elementos de contorno será apresentado um exemplo simples. O exemplo escolhido é um problema estático de transmissão de calor no qual uma placa retangular é submetida a fluxos conhecidos nos lados inferior, direito e superior e a uma temperatura conhecida no lado esquerdo. Este exemplo será analisado com mais detalhes adiante ainda neste capítulo e, por ora, será verificada somente a suavidade das matrizes resultantes.

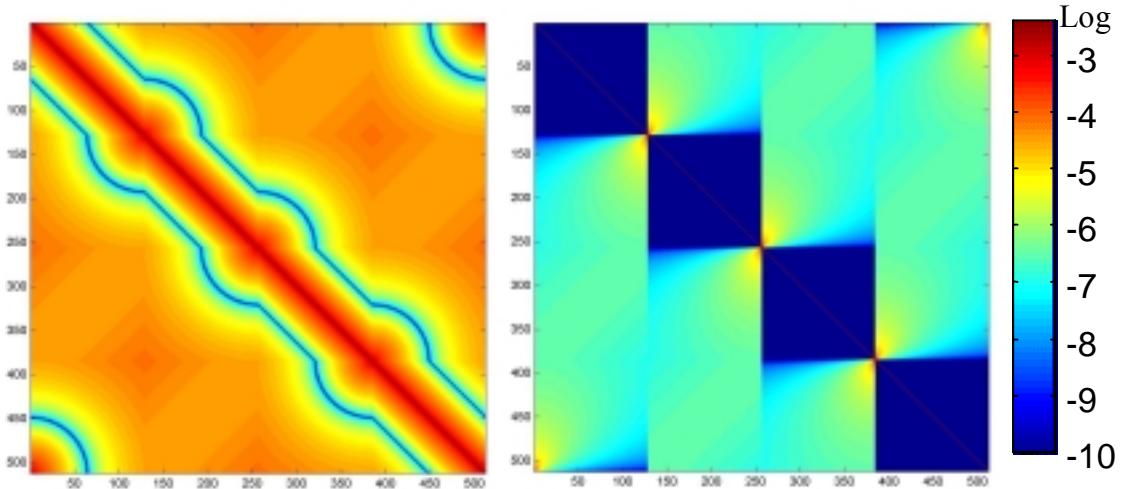
As matrizes resultantes  $G$  e  $H$  são mostradas na Figura 82, onde as cores correspondem a uma escala logarítmica mostrada à direita – os valores mostrados na escala são o logaritmo do módulo do valor dos respectivos coeficientes. Verifica-se que ambas possuem extensas regiões suaves combinadas com picos localizados.

A idéia principal dos tradicionais compressores com perda (*lossy compressors*) é representar tais regiões através de uma série de Fourier. Sendo estas regiões suaves, apenas poucos coeficientes são necessários para representá-las com um erro desprezível. Desafortunadamente, esta técnica falha quando existem descontinuidades fortes na imagem ou sinal sendo comprimidos. Os picos agudos da matriz, por exemplo, necessitariam tantos coeficientes para serem representados que a taxa final de compressão seria mínima. Por este motivo é que transformadas baseadas em Fourier como a *Discrete Cosine Transform* (DCT) precisam de técnicas de recorte por janelas (OPPENHEIM, SCHAFER, 1992) para poderem fornecer compressão efetiva de dados – explicações detalhadas a respeito são fornecidas em WALLACE, 1991, LE GALL, 1991 e CARPENTER, MUMFORD, 1992. Estas técnicas de recorte carreiam vários efeitos colaterais indesejados que devem ser devidamente tratados – aumentando a complexidade do algoritmo.

Ao substituir a transformada de Fourier pela transformada wavelet este problema é quase totalmente eliminado devido à característica hierárquica desta última, o que permite a compressão das regiões suaves utilizando-se coeficientes de baixa frequência

(hierarquia alta) e os picos utilizando-se coeficientes localizados de alta frequência (hierarquia baixa). Este processo seletivo é inerente à transformada wavelet de tal forma que, para uma extensa classe de problemas, somente uma pequena fração dos coeficientes transformados possuem valores significativos podendo os restantes serem eliminados com um erro desprezível resultante.

Uma função de Heaviside, difícil de ser representada pela transformada de Fourier, no sentido de que a maioria dos coeficientes é significante, pode ser representada com apenas uns poucos coeficientes wavelet – alguns de baixa-frequência localizados na parte positiva da função de valor unitário e constante e alguns coeficientes de alta frequência localizados no salto.



**Figura 82 Matrizes  $G$  e  $H$  geradas a partir do problema apresentado na Figura 99.  
Valores dos elementos são mostrados em escala logarítmica.**

Para compreender melhor a transformada wavelet e como ela pode ser inserida dentro do contexto do método dos elementos de contorno, faz-se necessária uma breve incursão por suas propriedades algébricas.

O algoritmo para a transformada wavelet rápida (Fast Wavelet Transform ou FWT) (DAUBECHIES, 1992, CHUI, 1992), em sua forma mais simples, transforma um vetor  $v$  de tamanho  $N$ , sendo  $N$  potência de dois, e produz outro vetor de coeficientes  $d$  do mesmo tamanho. Como tantos outros operadores, como a transformada de Fourier por exemplo (COOLEY, TUKEY, 1965), a FWT pode ser representada por uma matriz quadrada  $W$  tal que

$$d = Wv \quad (79)$$

onde  $W$  é uma matriz esparsa cujos elementos são compostos por translações de um vetor especial chamado máscara (*mask*), particular a cada família wavelet – existem

várias famílias, cada uma com suas propriedades específicas. Uma descrição de várias famílias com o desenvolvimento de suas propriedades pode ser vista em DAUBECHIES (1992). Se a família é ortogonal então a matriz  $\mathbf{W}$  transposta é idêntica a sua inversa, podendo-se então escrever

$$\mathbf{W}^T \mathbf{W} = \mathbf{I} \quad (80)$$

A transformada unidimensional – expressa na equação (79) – pode também ser aplicada a matrizes, caso no qual esta operação escreve-se

$$\tilde{\mathbf{A}}_c = \mathbf{W}\mathbf{A} \quad (81)$$

quando a matriz  $\mathbf{A}$  é transformada por colunas e

$$\tilde{\mathbf{A}}_r = \mathbf{A}\mathbf{W}^T \quad (82)$$

quando a matriz  $\mathbf{A}$  é transformada por linhas.

BOND, VAVASIS, 1994 apresentaram uma formulação baseada na transformada unidimensional (79), aplicada a sistemas de equações obtidos a partir do método de colocação. Nesta formulação o potencial e o fluxo são expandidos em uma série wavelet, o que é equivalente a pré-multiplicar o sistema padrão de equações (78) pela matriz wavelet  $\mathbf{W}$ , produzindo

$$\mathbf{W}\mathbf{H}\mathbf{u} = \mathbf{W}\mathbf{G}\mathbf{q} \quad (83)$$

Esta abordagem permite a derivação de resultados teóricos importantes como também de *upper-bounds* mas a taxa de compressão resultante é bem modesta. A razão para isto é que ambas as matrizes  $\mathbf{H}$  e  $\mathbf{G}$  possuem uma redundância espacial em ambas as direções – linhas e colunas. Aplicando-se a transformada wavelet duas vezes, resultando na denominada transformada bi-dimensional, atinge-se o objetivo de eliminar a redundância em ambas direções. Isto é feito aplicando-se a transformada wavelet primeiro às colunas e então às linhas, uma operação que pode ser algebraicamente expressa pela equação

$$\tilde{\mathbf{A}} = \mathbf{W}\mathbf{A}\mathbf{W}^T \quad (84)$$

onde  $\mathbf{A}$  é uma matriz quadrada arbitrária. Inserindo-se a identidade (80) após cada matriz BEM em (83), o seguinte sistema de equações é obtido

$$\tilde{\mathbf{H}}\mathbf{W}\{\mathbf{u}\} = \tilde{\mathbf{G}}\mathbf{W}\{\mathbf{q}\} \quad (85)$$

As novas matrizes  $\tilde{\mathbf{H}} = \mathbf{W}\mathbf{H}\mathbf{W}^T$  e  $\tilde{\mathbf{G}} = \mathbf{W}\mathbf{G}\mathbf{W}^T$  muito frequentemente possuem somente uns poucos coeficientes significantes e podem ser efetivamente comprimidas através de uma operação de seleção denominada *thresholding*.

A operação de *threshold* é definida por, dado uma constante positiva e real  $\alpha$ ,

$$\text{threshold}(x) = \begin{cases} x & \text{se } |x| \geq \alpha \\ 0 & \text{caso contrário} \end{cases} \quad (86)$$

Desta forma, novas matrizes esparsas  $\tilde{\mathbf{H}}^s = \text{threshold}(\tilde{\mathbf{H}})$  e  $\tilde{\mathbf{G}}^s = \text{threshold}(\tilde{\mathbf{G}})$  são obtidas, tendo-se o cuidado de notar que a operação de *threshold* é executada elemento-por-elemento da matriz.

Esta abordagem representa uma forma significantemente diferente de comprimir matrizes BEM se comparado com trabalhos anteriores (BEYLKIN et al, 1991, BOND, VAVASIS, 1994) onde a matriz do sistema  $\mathbf{A}$ , formada pelo procedimento padrão de troca de colunas entre as matrizes  $\mathbf{H}$  e  $\mathbf{G}$ , é inicialmente formada e então comprimida. A grande dificuldade que aparece neste procedimento é que não existe liberdade de trocar as condições de contorno após a matriz ter sido comprimida. Este é um requerimento essencial para aplicações envolvendo otimização ou casos múltiplos de carregamento.

Outra vantagem do novo processo é que valores de *threshold* distintos  $-\alpha_H$  e  $\alpha_G$  - podem ser aplicados a cada matriz independentemente. Isto não é possível se a matriz do sistema  $\mathbf{A}$  é gerada antes e informação importante pode ser perdida se uma das matrizes é dominante em relação à outra, como acontece frequentemente em problemas de elasticidade (BREBBIA et al, 1984). Portanto a escolha correta dos valores de *threshold* pode minimizar o erro final inserido na solução e, desta forma, também maximizar as taxas de compressão.

### **V.3 O Método da Montagem Virtual (MMV) da matriz do sistema**

Seguindo a forma tradicional de solução do método dos elementos de contorno, o próximo passo após a integração dos elementos, gerando-se as matrizes  $\mathbf{G}$  e  $\mathbf{H}$ , consiste em combiná-las pela técnica de troca de colunas de forma a gerar a forma canônica usual  $\mathbf{A} \mathbf{x} = \mathbf{b}$  para solução com Gauss ou com um *solver* iterativo.

A forma de trocar-se as colunas depende da condição de contorno prescrita ao grau de liberdade correspondente à respectiva equação. Se, por exemplo, uma condição de contorno essencial (potencial  $u=u_0$ ) é prescrita em um determinado grau de liberdade, a coluna correspondente à tal incógnita na matriz  $G$  é copiada para a matriz do sistema  $A$  e o resultado da multiplicação entre a matriz  $H$  e os valores prescritos é somada ao vetor de força  $b$ .

Este passo, no entanto, é dificultado pelo fato de que as matrizes, a este tempo, já estarão comprimidas impossibilitando assim continuar utilizando-se a equação (85) diretamente. Desta forma é necessário criar uma alternativa para montagem da matriz do sistema *virtualmente*, isto é, sem fisicamente alterar as estruturas matriciais pré-existentes.

A forma mais flexível de fazê-lo é através da aplicação das condições de contorno que passam a ser introduzidas somente dentro do ciclo iterativo. À primeira vista redundante e dispendioso, o processo de aplicar as condições de contorno a cada iteração introduz enorme flexibilidade ao *solver*, como será destacado adiante.

O processo utilizado para aplicar as condições de contorno consiste em produzir dois vetores  $u$  e  $q$  a partir de um único vetor de incógnitas  $x$ , processo este que passa-se a denominar “montagem virtual”. A premissa para utilizá-la é que pode-se sempre reescrever quaisquer condições de contorno, referentes a um grau de liberdade arbitrário “ $i$ ”, na seguinte forma:

$$\begin{aligned} u_i &= \lambda_i^H x_i + u_{0i} \\ q_i &= \lambda_i^G x_i + q_{0i} \end{aligned} \tag{87}$$

onde  $x_i$  é a incógnita correspondente ao grau de liberdade e  $\lambda_i^G$  e  $\lambda_i^H$  são operadores os quais, aplicados à incógnita  $x_i$  e somando o resultado às respectivas constantes  $u_{0i}$  e  $q_{0i}$ , produzem as soluções  $u_i$  e  $q_i$ .

O motivo para a separação da solução (variáveis  $u_i$  e  $q_i$ ) em duas partes – variável e constante – é somente devido ao critério de parada utilizado em muitos *solvers* iterativos: a relação da norma do resíduo sobre a norma do resíduo inicial, que é exatamente igual à norma do vetor de força. Como será visto adiante na equação (91), o vetor de força é formado justamente a partir dos vetores formados pelo conjunto de constantes  $u_{0i}$  e  $q_{0i}$ . Se, portanto, o vetor de força inicial é zero, esta opção para o

critério de parada é perdida. Casos comuns de condições de contorno reescritas neste formato estão listados na Tabela 5.

**Tabela 5 Condições de contorno disponíveis via técnica da montagem virtual do sistema**

Tipo	Expressão usual	Implementação com split
Potencial prescrito	$u = \bar{u}$	$\lambda^H x = 0, u_0 = \bar{u}$ $\lambda^G x = x, q_0 = 0$
Fluxo prescrito	$q = \bar{q}$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = 0, q_0 = \bar{q}$
Dependência linear	$q = a + bu$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = bx, q_0 = a$
Dependência quadrática	$q = a + bu + cu^2$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = cx^2 + bx, q_0 = a$
Não-linear genérica	$q = f(u)$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = f(x), q_0 = 0$

Substituindo a representação por operadores (87) dentro da equação integral original (78) produz-se

$$(\mathbf{H}X_H - \mathbf{G}X_G)x = -\mathbf{H}\mathbf{u}_0 + \mathbf{G}\mathbf{q}_0 \quad (88)$$

onde  $\mathbf{u}_0$  e  $\mathbf{q}_0$  são vetores resultantes da contribuição das respectivas constantes em (87).  $X_H$  e  $X_G$  são novas matrizes diagonais de operadores cujos elementos são

$$(X_H)_{i,j} = \begin{cases} \lambda_i^H & \text{se } i = j \\ 0 & \text{caso contrário} \end{cases} \quad (X_G)_{i,j} = \begin{cases} \lambda_i^G & \text{se } i = j \\ 0 & \text{caso contrário} \end{cases} \quad (89)$$

Desta forma, a nova matriz do sistema é reconhecida como função das novas matrizes

$$\mathbf{A} = \mathbf{H}X_H - \mathbf{G}X_G \quad (90)$$

e o vetor independente que naturalmente aparece no lado direito tem a forma

$$\mathbf{f} = -\mathbf{H}\mathbf{u}_0 + \mathbf{G}\mathbf{q}_0 \quad (91)$$

Como as matrizes  $X$  são na realidade matrizes de operadores e não possuem necessariamente uma forma numérica, a matriz do sistema  $A$  é melhor descrita como matriz virtual do sistema, isto é, utilizada na resolução do sistema mas nunca efetivamente montada.

Neste ponto é necessário realizar-se um parêntese para discutir – e inserir – uma técnica para lidar com partes incompressíveis das matrizes, denominadas *regiões residuais*. GONZALEZ et al, 2000, discutindo a compressibilidade das matrizes de elementos de contorno, conclui que tais matrizes frequentemente são compostas por partes extremamente compressíveis (suaves) e outras quase incompressíveis. Isto é particularmente verdadeiro quando em algumas partes do modelo existem geometrias representadas por uma malha insuficientemente refinada. Tais partes incompressíveis limitam a atuação da transformada wavelet porque introduzem erros bastante significativos na solução, desestabilizando-a. Se estas geometrias puderem ser localizadas de tal forma que os respectivos blocos sejam isolados, tais blocos podem ser salvos sem serem submetidos à compressão, livrando a transformada do trabalho inútil de comprimi-los.

Tal desacoplamento pode ser facilmente obtido através da separação das matrizes originais  $\mathbf{H}$  e  $\mathbf{G}$  em parcelas compressíveis e parcelas incompressíveis como mostrado a seguir

$$\mathbf{H} = \mathbf{H}_c + \mathbf{H}_r^s \quad \text{e} \quad \mathbf{G} = \mathbf{G}_c + \mathbf{G}_r^s \quad (92)$$

onde  $\mathbf{H}_c$  é uma matriz contendo a parte compressível da matriz  $\mathbf{H}$  original e de mesmas dimensões desta,  $\mathbf{H}_r^s$  é a parte residual também de mesmas dimensões (e armazenada sem compressão) e  $\mathbf{G}_c$  e  $\mathbf{G}_r^s$  são as respectivas parcelas da matriz  $\mathbf{G}$ . Evidentemente, espera-se que os blocos não-compressíveis sejam a exceção, não a regra e por este motivo espera-se que as correspondentes matrizes sejam esparsas, fato que é diferenciado pela notação esparsa ( $^s$ ) nas matrizes residuais.

Inserindo (92) em (88), obtém-se o seguinte sistema de equações

$$(\mathbf{H}_c + \mathbf{H}_r^s) \mathbf{X}_H - (\mathbf{G}_c + \mathbf{G}_r^s) \mathbf{X}_G \mathbf{x} = \mathbf{f} \quad (93)$$

Nesta nova forma, o sistema de equações é explicitamente dividido em uma parte compressível, onde as wavelets serão aplicadas, e uma parte não compressível, a qual será usada sem modificação.

Inserindo a identidade (80) duas vezes – antes e após as parcelas compressíveis  $\mathbf{H}_c$  e  $\mathbf{G}_c$  em (93) – e tornando esparsas as matrizes transformadas pelas wavelets como em (85), resulta no seguinte sistema de equações

$$\left( \left( \mathbf{W}^T \tilde{\mathbf{H}}_c^s \mathbf{W} + \mathbf{H}_r^s \right) \mathbf{X}_H - \left( \mathbf{W}^T \tilde{\mathbf{G}}_c^s \mathbf{W} + \mathbf{G}_r^s \right) \mathbf{X}_G \right) \mathbf{x} = \mathbf{f} \quad (94)$$

Esta forma final pode ser facilmente resolvida usando-se um *solver* iterativo porque ela requer somente a multiplicação da matriz do sistema por um vetor. Como este novo sistema é somente uma representação ligeiramente diferente da original (88), espera-se que este comporte-se da mesma forma que o sistema original também com respeito ao condicionamento do sistema. Usando uma linguagem orientada a objetos como C++ ou Java em conjunto com um *solver* iterativo como em BARRET (1994) e DONGARRA et al (1996) faz-se necessário somente criar uma nova classe de matrizes e sobrecarregar seu operador multiplicação. De fato, linguagens orientadas a objeto têm estado em voga ultimamente para aplicações em elementos de contorno devido às facilidades oferecidas para a representação de estruturas de dados complexas (BARSCH et al, 1997, LAGE, 1998).

As operações entre parênteses na equação matricial (94) devem ser efetuadas para cada multiplicação matriz-vetor  $\mathbf{b} = \mathbf{A} \cdot \mathbf{x}$  dentro do *solver* iterativo. Esta multiplicação é efetuada usando-se o seguinte algoritmo

- i.  $\mathbf{x}_h = \mathbf{X}_H \cdot \mathbf{x}$  e  $\mathbf{x}_g = \mathbf{X}_G \cdot \mathbf{x}$  (aplicação das condições de contorno)
- ii.  $\tilde{\mathbf{x}}_h = \mathbf{W} \cdot \mathbf{x}_h$  e  $\tilde{\mathbf{x}}_g = \mathbf{W} \cdot \mathbf{x}_g$  (transformada wavelet 1D)
- iii.  $\tilde{\mathbf{b}} = \tilde{\mathbf{H}}_c^s \cdot \tilde{\mathbf{x}}_h - \tilde{\mathbf{G}}_c^s \cdot \tilde{\mathbf{x}}_g$  (multiplicação matriz esparsa x vetor)
- iv.  $\mathbf{b} = \mathbf{W}^T \tilde{\mathbf{b}}$  (transformada wavelet inversa)
- v.  $resultado = \mathbf{H}_r^s \cdot \mathbf{x}_h - \mathbf{G}_r^s \cdot \mathbf{x}_g + \mathbf{b}$  (multiplicação matriz esparsa x vetor)

O conjunto de operações em (i) consiste em aplicar os operadores  $\lambda^G$  e  $\lambda^H$  a cada elemento do vetor de incógnitas  $\mathbf{x}$  para produzir as variáveis  $u$  e  $q$ , como determinado em (87). Para os casos mais simples na Tabela 5 – potencial ou fluxo prescrito – o efeito é somente a cópia de elementos do vetor de incógnitas  $\mathbf{x}$  para novos vetores de tal forma que  $\mathbf{x}_h$  conterá somente potenciais calculados (onde o fluxo é prescrito) e  $\mathbf{x}_g$  conterá somente fluxos calculados (onde o potencial é prescrito).

O segundo conjunto de operações (ii) consiste simplesmente da transformada wavelet unidimensional, uma transformada muito rápida – bastante mais rápida que a FFT –

envolvendo somente somas e multiplicações. O terceiro (iii) e o quinto (v) conjunto de operações são multiplicações esparsas dos vetores resultantes de (ii) e (iv) pelas matrizes residuais e comprimidas, respectivamente, um procedimento igualmente rápido porque somente uma pequena fração do número de elementos das matrizes originais é armazenada após compressão.

Como as condições de contorno agora são aplicadas dentro do *solver* (i), a cada iteração, estas podem ser livremente alteradas a qualquer tempo após a compressão. Isto permite que técnicas de otimização façam proveito extenso da compressão. Deve-se notar também que todas as operações envolvendo a aplicação dos operadores e as transformadas wavelets são efetuadas muito rapidamente, afetando muito pouco a performance do *solver*. No entanto a liberdade introduzida na solução do sistema compensa de longe qualquer perda de performance, mesmo que significativa.

Como o presente método baseia-se em *solvers* iterativos, uma questão importante a ser levantada é a existência de pré-condicionadores para a presente formulação. É claro que, como a equação (94) não está mais na forma padrão torna-se mais difícil extrair os dados diretamente para alimentar as rotinas de inicialização do pré-condicionador. No entanto, é sempre possível obter todos os dados necessários durante o processo de montagem. Para o caso particular do pré-condicionador diagonal, se as diagonais de ambas as matrizes  $\mathbf{G}$  and  $\mathbf{H}$  são salvas em separado, é trivial alimentar o pré-condicionador. Com este intento, se  $\mathbf{d}_G$  e  $\mathbf{d}_H$  são os vetores representando estas diagonais, então a diagonal do sistema após aplicação das condições de contorno pode ser facilmente recuperada utilizando-se a equação (90), produzindo

$$\mathbf{D}^T = \mathbf{d}_H^T \mathbf{X}_H - \mathbf{d}_G^T \mathbf{X}_G \quad (95)$$

Naturalmente, as matrizes  $\mathbf{X}_H$  e  $\mathbf{X}_G$  são matrizes de operadores na realidade e somente é possível obter uma representação numérica exata para estas matrizes se todos os operadores puderem ser expressos como um polinômio do primeiro grau. Para todos os demais casos uma aproximação tangencial pode ser obtida expandindo-se o operador em série de Taylor em torno da solução atual.

Entretanto, como já dito, para condições de contorno lineares é possível representar estes operadores exatamente por números. Como um exemplo, para a condição de contorno  $q = a + bu$  na Tabela 5, estes operadores podem ser substituídos por números

como  $\lambda^G = b$  e  $\lambda^H = 1$ , portanto permitindo o cálculo do vetor  $D$  necessário para alimentar o pré-condicionador diagonal.

Deve ser notado que esta não é uma limitação da presente técnica. Na forma tradicional não é trivial construir-se um pré-condicionador diagonal para estes casos de contorno não-lineares. De fato é comum nestes casos expandir a matriz do sistema com a ajuda de uma matriz jacobiana (BREBBIA *et al*, 1984).

Todas as aplicações numéricas realizadas neste trabalho utilizaram o pré-condicionador diagonal pois, embora não seja sofisticado, diminui significativamente o número de iterações necessárias para a convergência do *solver*.

#### **V.4 A Transformada em Blocos**

A compressão por blocos já vem sendo utilizada em processamento de imagens há algum tempo. Esta técnica, no entanto, não pode ser utilizada em aplicações numéricas pois não existe vantagem em comprimir uma matriz sem que esta possa ser utilizada no estado comprimido diretamente, sem descompressão. Isso não ocorre com imagens ou sons, que são comprimidas somente para intuito de armazenamento ou transmissão, sendo descomprimidas e visualizadas quando necessário. Para uso numérico, no entanto, é imperativo que estas matrizes nunca sejam descomprimidas pois, se forem, o problema de disponibilidade de memória/armazenagem continuaria insolúvel.

Quando *solvers* iterativos são usados, a única operação necessária de ser realizada com as matrizes é uma operação de multiplicação por um vetor – alguns *solvers* exigem também a multiplicação da matriz transposta por um vetor. A solução portanto baseia-se em multiplicar os blocos – já transformados e tornados esparsos – por um vetor dado, devendo o resultado ser igual ao obtido pela multiplicação da matriz cheia original pelo mesmo vetor.

A solução para esta tarefa é obtida por uma matriz de mapeamento conhecida no jargão do método dos elementos finitos como matriz de adjacências (CAREY, 1997, BOVA, CAREY, 2000). Este algoritmo fornece uma expansão em série da matriz cheia original na qual as funções de expansão da série são matrizes de adjacência e os “coeficientes” são os blocos propriamente ditos.

Este trabalho, como será visto adiante, avança além de esforços anteriores para definir a admissibilidade do particionamento adotado, isto é, definido-se um grid para o

particionamento da matriz global, existe um teste para definir se este particionamento é válido. Esta checagem do particionamento só é possível graças ao desenvolvimento de uma matriz de adjacências especial, denominada de ora em diante de *matriz Localizadora-Extratora*, ou matriz LE. Esta matriz será fundamental no desenvolvimento da transformada em blocos pois permitirá a introdução da transformada wavelet direta e inversa de uma forma algebricamente correta.

#### V.4.1 Propriedades fundamentais das matrizes LE

Cada matriz LE é notada pela simbologia  $L_{N,n}^i$  onde  $i$  é o termo da série,  $N$  é o tamanho da matriz cheia original e  $n$  é o tamanho do bloco. A matriz LE é intrinsecamente uma matriz identidade estendida de tal forma a possuir propriedades especiais para lidar com blocos, como ilustrado na Figura 83. Esta matriz é composta por “1” na diagonal que se inicia na linha “ $i$ ” e preenchida com zeros em todo o resto.

As matrizes LE possuem duas propriedades importantes que serão usadas extensivamente nos desenvolvimentos que se seguem:

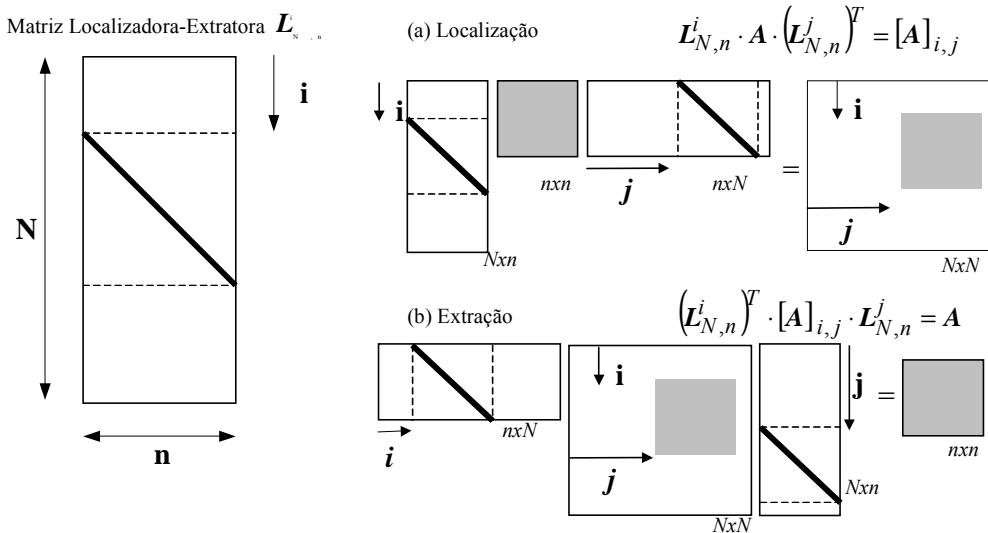
**Propriedade de Extração.** Um bloco  $A_k$  de tamanho  $n$ , localizado na posição  $(row_k, col_k)$  de uma matriz genérica  $A$  de ordem  $N$ , pode ser extraído através da pré-multiplicação da transposta da matriz LE  $L_{N,n}^{row_k}$  e pela pós-multiplicação do resultado pela matriz LE  $L_{N,n}^{col_k}$  de tal forma que

$$A_k = \left( L_{N,n}^{row_k} \right)^T A L_{N,n}^{col_k} \quad (96)$$

**Propriedade de Localização.** O bloco “ $k$ ” extraído usando a propriedade anterior pode ser restaurado à sua posição original  $(row_k, col_k)$  na matriz cheia original pela pré-multiplicação do bloco pela matriz LE  $L_{N,n}^{row_k}$  e pela pós-multiplicação do resultado pela transposta da matriz LE  $L_{N,n}^{col_k}$ . Evidentemente a matriz restaurada desta forma terá o mesmo tamanho da matriz original mas não será igual à matriz original  $A$  e, para enfatizar esta diferença, esta matriz restaurada será denotada por

$$[A_k]_{row, col} = L_{N,n}^{row_k} A_k \left( L_{N,n}^{col_k} \right)^T \quad (97)$$

Ambas as propriedades podem ser melhor compreendidas com a ajuda da Figura 83.



**Figura 83 Forma e propriedades da matriz “Localizadora-Extratora” (matriz LE)**

Desta forma, usando a propriedade (96) para decomposição e a propriedade (97) para reconstrução, a matriz quadrada  $A$  pode ser dividida em  $M$  blocos distintos  $A_k$ ,  $k = 1$  a  $M$ , cada um destes localizado em uma posição ( $row_k, col_k$ ) e tendo um tamanho  $n_k$ . Neste processo de separação a reconstrução exata da matriz original deve ser possível, o que é garantido pela propriedade de admissibilidade do particionamento utilizado para a construção em blocos, introduzido a seguir.

#### V.4.2 Admissibilidade do particionamento

Para que as matrizes LE apresentadas possam ser utilizadas é preciso que estas sejam corretamente introduzidas num esquema de particionamento da matriz original. Evidentemente, nem todas as combinações de particionamento são válidas. Se, por exemplo, em uma determinada forma de partição dois blocos compartilharem a mesma região da matriz, isto é, se dois blocos se sobrepuarem, será impossível gerar um esquema numérico que calcule a multiplicação da matriz original por um vetor. Da mesma forma, se uma determinada região da matriz não for coberta por nenhum bloco então o resultado não será igual ao esperado.

Estes requerimentos têm relação direta com a teoria dos subespaços vetoriais e podem ser sintetizados em uma única expressão denominada de ora em diante de *requisito de admissibilidade do esquema de particionamento*, definido a seguir.

**Requerimento de Admissibilidade.** O particionamento em blocos deve ser realizada de tal forma que dada uma matriz arbitrária  $A$  esta deve ser exatamente reconstruída a partir de seus blocos constituintes de tal forma que a seguinte expressão seja válida:

$$A = \sum_{k=1}^M L_{N,n_k}^{row_k} A_k (L_{N,n_k}^{col_k}) \quad (98)$$

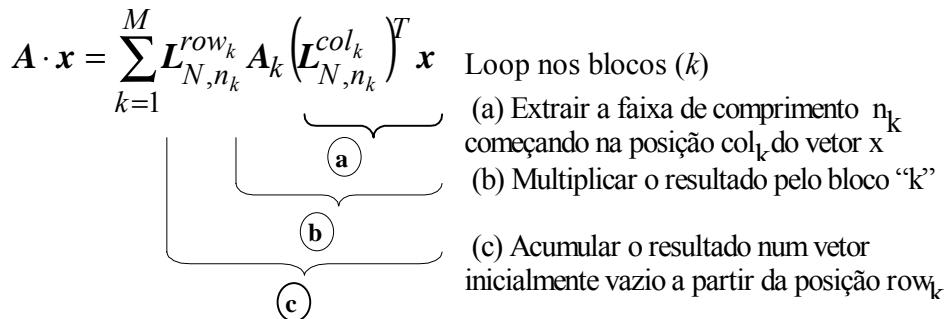
Este requerimento será a base para todos os desenvolvimentos teóricos efetuado de ora por diante. Deve-se lembrar, para completo entendimento dessa expressão, que cada bloco  $A_k$ ,  $k = 1$  a  $M$ , é quadrado de tamanho “ $n_k$ ” e está localizado em uma posição definida por ( $row_k$ ,  $col_k$ ) .

#### V.4.3 Cálculo da multiplicação matriz-vetor em blocos

O produto  $b = A \cdot x$  é calculado a partir da definição (98) através da expressão

$$b = A \cdot x = \sum_{k=1}^M L_{N,n_k}^{row_k} \cdot A_k \cdot (L_{N,n_k}^{col_k}) \cdot x \quad (99)$$

a qual é facilmente efetuada utilizando-se o algoritmo ilustrado na Figura 84



**Figura 84 Algoritmo para o cálculo da multiplicação matriz-vetor através de um esquema de particionamento em blocos**

Evidentemente o algoritmo anterior é inútil no sentido de que nenhuma compressão foi introduzida ainda na formulação. Para tanto, cada bloco é então transformado com a ajuda da transformada wavelet bidimensional (84), reescrita novamente abaixo para inserir a notação adotada para os blocos

$$\tilde{A}_k = W_{n_k} A_k W_{n_k}^T \quad (100)$$

onde  $\tilde{A}_k$  representa o bloco transformado e também tornado esparsa pela técnica do *threshold*. A operação inversa, isto é, a obtenção do bloco original  $A_k$  a partir de sua representação aproximada, é obtida pré e pós-multiplicando o bloco transformado  $\tilde{A}_k$  pela matriz identidade, representada pela forma alternativa proporcionada pela propriedade de ortogonalidade da matriz wavelet (80), obtendo-se

$$\mathbf{A}_k = \mathbf{W}_{n_k}^T \tilde{\mathbf{A}}_k \mathbf{W}_{n_k} \quad (101)$$

onde  $\mathbf{W}_{n_k}$  é a matriz representando a transformada wavelet unidimensional de tamanho  $n_k$ . Como o requerimento de admissibilidade (98) deve ser satisfeito para *qualquer* matriz, os blocos podem ser reagrupados novamente em uma única matriz por

$$\tilde{\mathbf{A}} = \sum_{k=1}^M \mathbf{L}_{N,n_k}^{row_k} \tilde{\mathbf{A}}_k \left( \mathbf{L}_{N,n_k}^{col_k} \right)^T \quad (102)$$

Inserindo (96) em (100) e então inserindo o resultado em (102), a Transformada Wavelet em Blocos (Block Wavelet Transform ou BWT) da matriz original  $\mathbf{A}$  é explicitamente escrita como

$$\tilde{\mathbf{A}} = \sum_{k=1}^M \mathbf{L}_{N,n_k}^{row_k} \mathbf{W}_{n_k} \left( \mathbf{L}_{N,n_k}^{row_k} \right)^T \mathbf{A} \mathbf{L}_{N,n_k}^{col_k} \mathbf{W}_{n_k}^T \left( \mathbf{L}_{N,n_k}^{col_k} \right)^T \quad (103)$$

A Transformada Wavelet em Blocos Inversa (*Inverse Block Wavelet Transform* ou IBWT) é obtida usando a propriedade de extração (96) para recuperar os blocos, aplicando-se a transformada inversa (101) nos blocos obtidos e finalmente usando a condição de admissibilidade do esquema de partição (98), resultando na equação da IBWT dada por

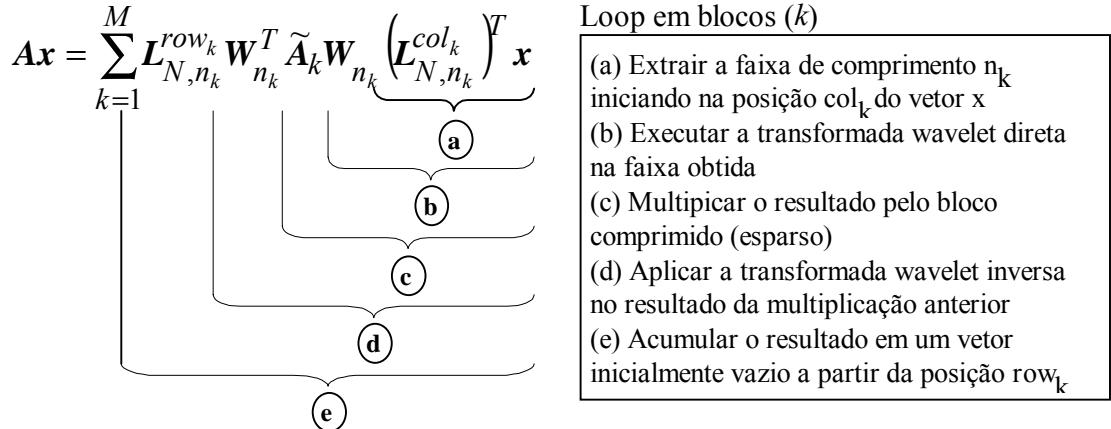
$$\mathbf{A} = \sum_{k=1}^M \mathbf{L}_{N,n_k}^{row_k} \mathbf{W}_{n_k}^T \left( \mathbf{L}_{N,n_k}^{row_k} \right)^T \tilde{\mathbf{A}} \mathbf{L}_{N,n_k}^{col_k} \mathbf{W}_{n_k} \left( \mathbf{L}_{N,n_k}^{col_k} \right)^T \quad (104)$$

Utilizando-se a propriedade de extração (96) a equação (104) é facilmente reescrita no formato que servirá finalmente ao cálculo numérico:

$$\mathbf{A} = \sum_{k=1}^M \mathbf{L}_{N,n_k}^{row_k} \mathbf{W}_{n_k}^T \tilde{\mathbf{A}}_k \mathbf{W}_{n_k} \left( \mathbf{L}_{N,n_k}^{col_k} \right)^T \quad (105)$$

Esta forma coloca a matriz original  $\mathbf{A}$  como uma expansão em série de blocos comprimidos  $\tilde{\mathbf{A}}_k$ . As multiplicações matriz-matriz acima nunca serão calculadas explicitamente mas serão utilizadas em (93) para calcular as multiplicações matriz-vetor requeridas pelo *solver* iterativo. A multiplicação da matriz  $\mathbf{A}$  - aqui representando as parcelas compressíveis  $\mathbf{H}_c^S$  e  $\mathbf{G}_c^S$  - por um vetor genérico  $\mathbf{x}$  pode

então ser efetuada de uma forma equivalente, embora mais complexa, à do algoritmo da Figura 84. O novo algoritmo é ilustrado na Figura 85.



**Figura 85 Algoritmo para o cálculo da multiplicação entre a matriz comprimida em blocos por um vetor como requerido pela equação (93) usando a transformada wavelet em blocos genérica BWT**

#### V.4.4 O grid regular

Quando o particionamento da matriz original é realizado utilizando-se blocos de tamanho constante, uma grande economia em tempo de processamento pode ser obtida, como será mostrado adiante.

Dado que todos os blocos em um particionamento regular possuem o mesmo tamanho  $n$ , a posição relativa de cada bloco dentro da matriz original é dada por

$$\begin{aligned} row_i &= (i - 1) \cdot n + 1 \\ col_j &= (j - 1) \cdot n + 1 \end{aligned} \quad (106)$$

com ambas as variáveis  $i$  e  $j$  variando de 1 a  $M$ . Utilizando esta regularidade, simplifica-se a transformada inversa BWT (103) para

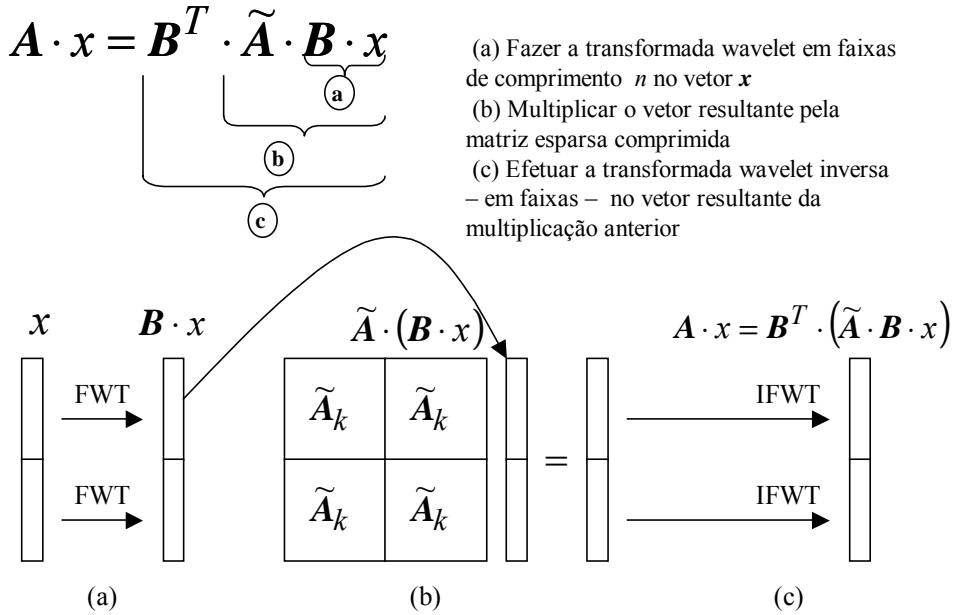
$$\tilde{\mathbf{A}} = \left( \sum_{i=1}^M [\mathbf{W}_n]_{row_i, row_i} \right) \mathbf{A} \left( \sum_{j=1}^M [\mathbf{W}_n^T]_{col_j, col_j} \right) \quad (107)$$

forma que pode ser resumidamente escrita como

$$\tilde{\mathbf{A}} = \mathbf{B} \mathbf{A} \mathbf{B}^T \quad (108)$$

denominada por ora em diante a *Transformada Wavelet Regular em Blocos (Regular Block Wavelet Transform ou RBWT)* onde a nova matriz  $\mathbf{B}$  é dada por

$$\mathbf{B} = \sum_{i=1}^M \left[ \mathbf{W}_n \right]_{row_i, row_i} \quad (109)$$



**Figura 86 Algoritmo para o cálculo da multiplicação matriz-vetor através de um esquema de particionamento regular em blocos de igual tamanho**

A transformada inversa (*Inverse Regular Block Wavelet Transform ou IRBWT*) é obtida facilmente a partir da simetria de (80) e (109) resultando em

$$\mathbf{A} = \mathbf{B}^T \tilde{\mathbf{A}} \mathbf{B} \quad (110)$$

O algoritmo para multiplicação matriz-vetor particular para este particionamento regular é então definido como ilustrado na Figura 86. Evidentemente a forma regular RBWT (108) é muito mais rápida que a forma genérica BWT (105) pois evita-se um somatório.

Como uma nota final, a transformada wavelet em blocos genérica BWT não se aplica somente à transformada wavelet tradicional. Todas as outras formas de transformadas wavelet como a forma não padrão ou o esquema de *lifting* podem ser utilizados. De fato, qualquer transformada pode ser utilizada desde que possa ser representada por uma matriz quadrada e desde que a propriedade (80) seja satisfeita. Desse ponto de vista, a BWT torna-se na realidade uma abordagem genérica para perfazer cálculos numéricos em matrizes particionadas em blocos.

## V.5 Condições de contorno não-lineares

A metodologia apresentada de montagem e compressão das matrizes de elementos de contorno em blocos foi bastante testada no presente trabalho em casos de condições de contorno de Dirichlet e Neumann e também de condições de contorno mistas lineares, como será visto adiante. Pela sua flexibilidade, torna-se relativamente fácil introduzir e testar condições de contorno genéricas não-lineares ao método de compressão por blocos. A introdução de tais condições de contorno exige a construção de algoritmos de solução não-lineares baseados no método de Newton-Raphson, para condições de contorno monótonas, e *simulated annealing* ou algoritmos genéticos para os casos genéricos. Somente o algoritmo de Newton Raphson foi implementado para este trabalho.

A implementação do algoritmo de Newton-Raphson para a solução de tais problemas não lineares é bastante conhecida, constando de apenas 18 linhas de código fonte C++, como será mostrado adiante. A idéia do solver Newton Raphson é expandir-se a solução em torno da solução atual em uma série de Taylor e manter-se somente os dois primeiros termos da série, o constante e o linear. Com isto consegue-se produzir uma equação linear que é usada num processo iterativo para achar por aproximações sucessivas a solução do problema não-linear.

Matematicamente a não-lineariedade das condições de contorno pode ser representada de tal forma que a solução do problema seja obter o vetor-solução  $\mathbf{x}$  para o operador não-linear  $A$  tal que a seguinte equação seja satisfeita:

$$A(\mathbf{x}) = \mathbf{b} \quad (111)$$

onde  $\mathbf{b}$  é o termo independente. Expandindo-se o operador  $A$  em torno da solução atual  $\mathbf{x}_0$  em uma série de Taylor, mantendo-se os dois primeiros termos desta série, desta forma ignorando a relevância dos termos de ordem superior e supondo-se que o vetor  $\mathbf{x}$  seja a solução exata, obtém-se

$$A(\mathbf{x}) = A(\mathbf{x}_0) + \frac{\partial A(\mathbf{x}_0)}{\partial \mathbf{x}} (\mathbf{x} - \mathbf{x}_0) = \mathbf{b} \quad (112)$$

Reordenando-se os termos desta última equação e fazendo  $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0$ , tem-se que

$$\frac{\partial A(\mathbf{x}_0)}{\partial \mathbf{x}} \Delta\mathbf{x} = \mathbf{r} = \mathbf{b} - A(\mathbf{x}_0) \quad (113)$$

onde  $\mathbf{r}$  é o resíduo da última iteração – portanto conhecido. O termo  $\partial A/\partial \mathbf{x}$  é um operador linear representando a tangente do operador  $A$  no ponto  $\mathbf{x}_0$ . Resolvendo-se esta equação obtém-se o vetor  $\Delta\mathbf{x}$  que é a correção necessária para minimizar a equação não-linear original. Este processo iterativo repete-se várias vezes até que uma determinada condição de parada seja satisfeita.

Dado que o método em si é trivial, falta somente calcular o termo  $\partial A/\partial \mathbf{x}$  sem tocar-se nas matrizes já comprimidas e também levando-se em conta as condições de contorno. Para tal basta aplicar o operador derivada  $\partial/\partial \mathbf{x}$  à equação (90) para obter-se:

$$\frac{\partial A}{\partial \mathbf{x}} = \mathbf{H} \frac{\partial \mathbf{X}_H}{\partial \mathbf{x}} - \mathbf{G} \frac{\partial \mathbf{X}_G}{\partial \mathbf{x}} \quad (114)$$

As matrizes  $\mathbf{H}$  e  $\mathbf{G}$  são constantes, portanto somente as derivadas das matrizes  $\mathbf{X}$  que representam as condições de contorno são necessárias. As matrizes comprimidas, portanto, permanecem intocadas. Para ser mais explícito, e introduzir também as matrizes residuais conforme a equação (94), o cálculo desta derivada é realizado por

$$\frac{\partial A}{\partial \mathbf{x}} = (\mathbf{W}^T \tilde{\mathbf{H}}_c^s \mathbf{W} + \mathbf{H}_r^s) \frac{\partial \mathbf{X}_H}{\partial \mathbf{x}} - (\mathbf{W}^T \tilde{\mathbf{G}}_c^s \mathbf{W} + \mathbf{G}_r^s) \frac{\partial \mathbf{X}_G}{\partial \mathbf{x}} \quad (115)$$

Vê-se que para obter a derivada tangente do operador não-linear  $A$  é preciso somente calcular a derivada das matrizes representando as condições de contorno, isto é, resta agora somente calcular-se os termos  $\partial \mathbf{X}_H/\partial \mathbf{x}$  e  $\partial \mathbf{X}_G/\partial \mathbf{x}$ , que são definidos exatamente como as matrizes de operadores  $\mathbf{X}_H$  e  $\mathbf{X}_G$ , da forma:

$$\left( \frac{\partial \mathbf{X}_H}{\partial \mathbf{x}} \right)_{i,j} = \begin{cases} \frac{\partial \lambda_i^H}{\partial \mathbf{x}} & \text{se } i = j \\ 0 & \text{caso contrário} \end{cases} \quad \left( \frac{\partial \mathbf{X}_G}{\partial \mathbf{x}} \right)_{i,j} = \begin{cases} \frac{\partial \lambda_i^G}{\partial \mathbf{x}} & \text{se } i = j \\ 0 & \text{caso contrário} \end{cases} \quad (116)$$

tarefa facilmente realizada com a ajuda da Tabela 5, reescrita na Tabela 6 para incluir a derivada das funções. Desta forma conclui-se que resolver a equação linear (113) para a correção de Newton Raphson é totalmente análogo a resolver um sistema linear equivalente utilizando-se o esquema de multiplicação matriz-vetor como ilustrado na Figura 85 exceto pelo fato de, ao invés de aplicar-se as condições de contorno na etapa (i), aplica-se a derivada das condições de contorno. Em outras palavras, ao invés de aplicar-se os operadores de contorno representados pelas matrizes  $\mathbf{X}_H$  e  $\mathbf{X}_G$  aplicam-se suas respectivas derivadas termos  $\partial \mathbf{X}_H/\partial \mathbf{x}$  e  $\partial \mathbf{X}_G/\partial \mathbf{x}$  como explicitado pela Tabela 6.

Note-se que nos casos lineares a atualização do Jacobiano é desnecessária, pois este é constante.

**Tabela 6 Derivadas das condições de contorno de acordo com a técnica da montagem virtual**

Tipo	Expressão usual	Implementação	Derivada
Potencial prescrito	$u = \bar{u}$	$\lambda^H x = 0, u_0 = \bar{u}$ $\lambda^G x = x, q_0 = 0$	$\frac{\partial \lambda^H}{\partial x} x = 0$ $\frac{\partial \lambda^G}{\partial x} x = 1$
Fluxo prescrito	$q = \bar{q}$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = 0, q_0 = \bar{q}$	$\frac{\partial \lambda^H}{\partial x} x = 1$ $\frac{\partial \lambda^G}{\partial x} x = 0$
Dependência linear	$q = a + bu$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = bx, q_0 = a$	$\frac{\partial \lambda^H}{\partial x} x = 1$ $\frac{\partial \lambda^G}{\partial x} x = b$
Dependência quadrática	$q = a + bu + cu$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = cx^2 + bx, q_0 = a$	$\frac{\partial \lambda^H}{\partial x} x = 1$ $\frac{\partial \lambda^G}{\partial x} x = 2cx + b$
Não-linear genérica	$q = f(u)$	$\lambda^H x = x, u_0 = 0$ $\lambda^G x = f(x), q_0 = 0$	$\frac{\partial \lambda^H}{\partial x} x = 1$ $\frac{\partial \lambda^G}{\partial x} x = \frac{\partial f(x)}{\partial x}$

## V.6 Estudo dos parâmetros para análise

A metodologia de compressão de matrizes e seu posterior uso em estado comprimido, conforme apresentado neste trabalho, permite uma grande flexibilidade no uso de diversas transformadas, não somente a wavelet, e de diversos parâmetros de análise. Esta flexibilidade, porém, acarreta um grande problema de definição dos melhores parâmetros a serem utilizados na análise, como a melhor família de wavelets a serem utilizadas, por exemplo.

### V.6.1 Escolha da família wavelet

As famílias de wavelet básicas são: Daubechies, Coiflet, Symlet e B-Splines, todas definidas em (DAUBECHIES, 92). Várias outras famílias de wavelets foram definidas

em diversos outros trabalhos. Somente um destes trabalhos constrói uma família de wavelets para o cálculo numérico, a wavelet de Ojanen (OJANEN, 1998).

Determinar a priori a qualidade e quantidade de compressão que possa resultar a escolha de determinado tipo de família de wavelet é uma tarefa que depende principalmente das matrizes a que a transformada será submetida, ou seja, da forma e propriedades dos dados sendo comprimidos.

Se, por exemplo, as matrizes forem compostas por planos constantes então uma wavelet Daubechies de ordem 1 (e filtro de comprimento 2) será suficiente. Se forem compostas por planos lineares então uma wavelet Daubechies de ordem 2 deverá ser utilizada sob pena de má compressão. A família de Daubechies, genericamente falando, comprime bem curvas formadas por polinômios: quanto maior a ordem do polinômio, maior deverá ser a ordem da wavelet.

A família das Coiflets é dirigida a dados com propriedades espectrais particulares. As Symlets são adaptadas a dados com planos de simetria definidos, como por exemplo um sinal composto por uma série de curvas gaussianas (ou sua segunda derivada, normalmente usada como fonte de impulso em análise numérica). As wavelets splines infelizmente não são ortogonais, mas sim biortogonais e, portanto, não podem ser utilizadas dentro do esquema de compressão apresentado.

A leitura dos parágrafos anteriores pode levar ao engano de que a seleção de uma wavelet de alta ordem possa ser a solução para todos os casos. Infelizmente a escolha de uma ordem alta, o que implica em uma máscara (o filtro FIR básico) de grande comprimento, afeta negativamente dois aspectos da análise: a qualidade da compressão e a velocidade de execução da transformada.

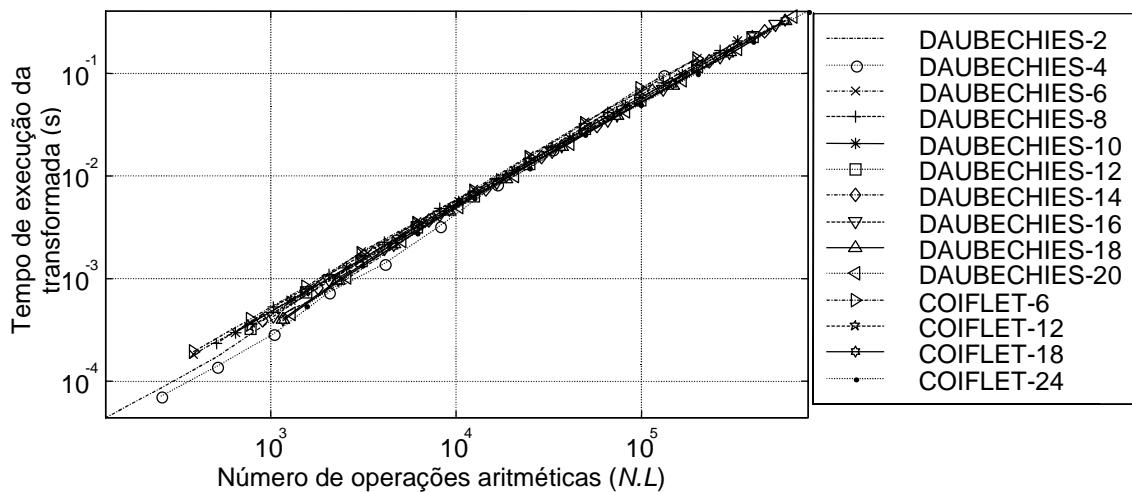
A qualidade da compressão é afetada porque filtros de grande comprimento tendem a “espalhar” a ocorrência de uma descontinuidade entre vários coeficientes. Isto é causado pelo fato de que o tamanho do filtro é diretamente proporcional ao comprimento da onda correspondente no espaço físico. Portanto, quanto maior o tamanho do filtro, maior quantidade de ondas serão atingidas pela descontinuidade, se existir.

Seja, por exemplo, um vetor de 1024 elementos composto por zeros em todos os pontos exceto no único ponto central cujo valor é “1”. Um filtro como o da wavelet de Daubechies de ordem 1, que possui 2 coeficientes, resulta em apenas 5 coeficientes

sendo afetados (os restantes são nulos) e o filtro da wavelet Daubechies de ordem 24, com 48 coeficientes, resulta em 192 coeficientes não nulos. Desta forma filtros de menor ordem comprimem melhor sinais com fortes descontinuidades.

O tamanho do filtro também afeta a velocidade de execução da transformada wavelet pois a transformada é uma sequência de convoluções entre o sinal original e o filtro wavelet. Defina-se, arbitrariamente, o número de operações aritméticas sendo realizadas, por simplicidade, como o tamanho do filtro wavelet ( $L$ ) multiplicado pelo tamanho do sinal ( $N$ ). Em verdade o número de operações aritméticas realizadas é proporcional a  $LN \log(N)$  mas o valor  $LN$  será adotado por simplicidade de cálculo para permitir uma noção intuitiva mais clara da velocidade do processo.

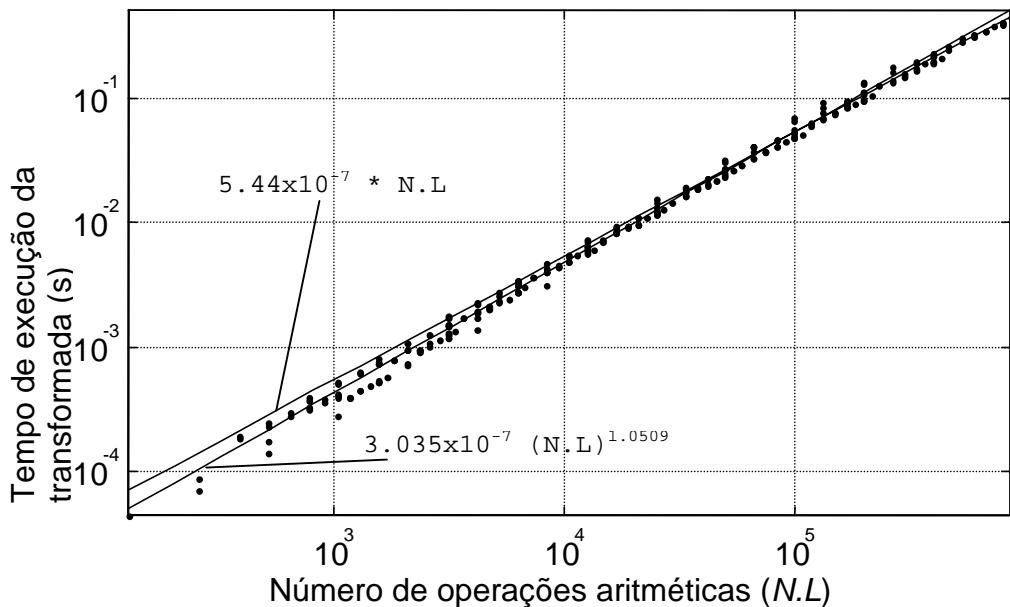
A Figura 87 mostra o tempo médio de processamento, em segundos, requerido por operação aritmética conforme definido acima. Os resultados foram obtidos utilizando-se um microcomputador AMD K6-2 300MHz e o sinal filtrado foi sintetizado por uma série randômica. Variou-se o tamanho dos vetores submetidos à transformada de 128 até 65536 elementos, o qual é uma faixa bastante larga, visto que o tamanho dos blocos a ser usado para compressão de matrizes dificilmente excederá 4096 elementos (que exige 256Mb de memória para armazenar as duas matrizes). O número de operações aritméticas que aparece nos gráficos é exatamente o tamanho do filtro ( $L$ ) multiplicado pelo tamanho do vetor ( $N$ ), uma aproximação grosseira.



**Figura 87 Tempos de execução da transformada wavelet para algumas famílias, variando-se o tamanho do vetor transformado de 64 a 32768 elementos.**

Percebe-se na Figura 87 que todos os gráficos se sobrepõem, mostrando que o tempo de execução depende principalmente do número de operações aritméticas realizadas. Esta conclusão sugere realizar-se uma regressão, linear ou logarítmica, a qual é mostrada na

Figura 88. Esta correlação foi realizada com 25 famílias de wavelets, dos grupos Daubechies, Coiflet e Ojanen. Cada ponto no gráfico representa uma combinação entre uma wavelet e um tamanho de vetor.



**Figura 88 Regressão linear e logarítmica realizadas sobre os testes de velocidade da transformada wavelet.**

As duas regressões, linear e logarítmica, apresentaram excelente correlação com os dados obtidos mas a primeira regressão, a linear, será adotada por simplicidade. Conclui-se que, para este computador em particular, o tempo de processamento para uma transformada pode ser previsto por  $5,44 \cdot 10^{-7} NL$  com o resultado obtido em segundos.

Para transformar-se um vetor de 4096 elementos utilizando-se uma wavelet cujo filtro FIR associado tenha 8 coeficientes será gasto, portanto, cerca de  $5,44 \cdot 10^{-7} * 4096 * 8 = 17,8$  milissegundos. A compressão de uma matriz ou bloco de N por N elementos necessita de  $2*N$  transformadas, N transformadas para cada direção – linha e coluna. Uma matriz de 4096 elementos necessitará, portanto, de  $2*4096*17,8$  ms = 146 segundos em transformadas para ser comprimida. Para um filtro com somente 2 coeficientes este tempo cai para apenas 36 segundos. Entretanto o processador no qual foram realizados estes testes, o AMD K6-2, é reconhecidamente ineficiente em operações em ponto-flutuante e tempos muito inferiores a estes são obtidos normalmente com os processadores K7 e Pentium-III/IV. Além disso, por efeitos de cache, estes tempos têm sido observados serem até três vezes inferiores aos previstos

como acima para a transformada bidimensional, dependendo da utilização prévia da matriz sendo transformada.

Esta matriz, somente para comparação, supondo ser resultado da integração de elementos de potencial constante 2D, exige cerca de 144 segundos para ser calculada por este mesmo processador. Sendo ambos os tempos, o gasto em transformadas wavelet para compressão e o tempo de montagem da matriz de integração, proporcionais a  $N^2$ , é possível concluir que o tempo gasto em transformadas wavelet está entre 25% e 200% do tempo de montagem da matriz, dependendo principalmente do comprimento do filtro FIR associado e da complexidade das rotinas de elemento (observa-se que esta rotina do elemento potencial 2D é uma das mais simples a implementar).

Para a etapa do solver, no entanto, não são necessárias transformadas bidimensionais mas simples transformadas unidimensionais direta e inversa nos vetores a serem multiplicados pelas matrizes comprimidas, duas diretas e duas inversas por iteração, para ser exato. Para a matriz de 4096 por 4096 elementos acima, por exemplo, deverão ser gastos 4 vezes o tempo de uma transformada unidimensional, ou  $4 \times 17,8\text{ms} = 71,2\text{ms}$  por iteração. Pode-se ver, portanto, que o esforço despendido com transformadas wavelet durante o solver é menor que o necessário durante a compressão.

Ao analisar-se tais tempos deve-se levar em conta que os algoritmos de convolução utilizados não estão otimizados, consistindo da forma acadêmica da convolução unidimensional. Existem rotinas em C especializadas em filtros de pequeno tamanho, como as do pacote freeware Seismic Unix da Colorado School of Mines, ou mesmo pacotes comerciais programados diretamente em linguagem Assembler, como o pacote de processamento de sinais para processadores Intel, da Intel, que utilizam operações especiais do processador em particular. A melhora da performance da transformada pode ser surpreendentemente melhorada – até 20 vezes em alguns casos – utilizando-se tais rotinas otimizadas.

O tamanho do filtro afeta, sem proporcionalidade, também a qualidade da transformada pelo número superior de operações aritméticas realizadas, o que por sua vez introduz erros de arredondamento. Exceto pela transformada *lifting* inteira todas as demais estão sujeitas a tais erros.

De uma forma geral pode-se dizer que o erro introduzido pela transformada é função de três fatores principais: o comprimento do filtro FIR associado, a qualidade dos coeficientes deste filtro e, finalmente, dos próprios dados sendo transformados. Evidentemente a arquitetura da máquina também influi mas, se for suposto que todas seguem a norma IEEE-754, espera-se que os resultados sejam iguais.

O comprimento do filtro influí no erro introduzido exatamente pelo número de operações aritméticas que adiciona à convolução. A qualidade dos coeficientes, no entanto, é importante devido ao fator de ortogonalidade, o qual define a inversibilidade da transformação. Sejam estes coeficientes mal calculados, ou calculados com um número de dígitos insuficientes, a transformação não será mais ortogonal. Aumentando-se o número de transformações para dez, como realizado nos testes da Tabela 7, aumentam-se também os erros introduzidos pela transformação pois estes são cumulativos.

Finalmente, os próprios dados influenciam no erro introduzido na transformada pelo fato de que eles irão definir o número e a quantidade de erros individuais introduzidos em arredondamento por operação aritmética. Um vetor composto por zeros não introduzirá erro algum à transformada, sejam quais forem os coeficientes do filtro.

Para avaliar a qualidade dos filtros utilizados para compressão neste trabalho, mostra-se na Tabela 7 os resultados de um teste realizado com vetores de 4096 elementos. Estes vetores foram submetidos à transformada direta e inversa uma dezena de vezes e o vetor resultante foi comparado com o vetor inicial utilizando-se duas normas de erro: a norma  $L^2$ , ou erro RMS, e a norma infinita, ou erro máximo.

Pela análise da Tabela 7 pode-se notar que não se pode relacionar o erro, RMS ou máximo, introduzido na transformada com o tamanho do filtro como pode-se fazer em relação à sua velocidade. Entretanto pode-se verificar os erros relativos a todos os filtros utilizados neste trabalho estão muito bem limitados e próximos à precisão da máquina, não influenciando, muito provavelmente, os resultados numéricos dos experimentos.

**Tabela 7 Avaliação de erro da transformada para várias famílias de wavelet.  
Vetores transformados randômicos com 4096 elementos.**

<i>Wavelet</i>	<i>Comprimento Filtro FIR</i>	<i>Erro RMS</i> <sup>*</sup>	<i>Erro máximo</i>
DAUBECHIES-2	2	1.130E-14	1.600E-14
DAUBECHIES-4	4	3.950E-15	8.100E-15
DAUBECHIES-6	6	1.310E-14	1.990E-14
DAUBECHIES-8	8	3.430E-14	4.550E-14
DAUBECHIES-10	10	1.500E-14	2.150E-14
DAUBECHIES-12	12	1.810E-14	2.800E-14
DAUBECHIES-14	14	2.920E-14	4.460E-14
DAUBECHIES-16	16	1.710E-14	2.850E-14
DAUBECHIES-18	18	4.320E-15	1.230E-14
DAUBECHIES-20	20	9.630E-15	1.930E-14
COIFLET-6	6	3.820E-15	7.770E-15
COIFLET-12	12	5.110E-15	1.230E-14
COIFLET-18	18	4.150E-15	1.340E-14
COIFLET-24	24	3.320E-15	1.440E-14
OJANEN-8	8	3.880E-15	8.440E-15
OJANEN-10	10	2.640E-15	8.990E-15
OJANEN-12	12	5.440E-15	1.170E-14
OJANEN-14	14	2.640E-15	9.880E-15
OJANEN-16	16	7.760E-15	1.490E-14
OJANEN-18	18	5.340E-15	1.330E-14
OJANEN-20	20	2.580E-15	9.880E-15
OJANEN-22	22	2.840E-15	1.020E-14
OJANEN-24	24	3.240E-15	1.220E-14
OJANEN-26	26	3.140E-15	1.040E-14

### V.6.2 Escolha dos limiares de compressão (thresholds)

Movendo-se da visão micro-escala para uma perspectiva mais ampla, o *threshold*, ou limiar de erro, o principal parâmetro de compressão, é o fator preponderante na determinação da acurácia e da taxa de compressão dos resultados.

Seja uma matriz quadrada  $\mathbf{A}$  com  $N$  por  $N$  elementos. Seja uma aproximação da matriz  $\mathbf{A}$  original, obtida por compressão, da qual  $\Delta\mathbf{A}$  é a diferença entre as matrizes, ou melhor, o erro da compressão elemento-por-elemento. O erro introduzido em cada elemento  $\mathbf{b}_i$  de um vetor  $\mathbf{b}$  com  $N$  elementos, resultante da multiplicação da matriz aproximada por um vetor arbitrário  $\mathbf{x}$  é limitado por

$$|\Delta\mathbf{b}_i| \leq \sum_m |\Delta\mathbf{A}_{im}| \|\mathbf{x}_m\| \leq N \Delta A_{\max} x_{\max} \quad (117)$$

---

\* Erros são residuais após 10 transformações sequenciais, direta e inversa.

sendo  $\Delta A_{\max}$  o maior erro de todos os elementos da matriz  $\mathbf{A}$  e  $x_{\max}$  o maior elemento, em módulo, do vetor  $\mathbf{x}$ . O erro  $\Delta A_{\max}$  por motivos melhor entendidos em (DONOHO, 93) pode ser tomado como exatamente igual ao threshold  $T_r$  adotado para compressão da matriz  $\mathbf{A}$ . Isto leva à conclusão que

$$|\Delta \mathbf{b}|_{\max} \leq N \cdot T_r \cdot x_{\max} \quad (118)$$

Ou seja, o erro máximo introduzido pela multiplicação de uma matriz comprimida por um vetor arbitrário é, no máximo, igual à dimensão da matriz multiplicada pelo threshold adotado na compressão, multiplicados pelo valor máximo do vetor sendo multiplicado. Para uma matriz de 1024 elementos comprimida por um threshold de  $10^{-5}$ , o erro a ser esperado no vetor resultante da multiplicação será sempre menor que  $0,01 \cdot x_{\max}$ . Este *upper-bound*, embora correto, é muito restrito no sentido probabilístico, devendo-se substituir  $x_{\max}$  por um valor menor baseado em alguma distribuição probabilística adequada.

O limite acima apresentado foi obtido assumindo-se as seguintes premissas:

1. todos os valores do somatório são positivos e de igual valor ao valor máximo do vetor; e
2. todos os fatores multiplicativos correspondentes na matriz  $\mathbf{A}$  possuam erro igual ao erro máximo por compressão na matriz.

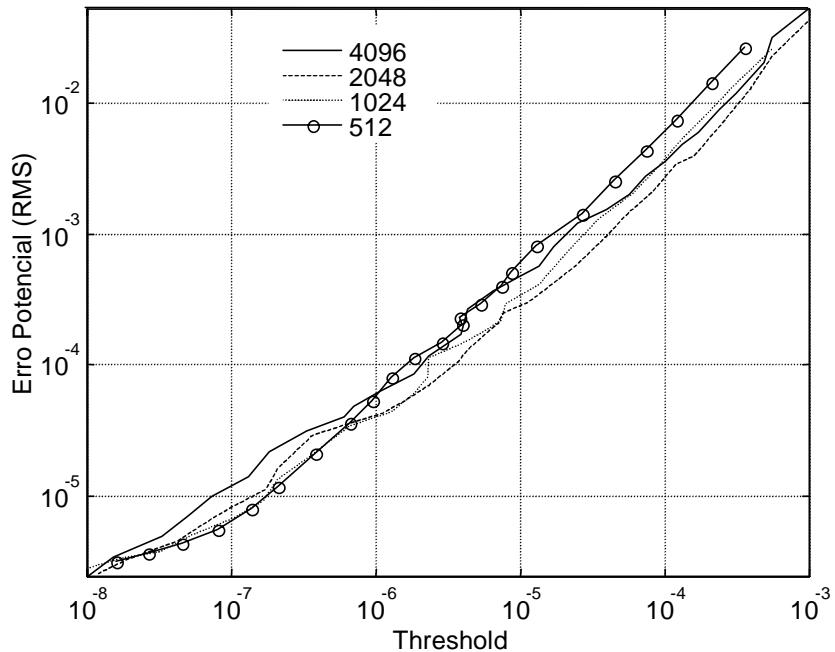
Evidentemente as duas premissas têm pouca ou nenhuma probabilidade de acontecer em uma aplicação real. Como as wavelets são compostas por ondas com média zero, é praticamente impossível que os erros resultantes de compressão nas matrizes tenham o mesmo sinal. Desta forma, os valores positivos e negativos do somatório tendem a anular-se conforme  $N$  tenda ao infinito, mantendo o erro final limitado a valores muito menores que os propostos acima pelo limite máximo, sendo este excessivamente elevado para ser tomado como provável. A discussão anterior serve, no entanto, para adicionar significado ao valor do threshold, ou limiar de erro,  $T_r$ .

### V.6.3 Influência do tamanho dos blocos

Para analisar a influência do tamanho dos blocos sobre a taxa de compressão final foi usado o modelo do problema externo de potencial ilustrado na Figura 99. Este modelo foi discretizado com 1024 elementos constantes por lado, totalizando 4096 elementos. Esta discretização foi escolhida de tal forma que a divisão por blocos fosse perfeita,

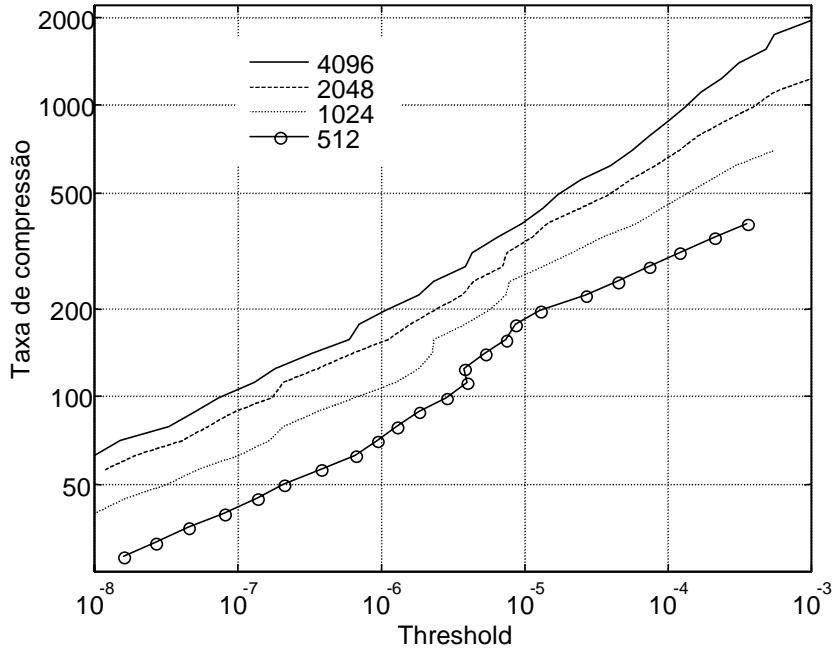
evitando assim a introdução de fatores indesejáveis. Um modelo padrão foi então resolvido utilizando-se o *solver* direto (Gauss) como referência para o cálculo do erro.

Este modelo foi particionado com blocos de 4 diferentes tamanhos: 512, 1024, 2048 e 4096 elementos. Cada particionamento foi submetido a uma bateria de 400 testes cada variando-se os thresholds na faixa de  $10^{-8}$  a  $10^{-3}$ . Os resultados foram então compilados e os erros RMS obtidos comparando-se com a solução de referência obtida usando o *solver* direto. Os resultados são apresentados na Figura 89, Figura 90 e Figura 91.



**Figura 89 Evolução do erro na solução do potencial com o valor do threshold para diferentes particionamentos (tamanhos de bloco)**

Os resultados mostrados na Figura 89 sugerem que o tamanho do bloco não influencia o erro pois para determinado threshold o erro permanece independente do tamanho do bloco. Este comportamento tem se repetido em todas as análises feitas durante o curso deste trabalho. Este resultado simplifica bastante o trabalho de determinação do threshold a ser utilizado para determinada simulação visto que este valor deve ser escolhido antes da compressão e não pode ser modificado após.



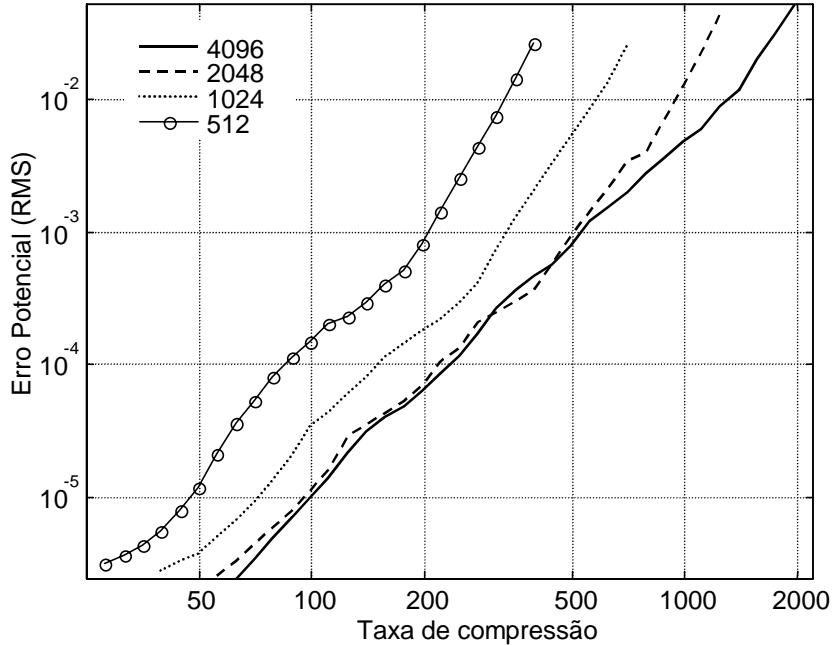
**Figura 90 Taxa de compressão resultante do threshold escolhido. Resultados para diferentes tamanhos de blocos.**

Verificado que o erro não varia com o tamanho do bloco, deve-se estudar se a taxa de compressão é dependente ou não do grid de particionamento. A Figura 90 mostra claramente que existe uma grande dependência entre o refinamento do grid e a taxa de compressão. Este resultado justifica-se por dois motivos principais: a estrutura necessária para armazenar um bloco é composta por um cabeçalho e os elementos em si. O tamanho do cabeçalho é fixo para cada bloco, mesmo que não exista nenhum elemento armazenado. Desta forma a memória exigida para guardar o mesmo número de elementos num grid com blocos pequenos será definitivamente maior que a necessária quando usado um grid com blocos de tamanho maior.

O segundo motivo deve-se à transformada wavelet em si que tem a facilidade de escolher os coeficientes que melhor representem o vetor a ser comprimido. Se o conteúdo espectral deste vetor for mantido seja qual for seu tamanho, a tendência é que o número de coeficientes wavelet não cresça proporcionalmente ao número de elementos do vetor, mas sim em uma taxa mais lenta. Desta forma, ao aumentar-se o tamanho do vetor ou matriz, aumenta-se também a relação entre o número total de elementos neste vetor ou matriz e o número de coeficientes wavelet necessários para representá-lo dentro de determinado limite de erro.

Esta relação é diretamente proporcional à taxa de compressão – exceto pelo *overhead* do cabeçalho da estrutura esparsa. Se este vetor tem um grande suavidade a

transformada wavelet irá escolher alguns poucos coeficientes de baixa frequência para representá-lo. Quanto maior o número de pontos, portanto, maior a probabilidade da transformada encontrar um coeficiente bem centrado na característica marcante do vetor ou função.



**Figura 91 Evolução do erro na solução do potencial com o aumento da taxa de compressão para diferentes particionamentos (tamanhos de bloco)**

Reapresentando os últimos resultados em uma forma alternativa, são mostrados os valores da evolução do erro com a taxa de compressão na Figura 91, na qual comprova-se que blocos maiores fornecem maiores taxas de compressão para o mesmo nível de erro.

Entretanto verifica-se que existe pouca diferença entre o grid particionado com blocos de tamanho 2048 e 4096. Este comportamento é também repetido em outros problemas e sugere um nível de saturação, isto é, um ponto a partir do qual não adianta aumentar-se o tamanho dos blocos visando o aumento da taxa de compressão. No entanto, observa-se que, a partir de uma determinada taxa de compressão existe uma tendência a um melhor comportamento dos blocos de maior tamanho, comportamento este que também se repete em outros casos.

Como uma referência quantitativa pode-se dizer que o tamanho de 2048 elementos é o tamanho mínimo do bloco para obter-se qualquer resultado útil e 4096 é um valor próximo do ótimo. Com este tamanho de bloco já foi possível resolver o sistema de equações de dois problemas – par galvânico e a placa infinita – com 250.000 graus de

liberdade cada em um microcomputador PC-600 MHz com 512 Mb de memória em menos de 3 minutos com um erro desprezível introduzido na solução.

## V.7 Paralelização e particionamento

A maior vantagem que a presente formulação para compressão de matrizes em elementos de contorno trouxe talvez seja a reutilização da estrutura comprimida para o cálculo de outros problemas com a mesma geometria mas condições de contorno diversos, o que faz deste método extremamente rápido quando utilizado por um esquema de otimização – algoritmos genéticos ou *simulated annealing*, por exemplo.

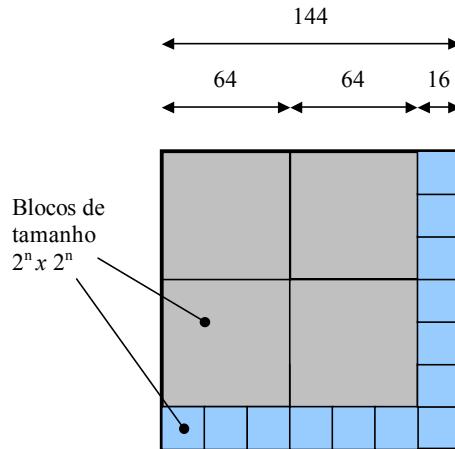
O calcanhar de Aquiles, entretanto, ainda é o tempo de montagem e compressão das matrizes, que continua de ordem  $O(N^2 \log N^2)$ . Apesar de lento, este processo é facilmente paralelizável se for utilizada a compressão por blocos como proposto.

Se a transformada wavelet utilizada estiver na forma do esquema de *lifting* (GONZALEZ, 2000) não importa qual o tamanho da matriz, somente a disponibilidade de memória é importante. Mas, se por alguma razão, como a pouca disponibilidade de famílias de wavelets (as famílias para serem usadas com o *lifting* precisam ser modificadas) ou mesmo facilidade de implementação, não for possível usar o *lifting* mas sim a forma padrão, deve-se lidar com o problema da exigência desta forma em realizar a transformada em blocos de potência de dois.

Esta restrição não é realmente uma desvantagem, apenas exige um cuidado adicional na implementação dos algoritmos. De fato, a BWT (103) permite a construção e o cálculo de problemas em grids de blocos de tamanho arbitrário através da equação (105). A Figura 92 mostra a solução para o problema de uma matriz de tamanho qualquer usando a transformada em blocos genérica BWT. A solução neste caso é dividir a matriz em blocos de tamanho máximo no lado esquerdo superior (menores índices) e na área restante preencher com blocos de tamanho menor.

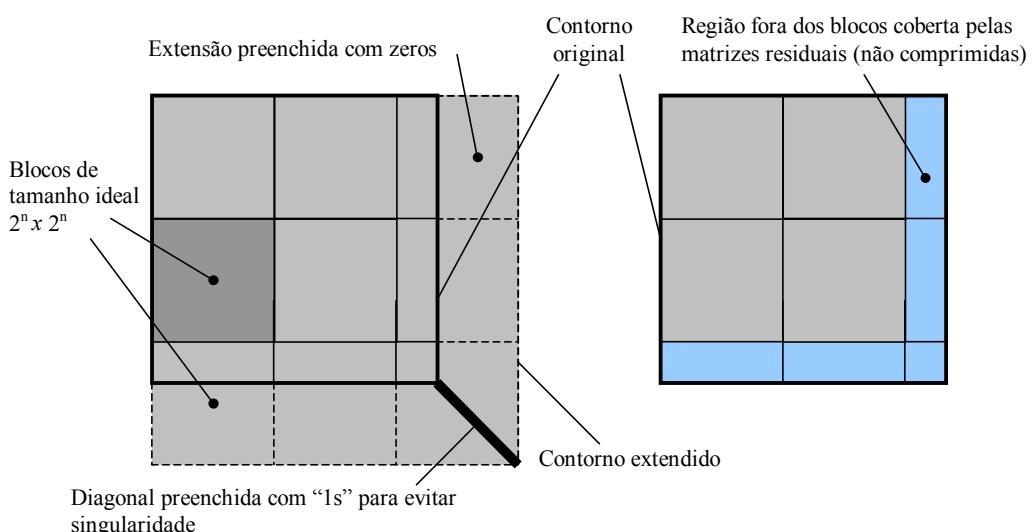
Se, por exemplo, a matriz tiver um tamanho de  $144 \times 144$ , uma divisão simples é usar a representação binária para este número, que é “1001 0000” ou  $128+16$ . Supondo que o bloco máximo possível seja 64, então usa-se blocos de tamanho  $2 \times 64 + 16$ , como visto na Figura 92. Como qualquer número inteiro pode ser expresso em sua representação binária, resolve-se o problema sem muito esforço.

No entanto, o tamanho do bloco – como visto anteriormente – é um fator que influencia decisivamente a taxa de compressão. Via de regra, quanto maior o bloco, maior a taxa de compressão. Devido a este fato cobrir uma grande área da matriz com vários blocos de pequeno tamanho é extremamente contraproducente.



**Figura 92 Solução para partição de matriz de tamanho qualquer usando a transformada em blocos genérica BWT**

A saída básica para este impasse é estender a matriz original além de seu tamanho original tal que o novo tamanho “virtual” seja composto por blocos de tamanho maior que o mínimo aceitável, idealmente o tamanho máximo permitido pela memória disponível. Se, no entanto, o excesso for pequeno não compensa alocar toda uma estrutura de dados apenas para guardar umas poucas linhas e colunas. Neste caso a solução é abrigar este pequeno excesso nas matrizes residuais (92) que são naturalmente esparsas. Estas duas abordagens são melhor entendidas com a ajuda da Figura 93.



**Figura 93 Soluções para a partição de matriz de tamanho qualquer para uso com a RBWT**

Na primeira abordagem, o contorno da matriz é efetivamente estendido para um valor múltiplo do tamanho mínimo adotado. As áreas externas à matriz original são então preenchidas com zeros e, para evitar singularidade no sistema global, a diagonal nesta área externa é preenchida com quaisquer valores não nulos. Tendo em vista a possível sensibilidade numérica do *solver*, estes valores devem ser escolhidos de forma que não sejam excessivamente grandes ou pequenos, geralmente escolhendo-se o valor 1.0. Se os valores das matrizes forem reconhecidamente grandes ou pequenos, deve-se escolher um valor na mesma ordem de grandeza destes. Uma boa medida é a média – em valor absoluto – dos valores da diagonal interna.

Esta é a solução mais fácil mas a mais perigosa pelo fato de que a extensão da matriz altera suas propriedades, entre as quais o próprio condicionamento. Entre todos os casos analisados não foi observado qualquer alteração negativa na convergência mas existe a possibilidade de algum efeito adverso ser introduzido, principalmente quando altas taxas de compressão forem utilizadas. A segunda abordagem – usando a matriz residual – é mais elegante e segura pois a matriz não é extendida mantendo assim suas propriedades originais. No entanto o *overhead* necessário para guardar cada elemento em forma esparsa (tipicamente o dobro do armazenamento em uma matriz densa) faz com que esta solução seja efetiva somente nos casos citados onde o excesso compõe-se de apenas algumas linhas e colunas.

Como pode-se notar, o algoritmo proposto para compressão em blocos é naturalmente paralelo. Uma abordagem tradicional “um mestre – vários escravos” é a escolha imediata para arquitetura do processamento paralelo pois neste caso a intercomunicação entre os escravos é inexistente, mantendo a eficiência da paralelização bem próxima de 100% e os *speedups* proporcionais ao número de máquinas – numa rede homogênea.

A forma mais fácil de iniciar o processamento em paralelo seria enviar, num primeiro estágio, a definição do problema por completo para todos os escravos e então sequencialmente outorgar blocos a serem calculados por cada um. A possibilidade de ter-se blocos com diferentes tamanhos (usando a forma genérica BWT) permite ótimas propriedades de balanceamento de carga porque desta forma máquinas com pouca memória e/ou lentas podem ser alocadas com blocos de menor tamanho enquanto máquinas mais rápidas e com mais memória podem lidar com grandes blocos.

Qualquer algoritmo, seja serial ou paralelo, poderia fazer uso de métodos como *panel clustering* ou ainda *multipole* para aumentar a velocidade do processamento. De fato, o

tempo de montagem é o gargalo deste processo, o qual não é devido ao uso da transformada wavelet mas sim ao crescimento em ordem quadrática do tamanho da matriz – que ainda precisa ser montada para ser comprimida.

## V.8 *Implementação computacional*

A implementação computacional das metodologias propostas neste capítulo incorre em um grande número de decisões sendo tomadas no sentido de aumentar a performance dos algoritmos propostos. Este subitem deste capítulo se dedicará a mostrar alguns tópicos relativos à programação em si das rotinas de forma aumentar esta performance. Os apêndices A e B discutem longamente alguns aspectos computacionais relevantes que precisaram ser estudados em maiores detalhes.

### V.8.1 Formato das matrizes esparsas

Existem basicamente dois formatos de armazenamento de matrizes esparsas populares em vários trabalhos (GOLUB, VAN LOAN, 1989, SAAD, 1996): o formato comprimido por colunas ou linhas (FCL) e o formato em coordenadas (FCCRD). A utilização de um ou outro formato de armazenamento depende basicamente da taxa de compressão esperada do modelo.

O formato de armazenamento por coordenadas (FCCRD) é bem simples, como pode-se ver pela Figura 94. A estrutura esparsa é formada por três vetores: um ponto flutuante VAL, que armazena os valores em si e os vetores COL e LIN que armazenam a posição na matriz original destes elementos, daí o seu nome *por coordenadas*. Pode-se ver que este formato tende a uma certa redundância no vetor LIN quando existem dois ou mais elementos numa mesma linha. Os dois elementos, 6.0 e 7.0, na primeira linha, por exemplo, inserem duas entradas de igual valor no vetor LIN – dois valores 1. O mesmo acontece com a linha 4. Por este motivo criou-se o formato comprimido por linhas, ou FCL, que adiciona um grau a mais de compressão à estrutura esparsa.

$N = 5$	$ne = 7 \text{ coeficientes}$																																
<table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>6.0</td><td></td><td>7.0</td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td>8.0</td><td></td></tr> <tr><td>9.0</td><td></td><td></td><td></td><td></td></tr> <tr><td></td><td>10.0</td><td></td><td></td><td>11.0</td></tr> <tr><td></td><td>12.0</td><td></td><td></td><td></td></tr> </table>	6.0		7.0						8.0		9.0						10.0			11.0		12.0				<b>VAL</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>6.0</td><td>7.0</td><td>8.0</td><td>9.0</td><td>10.0</td><td>11.0</td><td>12.0</td></tr> </table> tam = $8ne$	6.0	7.0	8.0	9.0	10.0	11.0	12.0
6.0		7.0																															
			8.0																														
9.0																																	
	10.0			11.0																													
	12.0																																
6.0	7.0	8.0	9.0	10.0	11.0	12.0																											
	<b>COL</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>1</td><td>3</td><td>4</td><td>1</td><td>2</td><td>5</td><td>2</td></tr> </table> tam = $4ne$	1	3	4	1	2	5	2																									
1	3	4	1	2	5	2																											
	<b>LIN</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>1</td><td>1</td><td>2</td><td>3</td><td>4</td><td>4</td><td>5</td></tr> </table> tam = $4ne$	1	1	2	3	4	4	5																									
1	1	2	3	4	4	5																											
	$Mem = 16ne \text{ bytes}$																																

**Figura 94 Exemplo de armazenamento utilizando o formato comprimido por coordenada (FCCRD)**

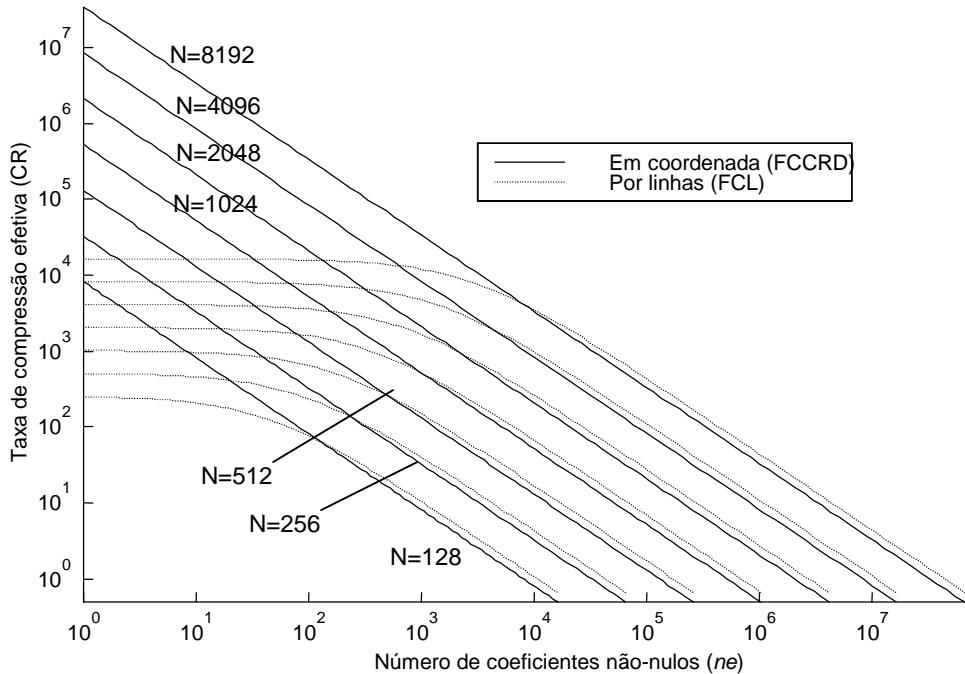
O formato comprimido por linhas, FCL, é melhor explicado por um exemplo, ilustrado na Figura 95. A estrutura esparsa é formada por três vetores: um vetor ponto-flutuante VAL e dois vetores inteiros COL e LINP. A matriz esparsa a ser armazenada é mostrada na esquerda, onde apenas os elementos significativos estão mostrados. O vetor VAL armazena os valores ponto-flutuante significativos e o vetor COL armazena a posição horizontal, a coluna onde este valor está armazenado. O vetor LINP armazena a posição, nos vetores VAL e COL, do primeiro elemento da linha especificada. Por exemplo, vê-se que o primeiro elemento da linha 2, o elemento de valor 8.0, inicia-se na posição 3 dos vetores COL e VAL. Da mesma forma, o primeiro elemento da linha 5, de valor 12.0, inicia-se na posição 7 destes vetores.

$N = 5$	$ne = 7 \text{ coeficientes}$																																
<table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>6.0</td><td></td><td>7.0</td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td>8.0</td><td></td></tr> <tr><td>9.0</td><td></td><td></td><td></td><td></td></tr> <tr><td></td><td>10.0</td><td></td><td></td><td>11.0</td></tr> <tr><td></td><td>12.0</td><td></td><td></td><td></td></tr> </table>	6.0		7.0						8.0		9.0						10.0			11.0		12.0				<b>VAL</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>6.0</td><td>7.0</td><td>8.0</td><td>9.0</td><td>10.0</td><td>11.0</td><td>12.0</td></tr> </table> tam = $8ne$	6.0	7.0	8.0	9.0	10.0	11.0	12.0
6.0		7.0																															
			8.0																														
9.0																																	
	10.0			11.0																													
	12.0																																
6.0	7.0	8.0	9.0	10.0	11.0	12.0																											
	<b>COL</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>1</td><td>3</td><td>4</td><td>1</td><td>2</td><td>5</td><td>2</td></tr> </table> tam = $4ne$	1	3	4	1	2	5	2																									
1	3	4	1	2	5	2																											
	<b>LINP</b> <table border="1" style="border-collapse: collapse; width: 100px;"> <tr><td>1</td><td>3</td><td>4</td><td>5</td><td>7</td></tr> </table> tam = $4N$	1	3	4	5	7																											
1	3	4	5	7																													
	$Mem = 4N + 12ne \text{ bytes}$																																

**Figura 95 Exemplo de armazenamento utilizando o formato comprimido por linhas (FCL)**

Este formato de armazenamento é útil quando existe pelo menos um coeficiente por linha, o que é o caso na maioria das aplicações de elementos finitos, por exemplo. Entretanto, no caso da compressão wavelet, as matrizes são realmente muito esparsas, algumas linhas não possuindo sequer um elemento. Inicialmente no desenvolvimento deste trabalho, foi adotado o formato FCL mas, com a evolução dos trabalhos viu-se que este formato não suporta as altas taxas de compressão obtidas.

Supondo-se os números ponto flutuantes como *doubles* de 64 bits e os inteiros de 32 bits, o formato FCCRD, como visto na Figura 94, exige  $16ne$  bytes de memória para armazenar a estrutura esparsa enquanto o formato FCL, visto na Figura 95, exige  $4N+12ne$  bytes, onde  $N$  é a dimensão da matriz quadrada e  $ne$  é o número de coeficientes não-nulos da matriz. Mostra-se na Figura 96 o gráfico da taxa de compressão efetiva, isto é, o valor  $8N^2/Mem$  onde  $Mem$  é a quantidade de memória exigida para armazenamento da estrutura esparsa, *versus* o número de coeficientes não-nulos da matriz ( $ne$ ). Neste gráfico foram estudadas matrizes de diversos tamanhos, variando de 128 a 8192 elementos, em potências de dois.

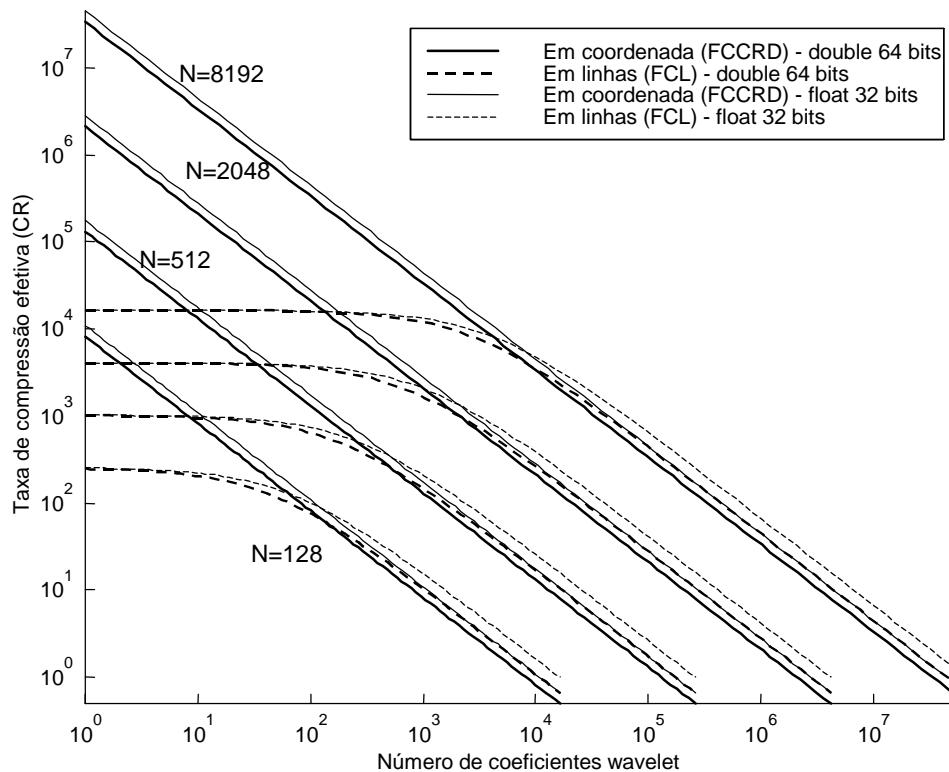


**Figura 96 Comparação da taxa de compressão efetiva utilizando-se o formato comprimido por linhas (FCL) e o formato comprimido por coordenadas (FCCRD)**

Percebe-se da Figura 96 que o formato FCL (comprimido por linhas) é superior ao formato FCCRD (por coordenadas) quando existem muitos elementos não-nulos na matriz, isto é, quando a taxa de compressão é baixa. Quando a taxa de compressão é alta, no entanto, o formato FCL não consegue taxas sucessivamente e proporcionalmente mais altas, tendendo a um limite finito. Este limite de compressão é dado exatamente pelo espaço exigido pelo vetor LNP que contém  $N$  elementos. Uma matriz esparsa no formato FCL irá exigir, no mínimo,  $8N$  bytes para armazenar qualquer matriz esparsa. O formato FCCRD, no entanto, não possui esta limitação pois a memória exigida para armazenamento ( $Mem$ ) é diretamente proporcional ao número de elementos não-nulos na matriz ( $ne$ ).

O limite em que ambos os formatos fornecem resultados iguais em termos de taxa de compressão efetiva é exatamente quando o número de valores significativos na matriz é igual à dimensão da matriz, isto é, quando  $N=ne$ . Como a intenção do presente desenvolvimento é alcançar as mais altas taxas de compressão, adotou-se o formato FCCRD, mesmo porque sua eficiência quando o número de elementos significativos na matriz é alto (e a compressão é baixa) não é muito menor que a eficiência do formato FCL.

Como dito anteriormente, os números em ponto flutuante adotados para armazenar as matrizes esparsas de coeficientes wavelet foram *floats* de 32 bits e não *doubles* de 64 bits. A discussão anterior, no entanto, permanece válida. Inclusive o limite de performance entre os dois formatos, quando o número de elementos não nulos é igual à dimensão da matriz, permanece exatamente o mesmo. Na Figura 97 mostra-se o efeito, sobre a taxa de compressão, da substituição dos elementos da matriz de *doubles* de 64 bits para *floats* de 32 bits.

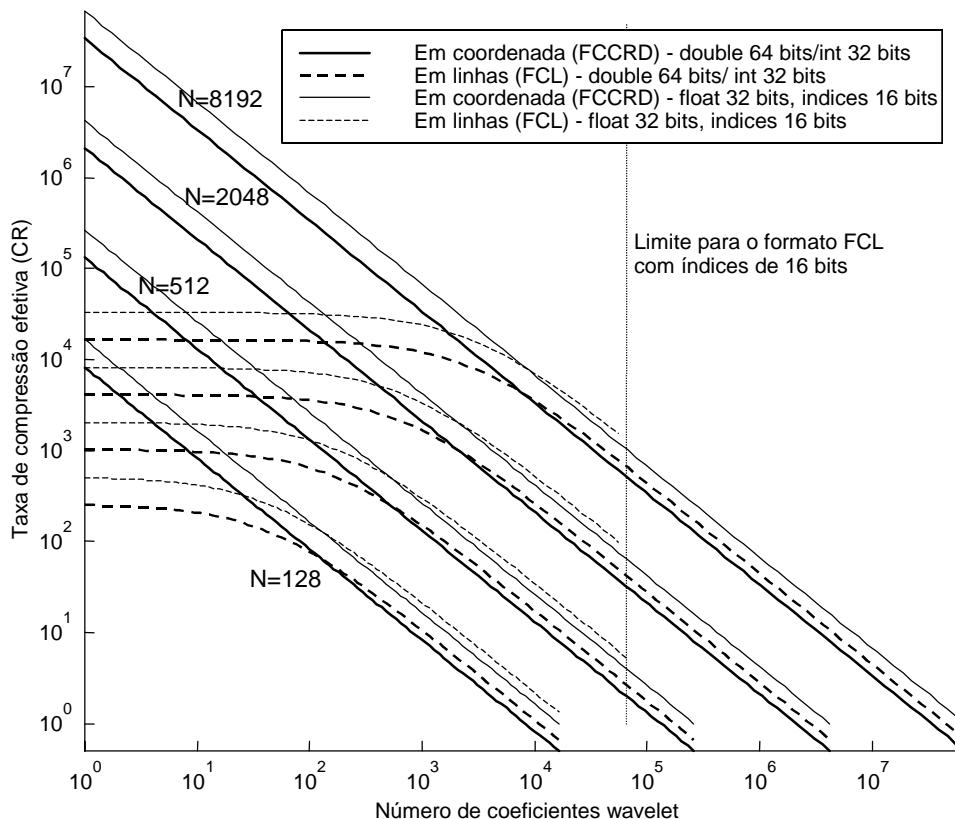


**Figura 97 Avaliação do efeito sobre a taxa de compressão efetiva da substituição dos elementos da matriz esparsa de *doubles* de 64 bits para *floats* de 32 bits**

Verifica-se na Figura 97 que o efeito sobre o formato em coordenadas (FCCRD) é quantitativamente o mesmo para toda faixa de coeficientes enquanto o efeito sobre o formato comprimido em linhas é significativo somente quando o número de coeficientes

é grande (e a compressão, baixa). Isto significa que, se utilizado o formato FCL, é inútil trocar-se o tipo dos números ponto-flutuante quando utilizados com as wavelets.

Um efeito de compressão adicional pode ser obtido se os números inteiros forem trocados de 32 para 16 bits. Esta é uma troca bastante limitante pois a dimensão das matrizes fica restrita a 65536. Esta não é realmente uma restrição para o formato FCCRD visto que um bloco de dimensão 8192 implica em um consumo de memória de 512 megabytes. Um bloco de dimensão 65536, 32 gigabytes! Se as matrizes esparsas forem armazenadas separadamente umas das outras, isto é, existindo uma estrutura esparsa única para cada bloco, pode-se tranquilamente trocar os índices de 32 para 16 bits. No formato FCL, entretanto, esta troca acarreta em uma restrição mais grave pois o número de elementos total ( $ne$ ) da matriz é limitado pelo vetor LIMP, que armazena a posição, nos vetores VAL e COL, do primeiro elemento de cada linha. A Figura 98 ilustra o efeito, sobre a taxa de compressão efetiva, da substituição dos inteiros de 32 bits por inteiros de 16 bits juntamente com a substituição dos números ponto flutuantes de *doubles* de 64 bits por *floats* de 32 bits.



**Figura 98 Avaliação do efeito sobre a taxa de compressão efetiva da substituição dos elementos da matriz esparsa de *doubles* de 64 bits para *floats* de 32 bits e da substituição dos índices de inteiros de 32 bits para inteiros de 16 bits**

Percebe-se da Figura 98 que a substituição dos índices de 32 bits por inteiros de 16 bits tem um efeito muito negativo sobre o formato FCL pois este é afetado na faixa onde seus efeitos são mais benéficos, quando o número de elementos na matriz é alto (e a compressão, baixa). O formato FCCRD é beneficiado uniformemente.

Desta forma considera-se que o formato esparsos por coordenadas com blocos esparsos utilizando índices de 16 bits e números ponto flutuante de 32 bits é o mais indicado para ser utilizado juntamente com a presente formulação de compressão de matrizes pelo método da wavelet.

### **V.8.2      Formato de armazenamento binário independente**

Revelando-se a compressão das matrizes bem sucedida, resta ainda a determinação do formato de transmissão e armazenamento das matrizes, que deve ser bastante estudado pois está-se lidando com matrizes cujo tamanho pode chegar facilmente a 500 Gigabytes.

Um formato binário, por exemplo, possibilita leituras/escritas extremamente rápidas pois não é necessária conversão de nenhum tipo mas é único para cada plataforma. Desta forma, uma matriz salva em uma Sun workstation não poderia ser lida por um micro PC. O formato texto é universal mas exige uma quantidade muito maior de espaço em disco – tipicamente 5 a 10 vezes o ocupado pelo formato binário – e leituras também muito mais lentas – tipicamente 100 vezes mais lento.

Uma alternativa a ambos é um formato binário universal denominado popularmente por XDR (SRINIVASAN, 1995). Este formato é utilizado amplamente em bibliotecas de trocas de mensagens como PVM ou MPI e já aceito também como parte da recomendação da linguagem XML (BRAY et al, 1998), com APIs para várias linguagens incluindo Java (MORDANI et al, 2001). Esta discussão, apesar da pouca relação aparente com o método de elementos de contorno em si, é decisiva para a performance dos algoritmos propostos.

Para ilustrar a presente discussão, foi realizado um teste para verificar as velocidades de escrita e leitura para números inteiros e ponto flutuante. Neste teste dois vetores de comprimento variável foram salvos e lidos de um arquivo temporário. Efeitos de cache foram minimizados pelo desligamento da máquina entre os testes de leitura e escrita. No entanto outros mecanismos de cache, intrínsecos ao sistema operacional e ao hardware

em si não foram levados em consideração por ocorrerem efetivamente em todas as situações.

Os resultados obtidos na Tabela 8 são particulares a um sistema AMD K6-2 333MHz com 64Mb RAM, HD Quantum Fireball ELX 5.1Gb e sistema operacional Windows 95.

**Tabela 8 Comparação de tempos de leitura e escrita de números inteiros e ponto flutuante em formato texto (ASCII) e binário (ponto flutuantes são *doubles* 64 bits e inteiros de 32 bits)**

Quantidade de memória ocupada pelo vetor em disco	Binário				Texto			
	Ponto Flutuante		Inteiro		Ponto Flutuante		Inteiro	
	Escrita	Leitura	Escrita	Leitura	Escrita	Leitura	Escrita	Leitura
500 kb	0.08	< 0.01	0.07	< 0.01	0.82	0.59	0.64	0.20
5 Mb	0.66	0.11	0.88	0.22	10.27	6.98	8.68	2.58
50 Mb	9.82	0.83	10.66	0.82	95.58	73.93	81.68	31.42
1Gb	192.4	15.97	208.7	15.4	1902.3	1450.9	1631.5	620.2

Conclui-se deste estudo que o fator dominante para a velocidade de acesso aos dados é o formato de armazenamento externo, texto ou binário, em que os mesmos são salvos. Afora este fator, os tempos de escrita para ambos os números inteiros e ponto flutuantes são aproximadamente os mesmos.

Para os tempos de escrita no dispositivo de armazenamento, o formato binário apresentou performance em média dez vezes superior ao formato texto. A diferença maior ocorre no acesso de leitura aos dados, o qual ocorre cerca de 100 vezes mais veloz quando os dados estão em formato binário, com exceção dos tempos de leitura dos números inteiros em formato texto, os quais são aproximadamente 60% menores dos esperados sendo, ainda, cerca de 40 vezes superiores.

O apêndice B mostra em detalhes como o padrão XDR foi implementado para permitir o cálculo e reutilização das matrizes em diferentes máquinas sem utilizar-se o lento formato texto.

### V.8.3 Interfaces básicas

Certos pontos da implementação do sistema computacional desenvolvido para este trabalho destacam-se como importantes contribuições que podem facilitar novas

implementações que venham a ser feitas no futuro. São os pontos fundamentais: o montador universal em blocos e as interfaces desenvolvidas.

Uma interface, no sentido emprestado pela linguagem Java é similar a um “contrato” que determinado objeto deve seguir para que este seja classificado como pertecendo a uma determinada classe. No sentido de programação, uma interface é tão somente a declaração de uma ou mais subrotinas, onde define-se o nome da rotina, argumentos de chamada e argumentos de retorno. Interface também pode ser razoavelmente definido como um conjunto de rotinas em torno de uma determinada finalidade.

A idéia por trás do conceito de interfaces é que outros objetos possam comunicar-se com uma extensa variedade de objetos, de variados tipos e tamanhos, desde que tais objetos “implementem” uma interface. O solver GMRES (SAAD, SCHULTZ, 1986) utilizado nesta tese, por exemplo, é um exemplo típico. Esta rotina aceita, como seu primeiro argumento, um ponteiro a qualquer objeto que implemente a interface “Operator”, que por sua vez define uma única rotina “Apply”. A idéia é que qualquer objeto da classe “Operator” seja capaz de aplicar-se a um determinado vetor. Isto permitirá que a mesma rotina resolva diferentes tipos de matriz, desde a clássica densa até a matriz wavelet comprimida.

O conceito de interface é mais geral que a linguagem C++ em si. De fato, o conceito de “contrato” pode ser implementado em C++ de várias formas e mesmo linguagem não orientadas a objetos, como o C e Fortran podem utilizá-la, talvez com um pouco mais de dificuldade. A linguagem Java possui suporte intrínseco a este conceito que hoje é a base de tecnologias como COM e CORBA (PRITCHARD, 1997) e de interfaces de produtividade em produção de software (RAD) como *Visual Basic* (PATTISON, 2000) e *Kylix* (WHIPPLE *et al*, 2001).

O apêndice A detalha as principais interfaces utilizadas para encapsular as principais características do método dos elementos de contorno.

## V.9 Aplicações

Um conjunto de problemas simples foi analisado para testar a eficiência da presente formulação. Todos os problemas resolvem a equação de Laplace em duas dimensões. Para manter o tratamento numérico tão simples quanto possível e para evitar a introdução de particularidades de elementos na formulação, o contorno foi discretizado

com elementos constantes – as propriedades e restrições dos quais são bastante conhecidas na literatura.

Na análise das aplicações que se seguem, a caracterização do erro será baseada no valor RMS (*Root Mean Square* ou raiz quadrada média), o qual é definido – para um vetor  $\mathbf{x}$  – como

$$RMS(\mathbf{x}) = \sqrt{\sum_{i=1}^N \frac{1}{N} x_i^2} \quad (119)$$

Usando esta medida, o erro relativo RMS entre um vetor de referência  $\mathbf{x}^{ref}$  e sua respectiva aproximação  $\mathbf{x}^{app}$  é dada por

$$RMSE(\mathbf{x}^{ref}, \mathbf{x}^{app}) = \frac{RMS(\mathbf{x}^{ref} - \mathbf{x}^{app})}{RMS(\mathbf{x}^{ref})} \quad (120)$$

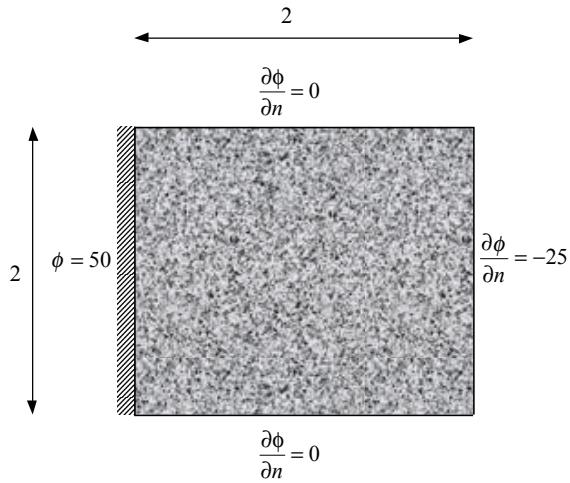
Definir a faixa aceitável do erro RMS é uma tarefa dependente da aplicação sendo estudada mas, para grande classe de aplicações em engenharia, valores menores que  $10^{-2}$  (1%) são satisfatórios. Para todas as aplicações numéricas, a solução de referência foi obtida usando um *solver* direto (Gauss) e matrizes completas sem compressão.

### V.9.1 Placa quadrada

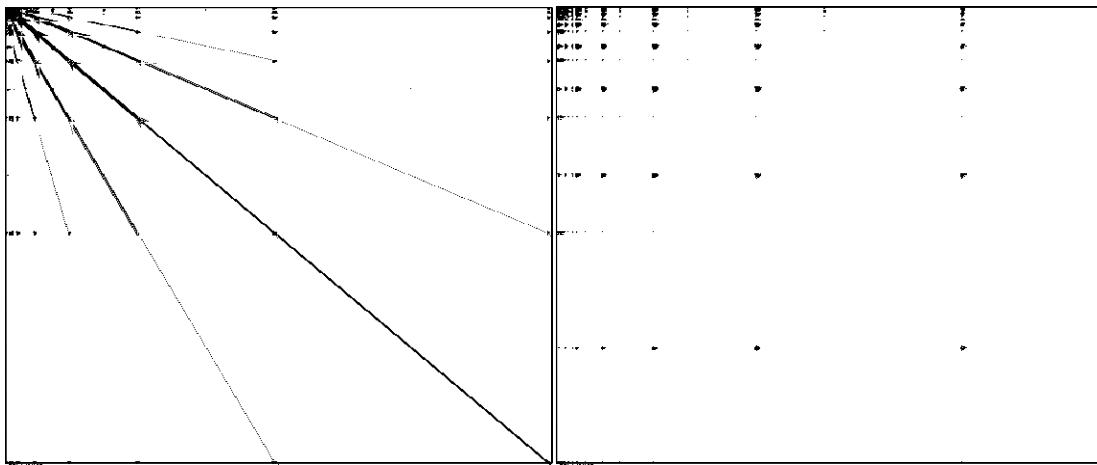
O primeiro problema resolve a equação de Laplace numa placa quadrada de lado 2, sujeita a um potencial 50V no lado esquerdo e diferentes fluxos nos demais lados – como mostrado na Figura 99.

Para este problema, a geometria foi discretizada com 4096 elementos constantes, 1024 em cada lado. A solução obtida para cada problema foi comparada com a solução padrão obtida usando-se Gauss, de tal forma que os erros medidos são somente devidos à compressão, não levando-se em consideração os erros introduzidos pela formulação em si ou mesmo a discretização.

A wavelet Daubechies com 10 coeficientes, a qual é apta para representar exatamente polinômios de graus até quarto, foi usada por apresentar, em média, o melhor aproveitamento para os casos estudados neste trabalho. O *solver* iterativo usado foi o GMRES (SAAD, SCHULTZ, 1986), com critério de parada sendo a norma residual relativa menor que  $10^{-16}$ . Não foram usadas matrizes residuais.



**Figura 99 Problema misto padrão, placa quadrada com fluxos prescritos em três lados e potencial constante prescrito no lado esquerdo**

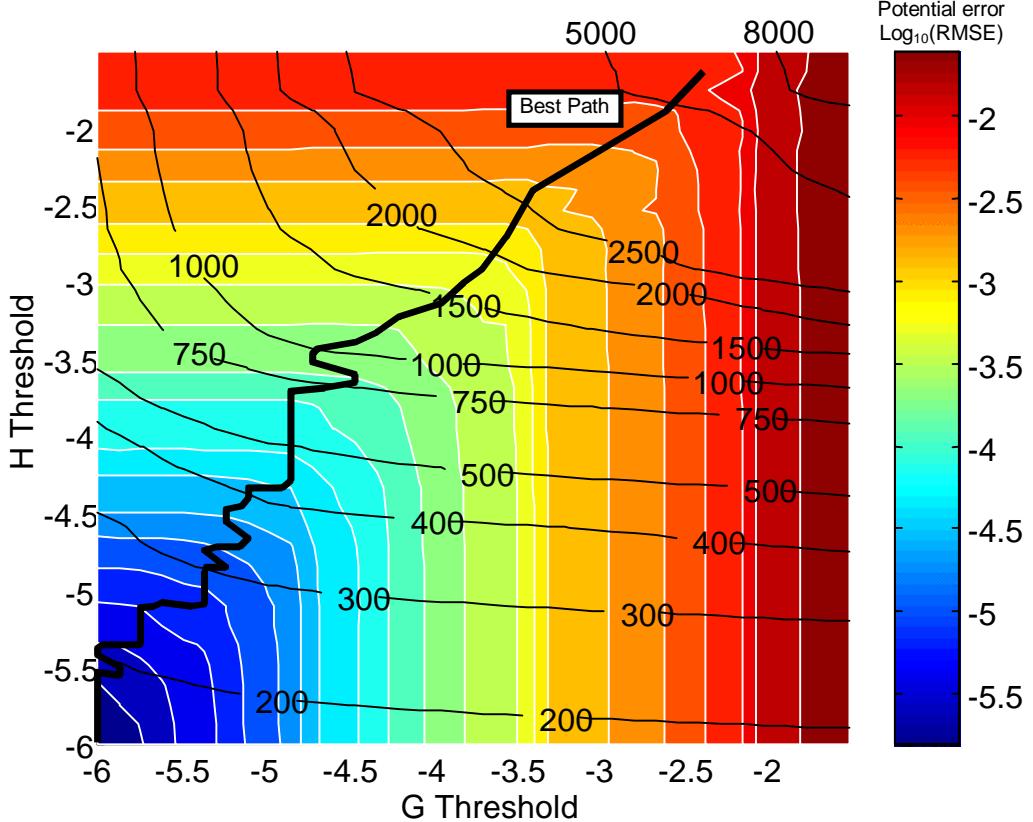


**Figura 100 Versões esparsas das matrizes  $G$  e  $H$ , respectivamente, geradas para o problema da placa quadrada. Somente elementos cujo módulo excede  $10^{-3}$  são mostrados em cor preta.**

A Figura 100 mostra as matrizes  $G$  e  $H$  após a compressão e esparsificação. Neste gráfico somente os elementos cujo módulo é maior que  $10^{-3}$  são mostrados em cor preta.

Para inserir as matrizes de elementos de contorno na presente formulação, é necessário prover ao *solver* parâmetros de compressão confiáveis (os *thresholds*). Está claro que quanto maior o valor dos *thresholds*, maior será o erro. Entretanto, se os *thresholds* fossem representados por um valor somente, seria fácil encontrar um valor que gere um determinado nível de erro usando alguma experiência a priori. No entanto, o problema é que existem dois valores de *threshold* independentes, um para cada matriz e a questão principal que se coloca é como o erro da solução final comporta-se para diferentes valores de *thresholds*.

Com esta questão em mente, um conjunto de 1600 problemas, cada qual representando um par de *thresholds* ( $\alpha_G, \alpha_H$ ), foi resolvido. Cada *threshold* cobre, em progressão geométrica, a faixa de  $10^{-6}$  a  $10^{-1}$ .



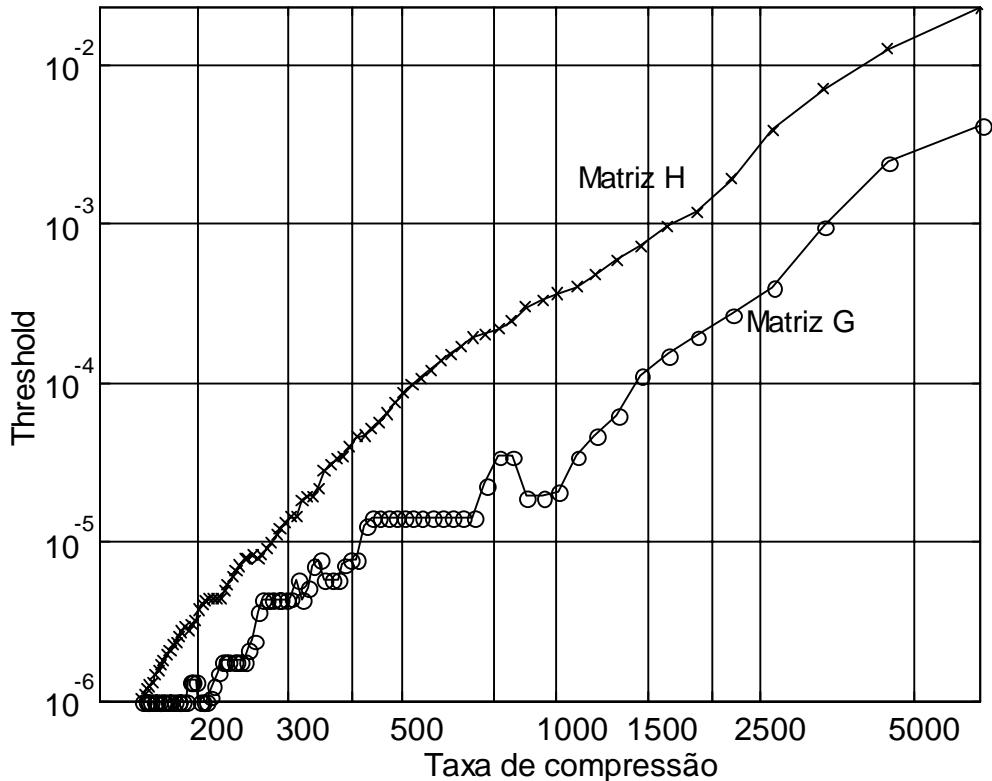
**Figura 101 Problema da placa quadrada: erro RMSE (potencial) como função dos valores de threshold  $G$  e  $H$  (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)**

A Figura 101 mostra o erro RMSE resultante para a solução do potencial, o qual é plotado em curvas de nível preenchidas em cor. Superposta, em curvas de nível em preto somente, está a taxa de compressão  $CR$ , definida como a relação entre a quantidade de memória requerida para armazenar as matrizes  $G$  e  $H$  e a quantidade de memória exigida por todas as matrizes esparsas geradas, incluindo seus respectivos índices e quaisquer estruturas adicionais necessárias.

A primeira conclusão importante obtida da Figura 101 é que, se uma das matrizes já está mal-representada então o aumento da precisão da outra matriz é totalmente inútil. Isto é verificado pela forma de degrau do gráfico do erro nas regiões distantes das diagonais.

A segunda conclusão importante é que as curvas de nível referentes à taxa de compressão tendem a ser perpendiculares às curvas de erro em todos os pontos, o que é de certa forma esperado. O significado deste comportamento é que qualquer ganho em

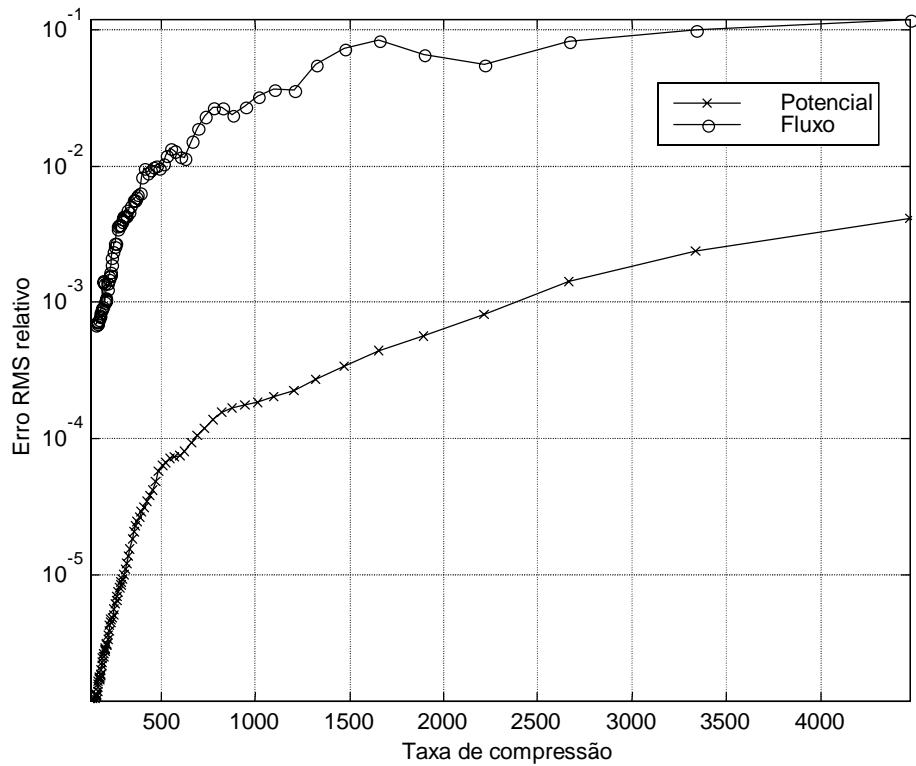
compressão será sempre seguido por uma perda de qualidade da solução. Perto das diagonais, no entanto, esta perpendicularidade não é clara e, por este motivo, a melhor combinação de thresholds que minimiza o erro para uma taxa de compressão não é uma linha reta mas um caminho angular representado por uma linha espessa na Figura 101. Os valores de threshold obtidos ao longo deste caminho são mostrados na Figura 102.



**Figura 102 Problema da placa quadrada. Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)**

Como pode-se observar, as duas curvas que aparecem na Figura 102 têm uma clara correlação. Este comportamento, o qual aparece em todos os experimentos neste trabalho, é a chave para definir-se os *thresholds* para novos problemas.

Supondo-se que estes dois valores de *threshold* são proporcionais, pode-se então realizar uma regressão estatística – em escala logarítmica – para definir os parâmetros de compressão que minimizem o erro final. Para a placa quadrada a relação entre os thresholds das matrizes **H** e **G** que minimizam o erro é 0.256. Desta forma, fazendo-se  $\alpha_G = 0.256\alpha_H$  tem o efeito de manter a solução perto do melhor caminho mostrado na Figura 101, otimizando ambos o erro e compressão ao mesmo tempo. Os erros obtidos para o caminho ótimo são mostrados na Figura 103.



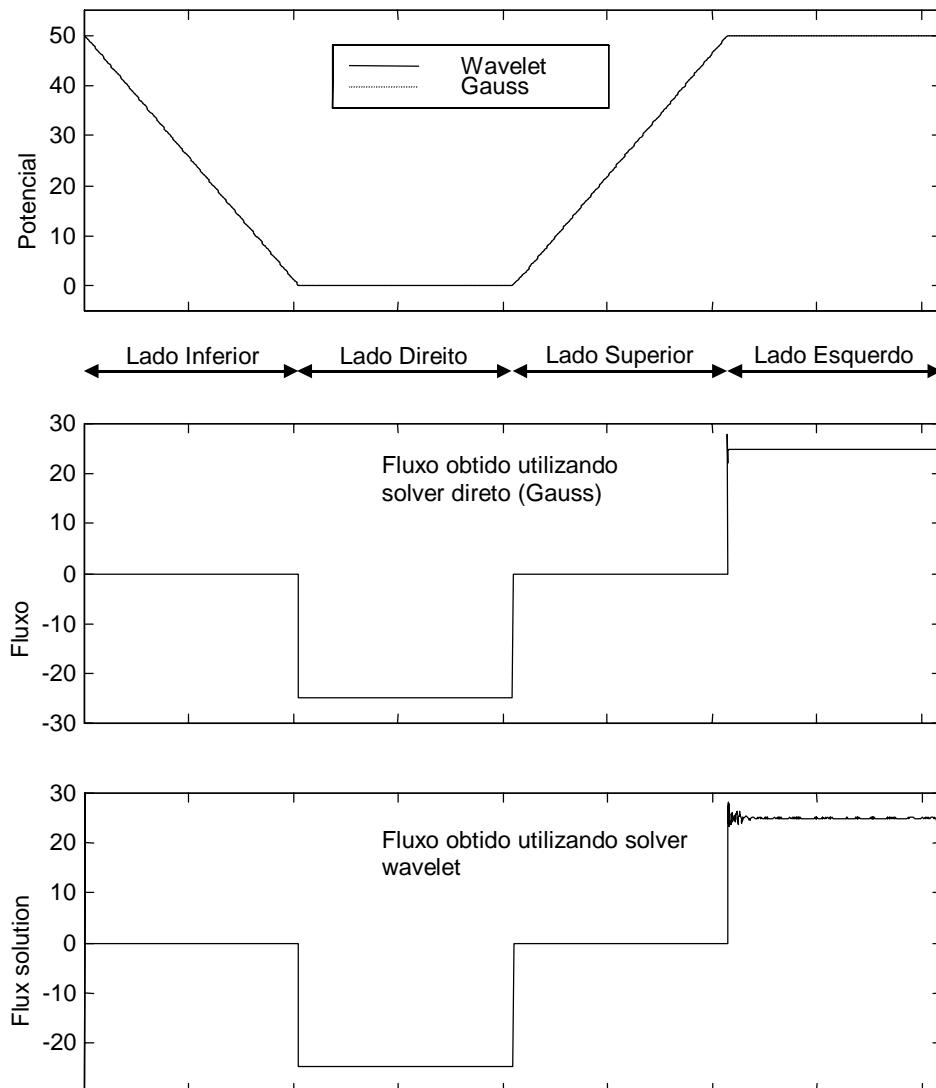
**Figura 103 Problema da placa quadrada: erros do fluxo e potencial como função da taxa de compressão**

Um comportamento notável da evolução deste erro é que existe um grande aumento até a taxa de compressão 500. Após este ponto, a evolução é menos rápida mas o erro do fluxo já está no limite imposto de 1%. Este resultado particular foi o pior de todos os exemplos e é parcialmente explicado pelo conhecido mal condicionamento da matriz  $G$  para a equação de Laplace em duas dimensões, o que pode fazer o *solver* mais robusto como Gauss falhar em fornecer uma solução aceitável.

De fato, a matriz do sistema é uma combinação das matrizes  $G$  e  $H$ , e a contribuição de cada é diretamente dependente das condições de contorno. Na medida que o número de nós com potencial prescrito cresce, a matriz do sistema resultante torna-se mais e mais mal-condicionada. Para todos os casos onde o número de nós com potencial prescrito é menor ou as condições de contorno são combinações lineares de potencial e fluxo, como é o caso de todos os outros exemplos deste trabalho, os erros são perfeitamente bem comportados.

Deve ser ressaltado que estas instabilidades do fluxo só ocorrem a taxas de compressão bastante altas, acima de 500 neste problema particular. Quando olha-se a faixa de aplicações de elementos de contorno em problemas de engenharia, este pode ser considerado um caso especial. Por outro lado, se a solução do potencial é o principal

foco dos resultados, é possível usar a faixa completa de compressão – acima de 5000 – e ainda assim continuar a obter soluções satisfatórias.



**Figura 104 Solução do problema da placa quadrada; taxa de compressão igual a 440**

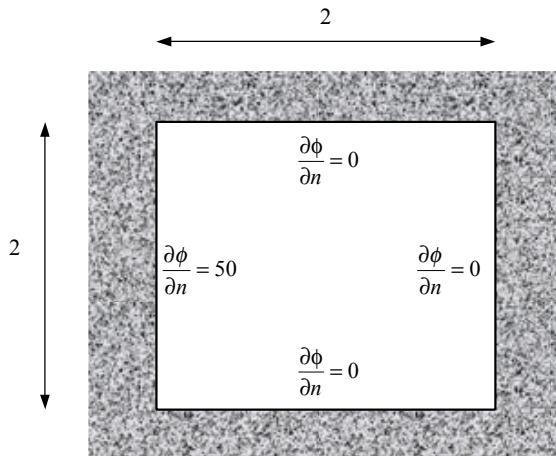
A Figura 104 mostra um resultado típico para o problema do potencial. Para esta solução particular, *thresholds*  $10^{-1}$  ( $H$ ) e  $0.256 \times 10^{-1}$  ( $G$ ) foram usados, resultando em uma taxa de compressão de 440. Na figura superior é mostrada a solução para o potencial com ambos os resultados – de referência e comprimido – plotados superpostos. A figura do meio mostra o resultado de referência para a solução do fluxo e a figura inferior mostra o resultado para a solução do fluxo utilizando compressão.

As principais características apresentadas nesta figura são os picos nas arestas das interseções entre os lados superior e esquerdo. Note-se que mesmo a solução do *solver* direto (Gauss) apresenta uma descontinuidade muito forte para o fluxo no elemento

interfaceando os lados superior e esquerdo. Este é uma característica muito bem conhecida que aparece devido ao uso de elementos constantes.

### V.9.2 Placa infinita com cavidade (problema externo)

O segundo problema, mostrado na Figura 10, é geometricamente igual ao primeiro mas representa um problema externo de uma cavidade em um domínio infinito. Condições de contorno são prescritas como fluxos prescritos em todos os lados da cavidade.

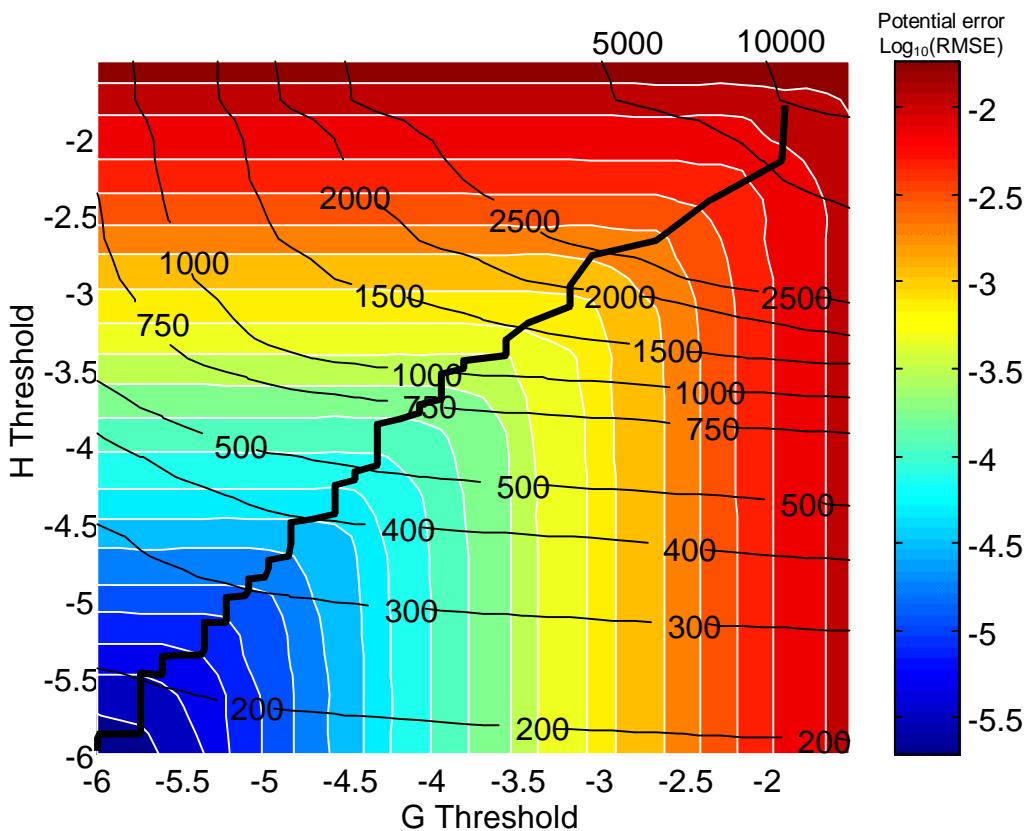


**Figura 105 Cavidade quadrada com fluxos prescritos em todos os lados**

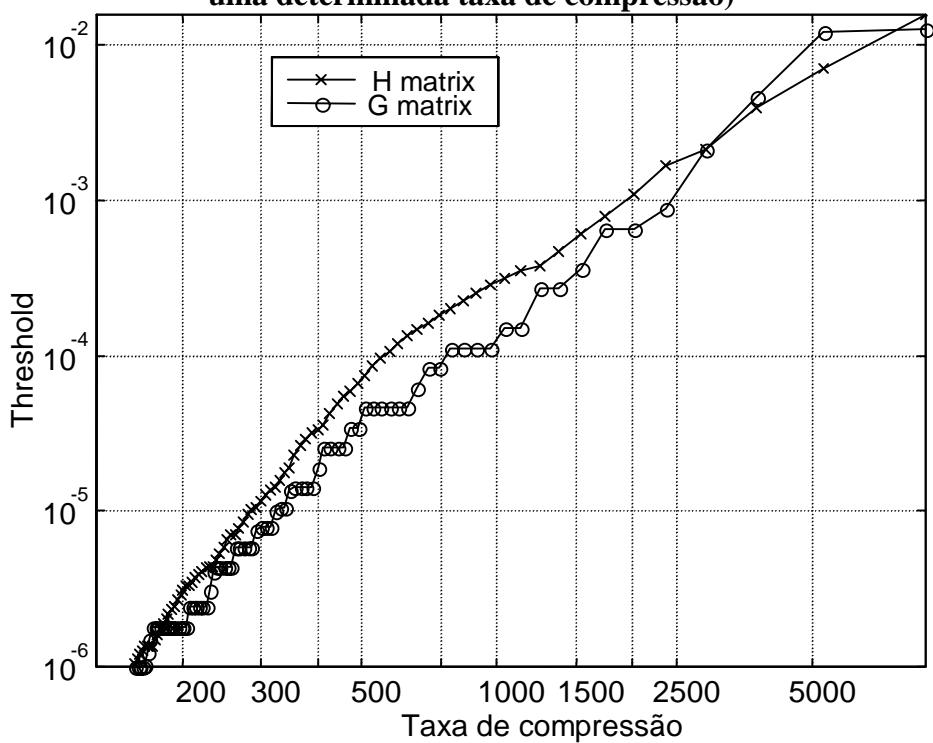
Este problema apresenta os melhores resultados obtidos para todos os experimentos numéricos realizados. A razão é que a matriz do sistema resultante é a matriz  $\mathbf{H}$ , a qual é reconhecidamente bem condicionada e diagonal dominante.

A Figura 106 mostra as curvas de nível para o erro e compressão superpostos, similar ao já mostrado para o exemplo anterior (Figura 101). Comparando-se os dois gráficos, pode ser visto que neste experimento os erros do potencial são bem menores para as mesmas taxas de compressão.

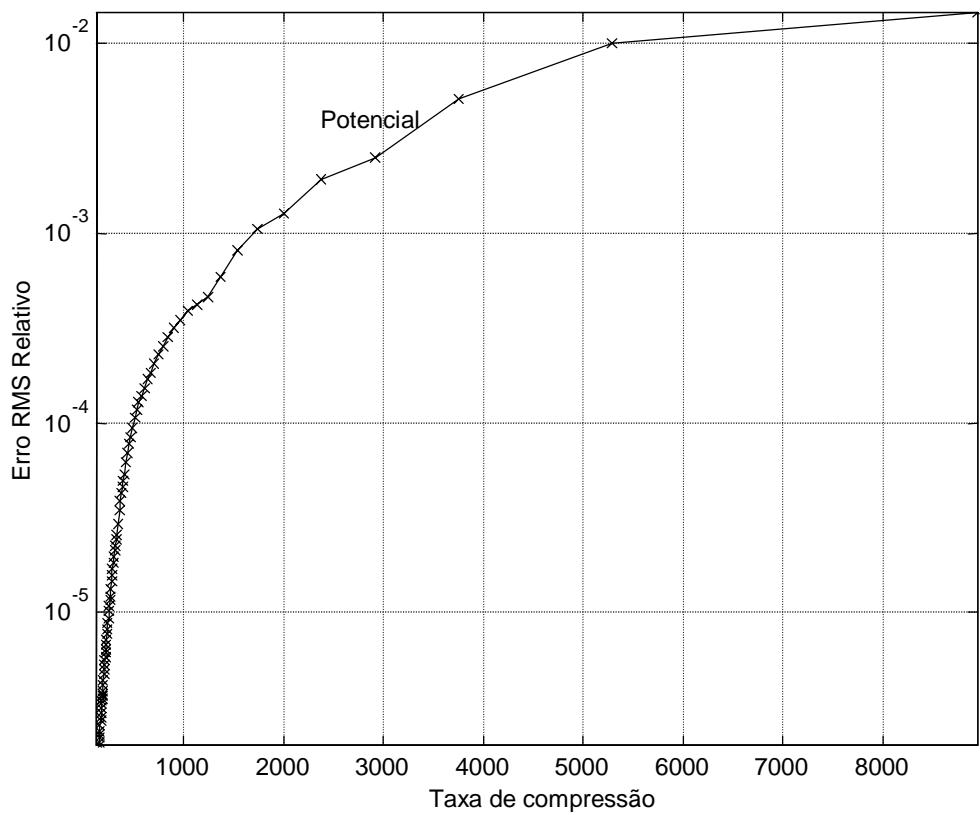
Uma regressão linear realizada nos valores mostrados na Figura 107 em escala logarítmica revela que os *thresholds* das matrizes  $\mathbf{G}$  and  $\mathbf{H}$  obtidos para o melhor caminho são empiricamente correlacionados como  $\alpha_G = 0.664\alpha_H$ . Para outros problemas, no entanto, é necessária uma investigação teórica mais aprofundada para obter os valores de *threshold* a priori. Não sendo isto possível, é necessário comprimir-se as matrizes utilizando *thresholds* bem conservadores e então fazer-se alguns testes variando-se um *threshold* por vez para determinar qual deles permite ser relaxado sem introduzir-se erros significantes na solução.



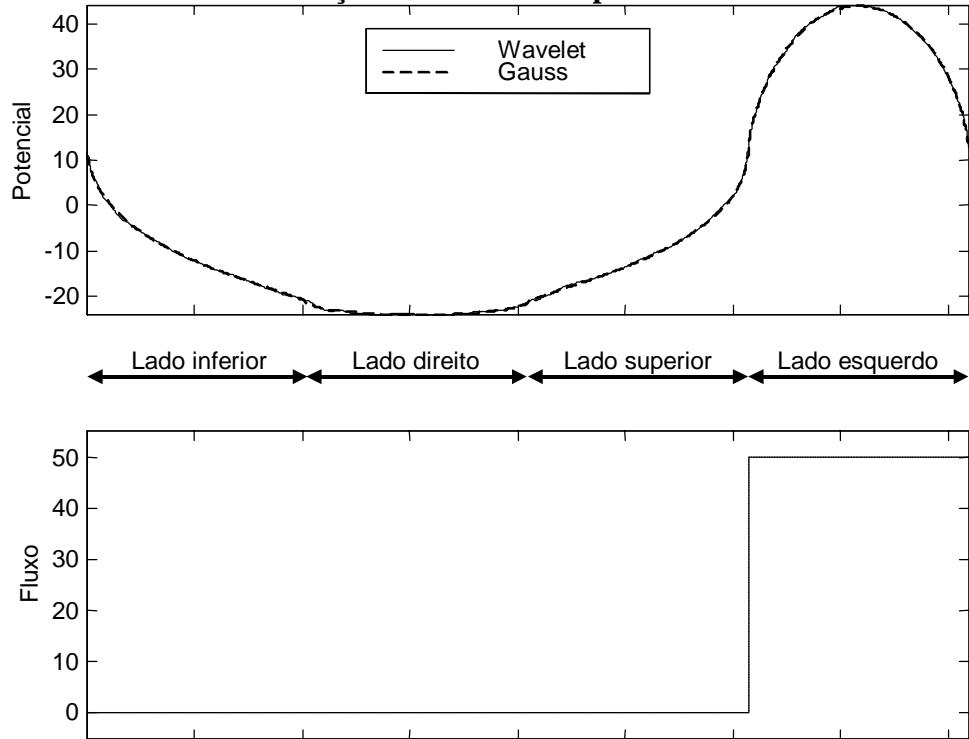
**Figura 106** Placa infinita com cavidade: erro RMSE (potencial) como função dos valores de threshold G e H (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)



**Figura 107** Problema da placa infinita com cavidade: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)



**Figura 108 Problema da placa infinita com cavidade: erro do potencial como função da taxa de compressão**



**Figura 109 Problema da placa infinita com cavidade: soluções do fluxo e potencial, com e sem compressão; lado inferior; taxa de compressão resultante é 5015**

A Figura 108 mostra o erro obtido para o potencial seguindo o caminho ótimo. Como os fluxos são conhecidos, seus respectivos erros são zero e portanto não foram plotados. Como a figura revela, taxas de compressão acima de 5500 podem ser utilizadas com erros relativos nunca maiores que 1% sendo introduzidos na solução. Se um baixo nível de erro é necessário, 0.1% por exemplo, então taxas de compressão até 1700 podem ainda ser alcançadas.

A Figura 109 mostra uma solução típica obtida para o problema externo, com taxa de compressão de 5015. Como a curva do topo mostra, a solução do potencial é quase indistinguível da solução direta e a diferença só pode ser observada numericamente. A solução do fluxo prescrito ao longo do contorno também foi plotada por efeito de complementação.

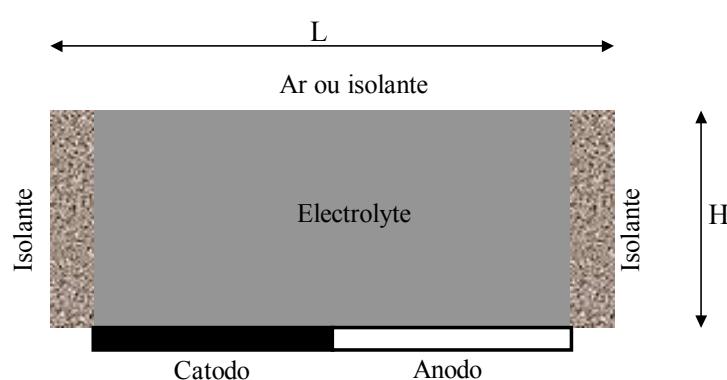
### V.9.3 Par Galvânico

Este exemplo interessante estuda a transmissão de corrente através de um eletrólito ao qual é aplicado um par galvânico consistindo de duas faixas semi-infinitas como mostrado na Figura 110 (AOKI et al, 1985). A descrição matemática deste problema é dada pela equação de Laplace com condições de contorno lineares. As condições de contorno nos isolantes são dadas por fluxo (corrente) zero enquanto curvas de polarização lineares para o catodo e anodo são dadas pelas seguintes fórmulas

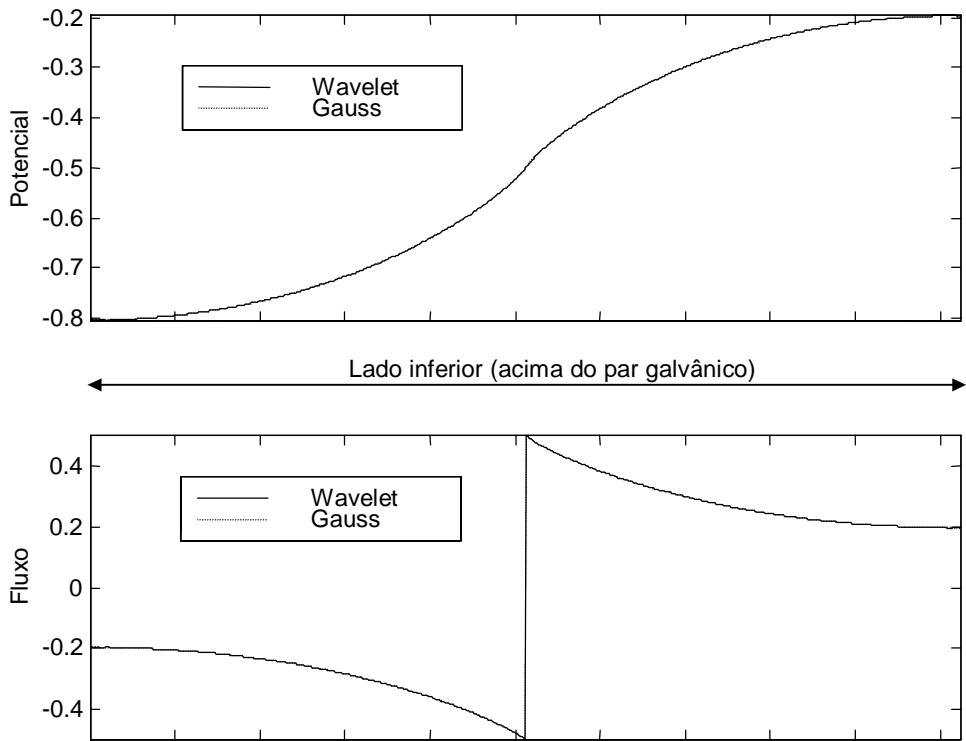
$$u - \alpha \cdot q = 0 \text{ (Catodo)} \quad (121)$$

$$u - \alpha \cdot q = u_0 \text{ (Anodo)} \quad (122)$$

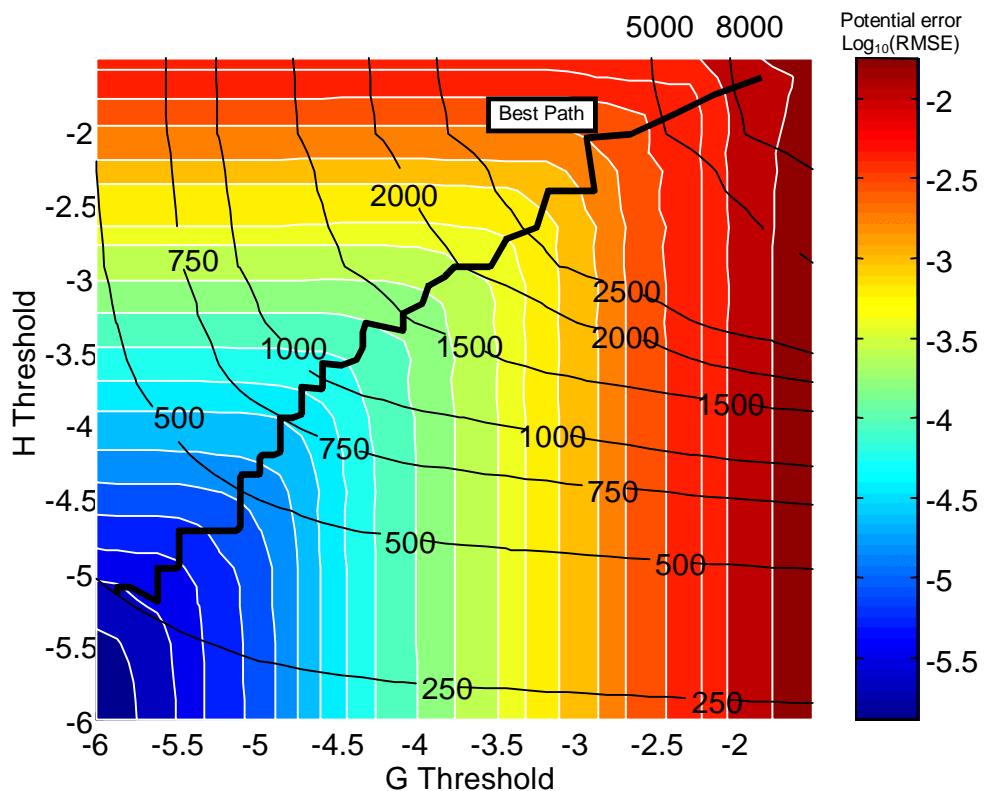
onde  $u_0$  é o potencial no anodo extrapolado para a corrente zero e  $\alpha$  é o parâmetro de polarização ( $\partial u / \partial q$ ), assumido ser igual para ambos anodo e catodo. A relação entre os lados (H/L) foi escolhida ser igual a 0.1.



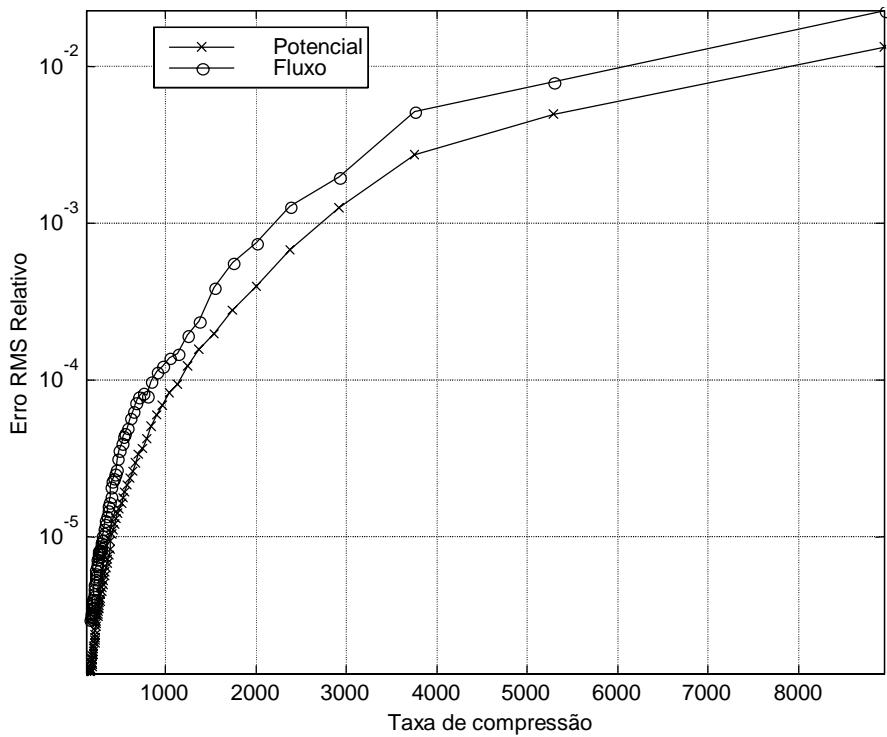
**Figura 110 Modelo de corrosão galvânica**



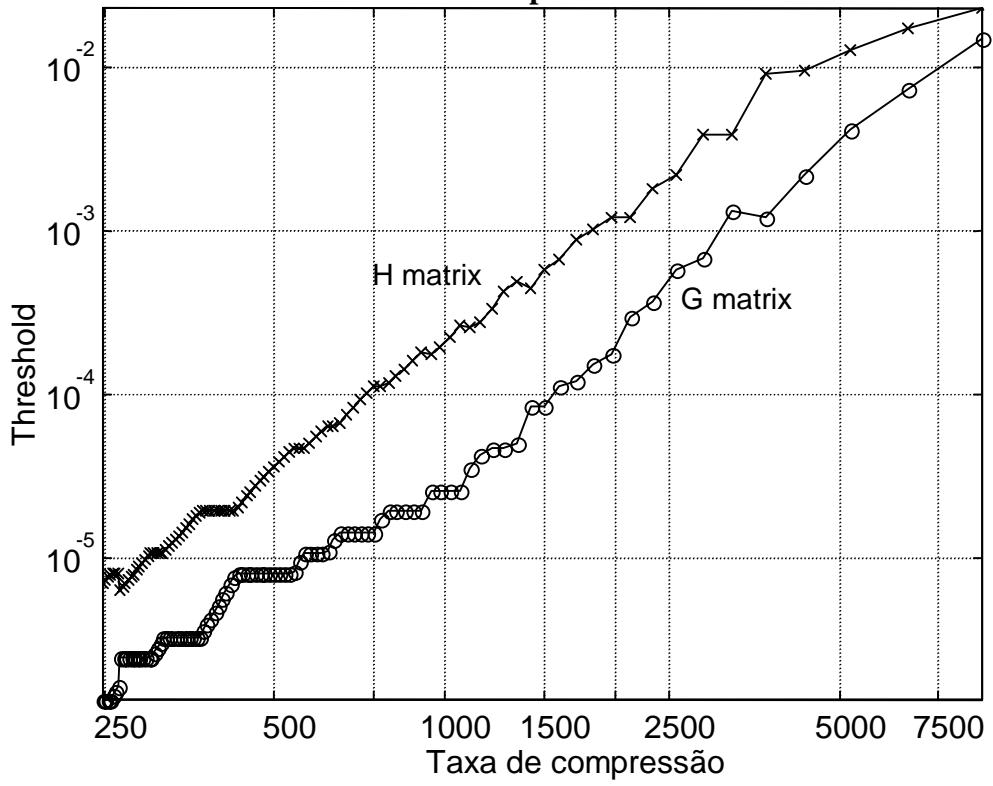
**Figura 111 Problema do par galvânico: soluções para o fluxo e potencial, com e sem compressão ao longo do par galvânico (lado inferior); taxa de compressão obtida igual a 1220**



**Figura 112 Problema do par catódico: erro RMSE (potencial) como função dos valores de threshold G e H (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho (erro mínimo para uma determinada taxa de compressão)**



**Figura 113 Problema do par galvânico: erros do fluxo e potencial como função da taxa de compressão**



**Figura 114 Problema do par galvânico: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)**

Uma solução típica para o potencial e densidade de corrente (fluxo) ao longo do par galvânico é mostrada na Figura 111. Nota-se que ambas as soluções são bem

comportadas e – para uma taxa de compressão de 1220 – a diferença entre os gráficos de compressão e a solução direta é quase indistinguível.

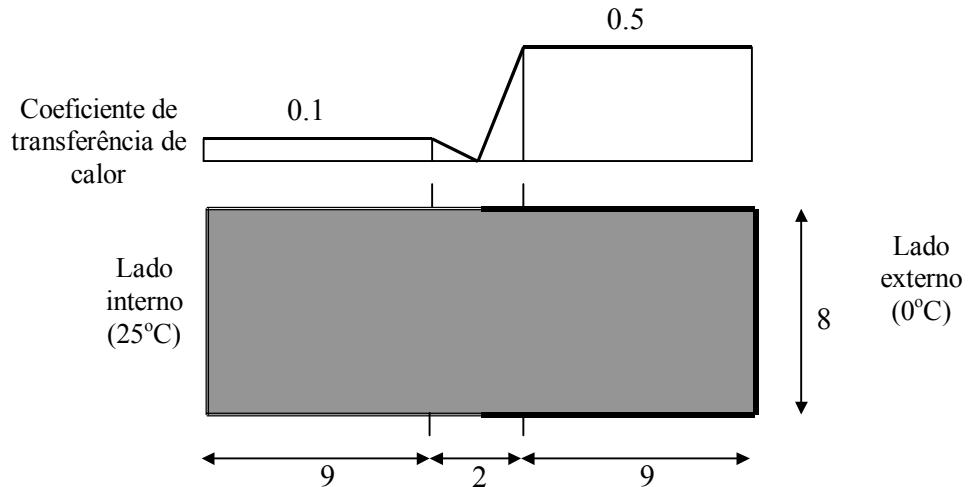
A Figura 112 mostra o gráfico de erro-compressão, que apresenta a forma usual. Os erros do fluxo e potencial ao longo do caminho ótimo (erro mínimo para determinada taxa de compressão) são mostrados na Figura 113 onde pode ser visto que ambos os erros são bem comportados, permitindo taxas de compressão da ordem de 6000 com erros RMSE menores que  $10^{-2}$ . Figura 114 mostra os *thresholds* obtidos ao longo do melhor caminho, com a regressão linear revelando boa correlação entre ambos valores na forma de  $\alpha_G = 0.272\alpha_H$ . Esta constante relaciona-se tanto com a ordem de grandeza dos coeficientes das matrizes  $G$  e  $H$  como também com as próprias condições de contorno prescritas.

#### V.9.4 Coluna de concreto

Neste exemplo estuda-se uma coluna de concreto sujeita a um fluxo de calor dependente da temperatura em cada lado, como mostrado na Figura 115. No contorno, o fluxo de calor ( $q$ ) é proporcional à diferença de temperatura entre a superfície da coluna ( $u$ ) e o ambiente ( $u_s$ ). Esta condição de contorno pode ser expressa por

$$q = -h(u - u_s) \quad (123)$$

onde  $h$  é o coeficiente de transferência de calor entre a viga e o ar.



**Figura 115 Geometria e condições de contorno para o problema da coluna de concreto**

O ambiente interior tem uma temperatura interna de  $25^\circ\text{C}$ , e nesta região a viga supostamente recebeu um tratamento que diminuiu o coeficiente de transmissão de calor para  $0.1 \text{ cal/m}^2/\text{s}^\circ\text{C}$ . O ambiente externo está a uma temperatura constante de  $0^\circ\text{C}$  e sua

superfície está sujeita à convecção forte com um coeficiente de transferência de calor igual a  $0.5 \text{ cal/m}^2/\text{s}/^\circ\text{C}$ . O valor do coeficiente de transferência de temperatura decresce linearmente até zero na região média – melhor isolada. As condições de contorno dadas por (123) são representadas na técnica da montagem virtual como

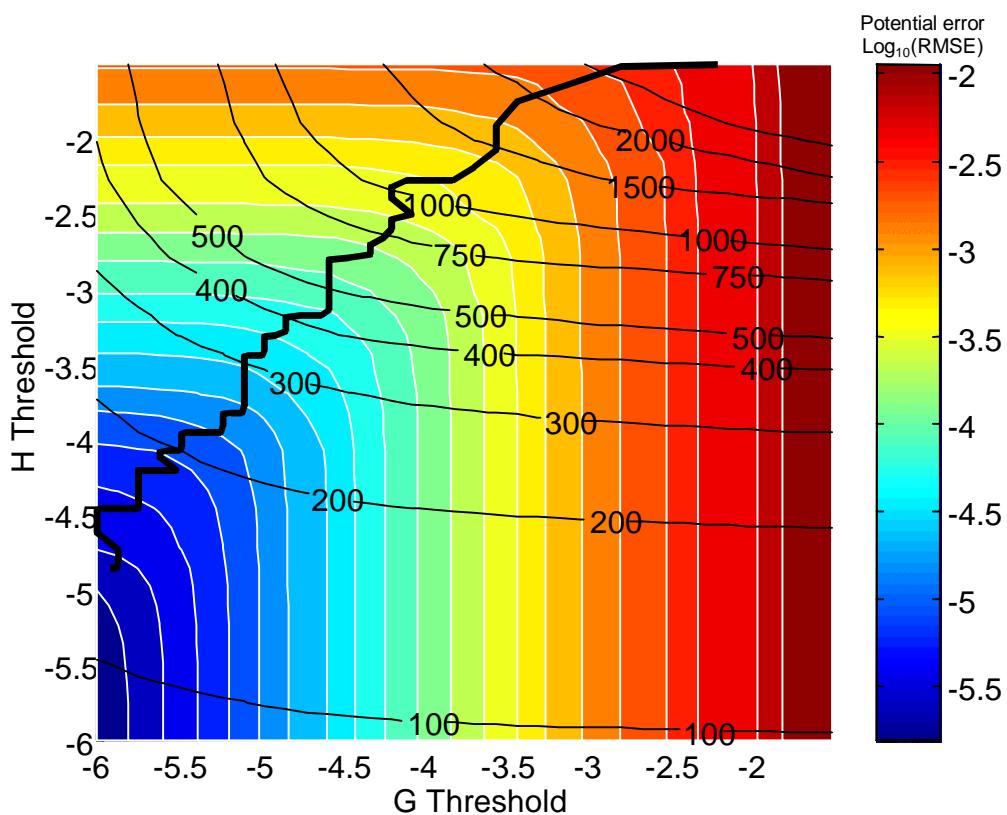
$$\begin{aligned}\lambda^H x &= x, \quad u_0 = 0 \\ \lambda^G x &= -hx, \quad q_0 = hu_s\end{aligned}\tag{124}$$

Seguindo o mesmo procedimento do primeiro teste, o terceiro conjunto de problemas foi resolvido para uma combinação de 1600 pares de *threshold* diferentes, cada qual variando na faixa de  $10^{-6}$  a  $10^{-1}$ . O gráfico de análise erro-compressão para este problema é mostrado na Figura 116. Verifica-se nesse gráfico que o caminho ótimo está bem mais acima que nos casos anteriores, isto é, para minimizar-se o erro deve-se adotar para o *threshold* da matriz  $\mathbf{H}$  um valor bem superior que o *threshold* da matriz  $\mathbf{G}$ . A Figura 116 mostra que, para alguns casos, a taxa de compressão pode ser bastante aumentada se algum tipo de informação a priori puder ser obtida.

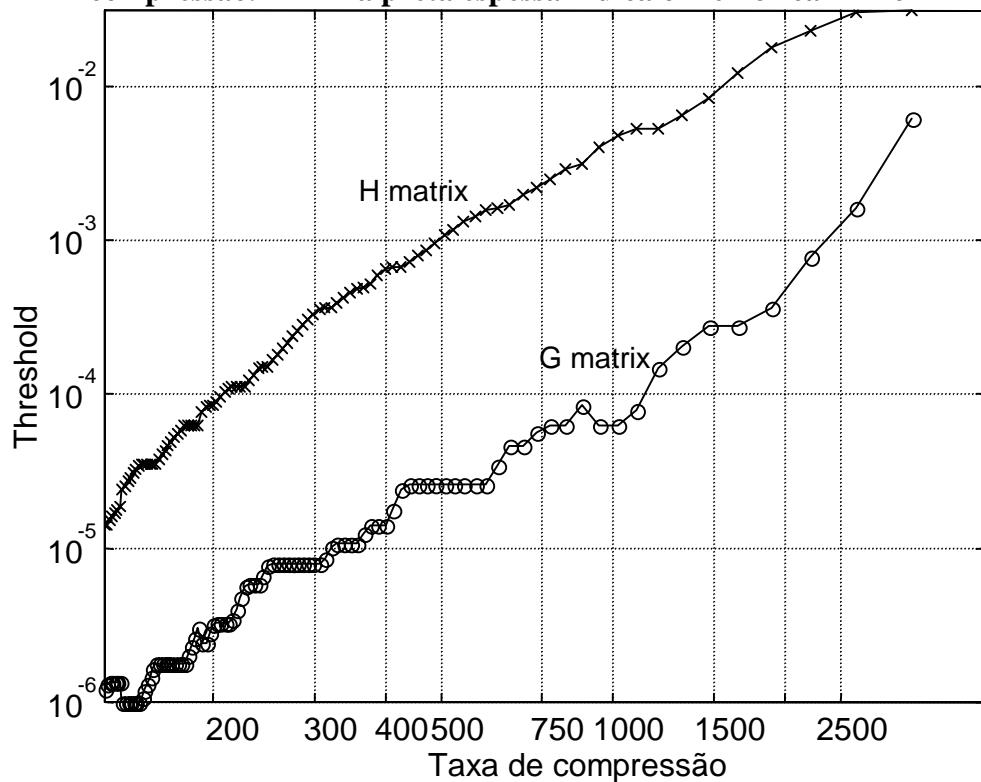
Na Figura 117, onde são mostrados os valores de *threshold* obtidos para o caminho ótimo (erro mínimo para uma dada taxa de compressão), regressão linear revela uma boa correlação entre os valores na forma  $\alpha_G = 0.0314\alpha_H$ . Esta constante bem menor que a unidade mostra que a adoção de valores idênticos para ambos os *thresholds* pode resultar em perda de tempo de computação bem como de memória.

A Figura 118 mostra a evolução do erro para ambos fluxo e potencial como função da taxa de compressão. Como no primeiro modelo, a solução do potencial apresenta um erro mais bem comportado que o fluxo. Para o erro especificado de 1%, taxas de compressão de 2000 podem ser usadas. Se o principal foco da análise é a solução do potencial então taxas de compressão maiores que 5000 são possíveis.

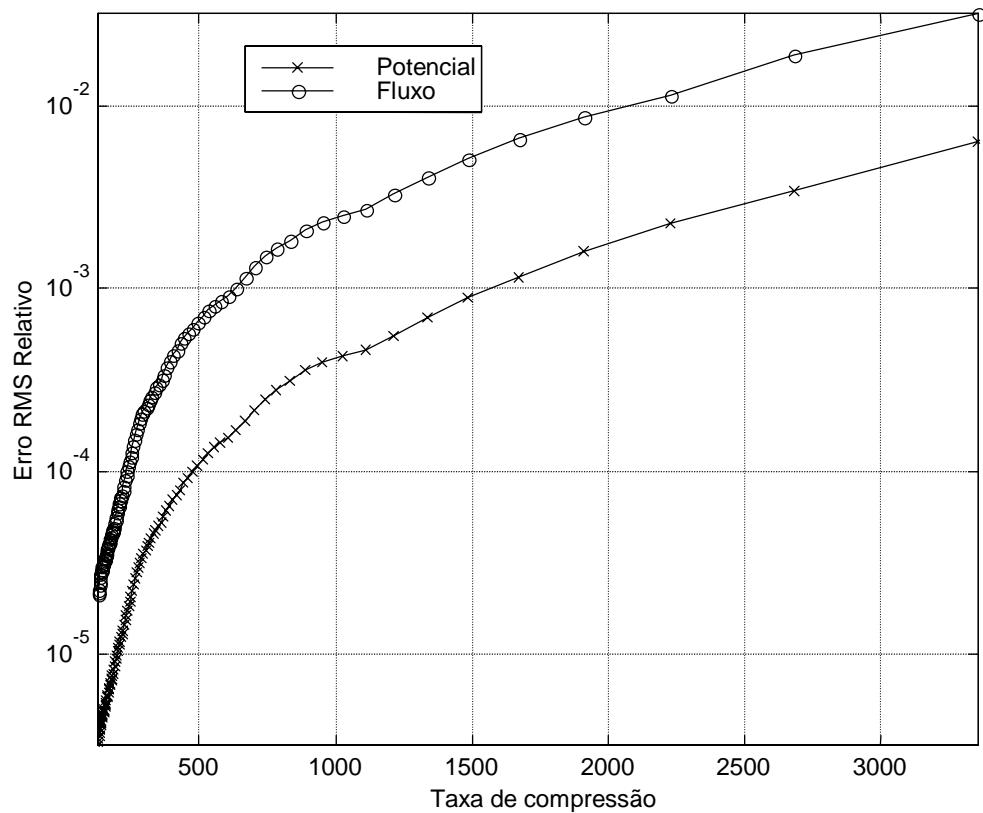
A Figura 119 mostra uma solução típica obtida para este problema. Para este caso particular, a taxa de compressão obtida foi igual a 670 e os principais efeitos da compressão são notados como pequenos distúrbios nos extremos de cada lado. Estes distúrbios são resultados de fortes discontinuidades geradas na matriz  $\mathbf{H}$  devido à rápida variação das normais envolvendo o cálculo dos coeficientes desta matriz. Para taxas de compressão maiores que 500, entretanto, a solução aproximada e a direta são quase indistinguíveis.



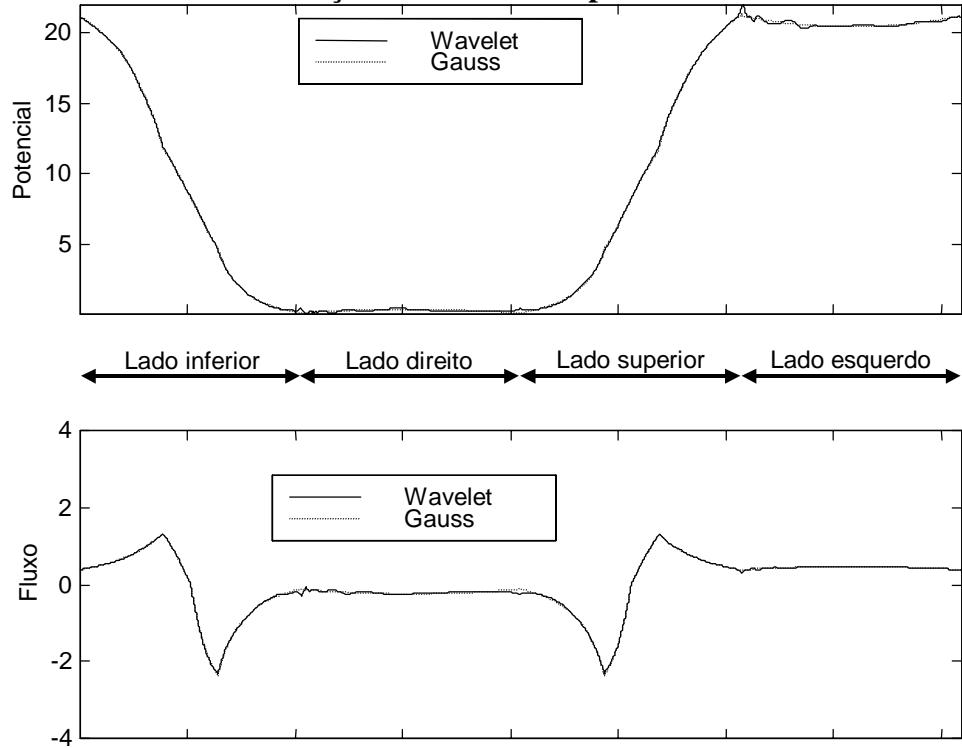
**Figura 116** Problema da coluna de concreto: erro RMSE (potencial) como função dos valores de threshold G e H (em cor). Em preto, curvas para iguais valores de compressão. A linha preta espessa indica o melhor caminho



**Figura 117** Problema da coluna de concreto: Valores de threshold seguindo o melhor caminho (erro mínimo para uma determinada taxa de compressão)



**Figura 118 Problema da coluna de concreto: erros do fluxo e potencial como função da taxa de compressão**



**Figura 119 Problema da coluna de concreto: soluções aproximadas para o potencial e fluxo, com e sem compressão; taxa de compressão igual a 670**

A principal vantagem da presente implementação é que as mesmas matrizes podem ser usadas na forma comprimida para resolver virtualmente qualquer tipo de condição de contorno. Note-se que, devido ao tipo de condição de contorno de convecção do presente problema, a montagem completa, tradicional, produziria o seguinte sistema de equações:

$$(\mathbf{H} + \mathbf{G}\mathbf{D})\mathbf{u} = \mathbf{G}\mathbf{D}\mathbf{e} \quad (125)$$

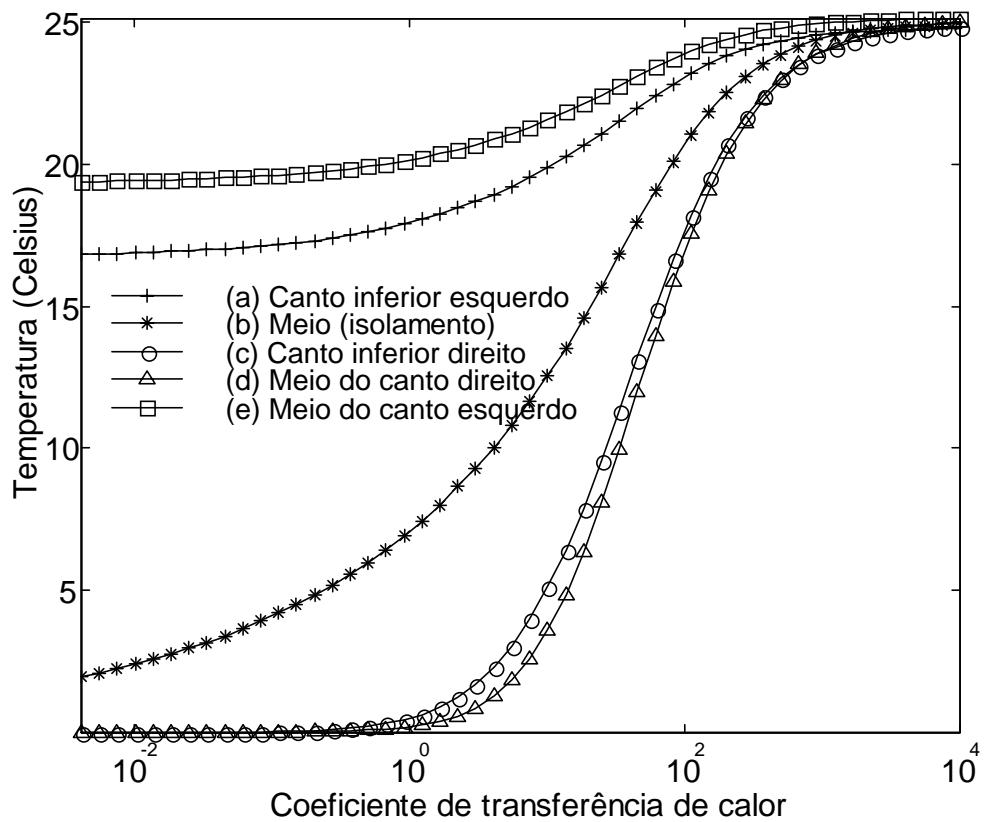
no qual  $\mathbf{D}$  é uma matriz diagonal contendo os valores do coeficiente de transferência de calor  $h$  e  $\mathbf{e}$  é um vetor contendo as temperaturas ambiente  $u_s$ . Portanto, se os valores de  $h$  variarem, a matriz do sistema deve ser comprimida novamente para cada caso de carga diferente.

O procedimento de montagem virtual evita a recompressão da matriz do sistema por trabalhar diretamente com a forma comprimida das matrizes de elemento de contorno  $\mathbf{H}$  e  $\mathbf{G}$ . Para ilustrar isto, a Figura 120 mostra as temperaturas obtidas em cinco diferentes pontos ao longo do contorno para um conjunto de 50 diferentes condições de contorno geradas através da mudança sucessiva de coeficientes de troca de calor no lado externo, desta forma representando vários graus de isolamento aplicados à superfície externa.

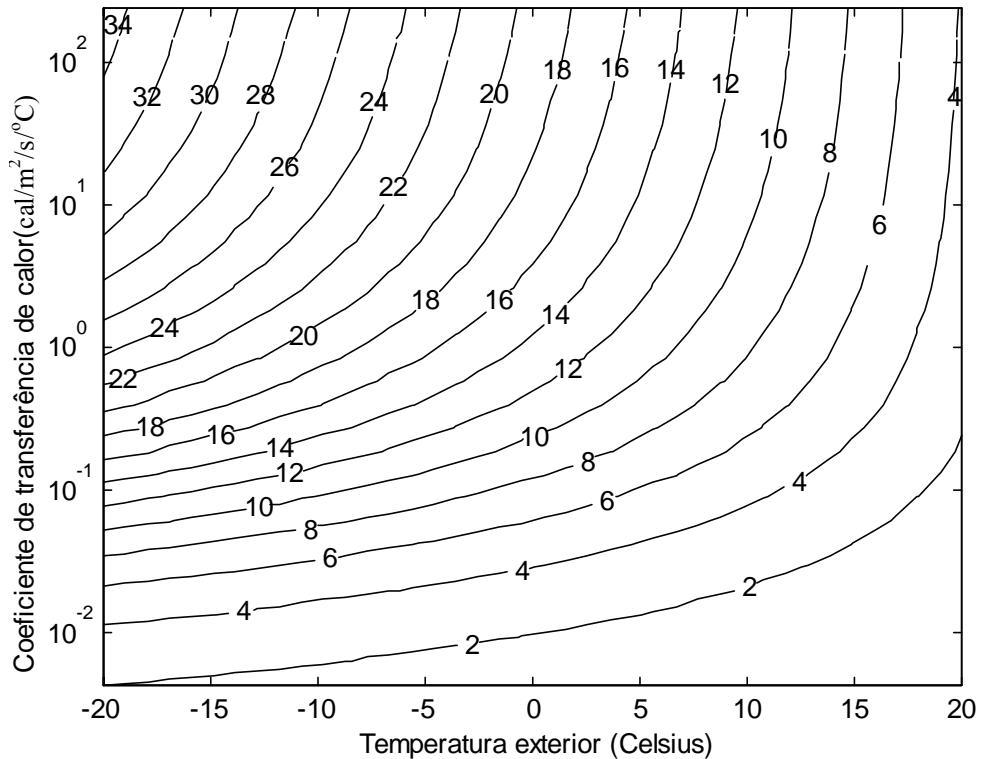
Uma vez que as matrizes foram comprimidas e armazenadas em um arquivo, a solução foi obtida lendo-se as matrizes de volta para a memória e rodando-se diretamente o solver iterativo sobre elas. Cada solução foi obtida muito rapidamente, desta forma permitindo a construção fácil e eficiente de gráficos para múltiplos carregamentos.

A Figura 121 serve para acentuar ainda mais esta característica. Para esta figura, 2500 modelos foram gerados, cada um representando uma combinação de 50 diferentes temperaturas externas e 50 diferentes coeficientes de transferência de calor. Para cada solução, o fluxo de calor total entre a coluna e o ambiente interno foi obtido e posteriormente o conjunto de valores obtidos foi plotado na forma de um gráfico de contorno. Não houve necessidade de recomprimir as matrizes desta vez porque as mesmas matrizes comprimidas, usadas para calcular o gráfico anterior, foram reutilizadas.

O gráfico da Figura 121 foi produzido em um microcomputador AMD K7 Athlon 1.2GHz com 768Mb RAM e exigiu menos que cinco minutos para resolver todos os 2500 carregamentos, cada um envolvendo a solução de um problema de 2048 graus de liberdade.



**Figura 120** Temperaturas obtidas em vários pontos do contorno como uma função de diferentes condições de isolamento na superfície exterior.



**Figura 121** Fluxo de calor total trocado por unidade de tempo entre a viga de concreto e o ambiente interior como uma função da temperatura exterior e do coeficiente de transferência de calor da superfície externa

### V.9.5 Estudo de escalamento

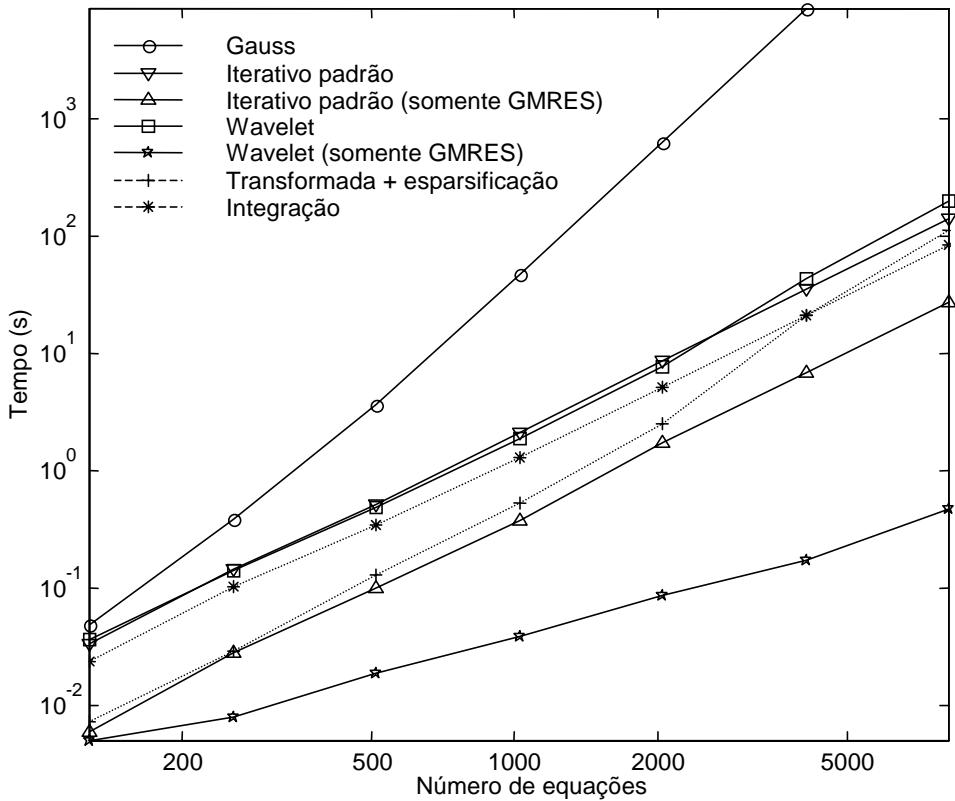
Para estudar como o método se comporta com o aumento sucessivo do tamanho do número de graus de liberdade do modelo, um conjunto de testes foi realizado onde a geometria do problema anterior (Figura 115) foi utilizada, mas agora o problema foi discretizado com diferentes números de elementos, de 128 a 8192, totalizando 7 modelos diferentes.

Cada modelo foi resolvido com o solver padrão Gauss, com o solver iterativo padrão utilizando as matrizes densas e com o solver iterativo utilizando as matrizes comprimidas pela transformada wavelet. O tempo gasto com a integração das matrizes densas e também o tempo adicional requerido pelo solver wavelet para comprimir estas matrizes foram medidos com o propósito de comparação.

Os valores de limiar de compressão foram controlados de tal forma que o erro RMS do potencial não foi permitido exceder  $10^{-3}$  e o erro RMS do fluxo não foi permitido exceder  $10^{-2}$  (1%). O solver iterativo GMRES foi usado e o critério de parada foi definido tal que a norma residual fosse menor que  $10^{-10}$ . Todos os modelos foram resolvidos em um microcomputador AMD K7 (Athlon) 1.2GHz com 768Mb RAM. Um cronômetro multimídia com 1 milissegundo de resolução foi usado para obter-se maior acurácia nos testes com os modelos menores. Os resultados estão mostrados na Figura 122.

Pode ser visto que o *solver* padrão Gauss com pivotamento parcial desenvolve-se aproximadamente na ordem  $\mathcal{O}(N^3)$  com o número de graus de liberdade  $N$  do modelo. Para o último problema com 8192 graus de liberdade, o solver Gauss gastou um dia e nove horas para resolver o sistema de equações. Nota-se que o solver Gauss é ainda o solver padrão para a maioria dos códigos escritos para o método dos elementos de contorno, particularmente aqueles disponíveis em muitos livros-texto.

O tempo gasto *solver* GMRES padrão, usando matrizes densas, é basicamente composto pelo tempo de integrar as matrizes e aplicar as condições de contorno para formar a matriz do sistema, tarefa de ordem  $\mathcal{O}(N^2)$ . O número de iterações executadas permaneceu basicamente o mesmo para todos os testes.



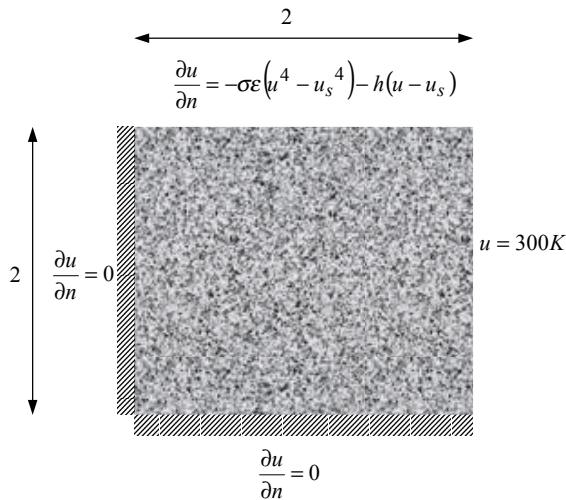
**Figura 122 Estudo de escalamento do método de compressão**

O tempo gasto pelo *solver* wavelet para comprimir as matrizes, na Figura 122, é composto pelo tempo de integrar estas matrizes, comprimi-las e resolver o sistema. O gráfico mostra que o algoritmo wavelet, como implementado, não irá aumentar a velocidade dos *solvers* iterativos se o sistema de equações for resolvido uma única vez porque o tempo de compressão é de ordem  $O(N^2 \log^2 N)$ , que torna-se a parcela dominante do tempo de solução. Entretanto a rotina de convolução utilizada neste trabalho para implementar a transformada wavelet não está otimizada de forma alguma, estando na forma acadêmica, e portanto o tempo de compressão pode ser ainda diminuído várias vezes pelo uso de algoritmos de convolução rápidos.

Entretanto, se o sistema de equações dever ser resolvido várias vezes, com diferentes casos de carregamento, então o tempo total de resolução dos vários casos com o solver wavelet será incrementado por somente uma pequena fração de tempo do tempo inicial de compressão. Comparando apenas o tempo gasto pelo solver GMRES em ambas as formulações, o solver padrão gastou 27.5 segundos para resolver o último modelo com 8192 equações enquanto o solver wavelet gastou apenas 0.47 segundos. Uma regressão linear em escala logarítmica mostra que as últimas 6 soluções produzidas pelo solver wavelet correlacionam-se como  $2.05 \times 10^{-5} N^{1.1}$  com o número de graus de liberdade do modelo.

### V.9.6 Modelo não-linear em grande escala

Como um teste final para o método de compressão por blocos e também para o solver de Newton-Raphson, o modelo da placa quadrada da Figura 99 foi escolhido, sendo este agora submetido a condições de contorno não-lineares. Este problema é descrito como um modelo de transferência de calor com condições de contorno de radiação e convecção combinadas e foi estudado em detalhes por AZEVEDO (1985).



**Figura 123 Problema misto padrão, placa quadrada com fluxos prescritos em três lados e potencial prescrito no lado direito**

As condições de contorno para este problema estão descritas na Figura 123. Os lados esquerdo e inferior estão isolados, isto é, o fluxo através destes lados é zero. O lado direito está submetido a uma temperatura constante igual a 300K e o lado superior está submetido a uma condição de contorno combinada de convecção, caracterizada pelo coeficiente usual de transferência de temperatura  $h$ , e radiação, caracterizada pela emissividade da superfície  $\epsilon$ .

A condição de contorno de radiação, adicionada à convecção, torna o problema não linear pois o fluxo dependente da radiação tem dependência da quarta ordem da temperatura.

A radiação consiste na transmissão de energia através de ondas eletromagnéticas, sem necessidade de meio material para sua propagação, para um outro corpo onde as ondas eletromagnéticas são geralmente convertidas novamente em calor. O fenômeno de radiação acontece com qualquer corpo cuja temperatura esteja acima do zero absoluto – ou zero Kelvin.

Não somente a qualidade mas também a quantidade de energia emitida por uma determinada superfície dependem da temperatura desta. Em 1879 Joseph Stefan determina que a quantidade de energia emitida por radiação é proporcional à quarta potência da temperatura do corpo emissor. Posteriormente seu aluno de doutorado, Ludwig Boltzmann formula a completa lei da transferência de energia por radiação, que toma a forma

$$q = -\sigma \varepsilon u^4 \quad (126)$$

onde  $\sigma$  é denominada constante de Stefan-Bolzmann, com valor  $5.67032 \times 10^{-8} \text{W/m}^2/\text{K}^4$ ,  $\varepsilon$  é denominada emissividade da superfície, adimensional, cujo valor pode variar de zero a um, e  $u$  é a temperatura da superfície emissora (em Kelvin).

Considerando-se a troca de radiação entre duas superfícies, uma à temperatura  $u$  (a emissora) e outra a uma temperatura de referência  $u_s$  (a receptora), esta lei toma a forma seguinte:

$$q = -\sigma \varepsilon (u^4 - u_s^4) \quad (127)$$

O modelo da Figura 123 foi então discretizado com 32768 elementos constantes por lado, resultando em um sistema com 131072 equações.

Sendo 2048 o número escolhido para o tamanho dos blocos, o que exige uma memória RAM de cerca de 100 megabytes para a montagem e compressão das matrizes, o sistema de equações ficou dividido em um grid de 64 por 64 blocos. O threshold utilizado para ambas as matrizes foi o mesmo,  $5 \times 10^{-5}$ .

**Tabela 9** Tempo exigido por tarefa para compressão do modelo de transferência de calor com 131072 equações

Tarefa	Por matriz	Por bloco	Total (est.)
<b>Integração</b>	2.50s	5.01s	5h 42min
<b>Transformada</b>	1.16s	2.31s	2h 37min
<b>Esparsificação</b>	0.12s	0.24s	0h 16min
<b>Total</b>	3.78s	7.56s	8h 36min

O tempo médio discriminado para o processamento de cada bloco num micro AMD K7 1.2GHz está mostrado na Tabela 9. O tempo combinado de CPU para montagem, compressão e armazenamento dos blocos em disco foi de 8 horas e 36 minutos mas como foi utilizado um conjunto de quatro microcomputadores Intel Pentium e AMD K7

com frequências variando de 650MHz a 1.2GHz, o tempo total “wallclock” ficou abaixo de 3 horas.

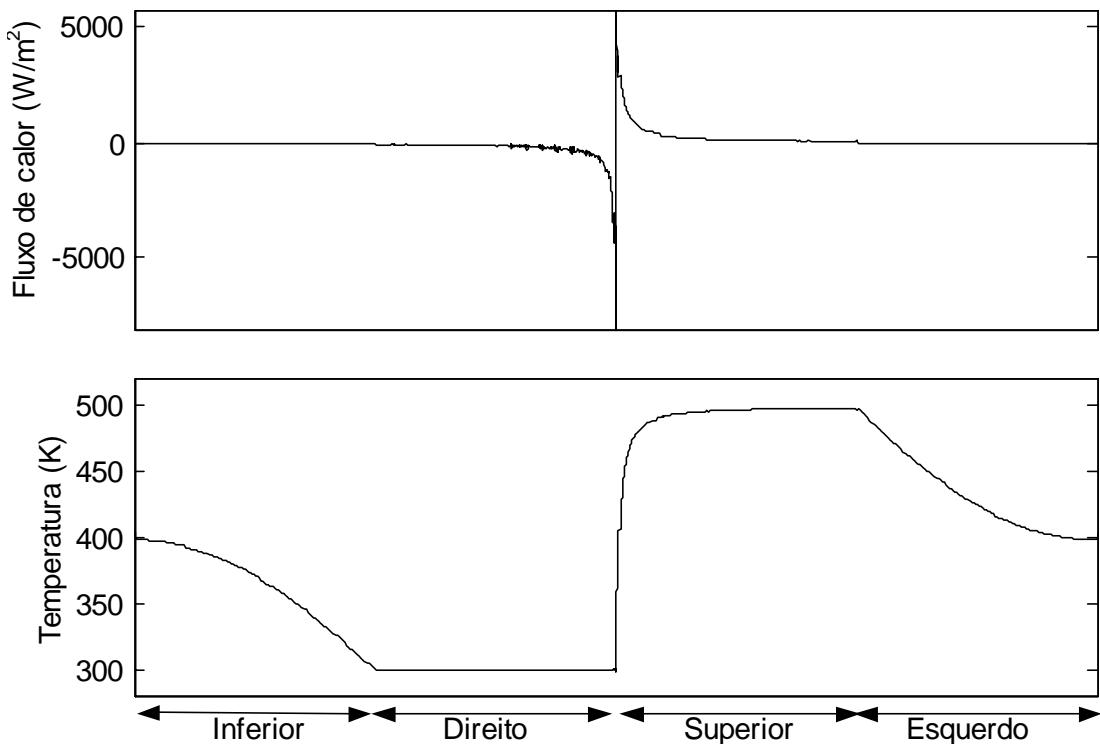
O espaço em memória exigida para armazenar as duas matrizes  $\mathbf{G}$  e  $\mathbf{H}$  para este problema seria de 256 gigabytes. Com a compressão, o espaço em memória necessário para armazenar todas as estruturas esparsas foi de 359 megabytes, resultando em uma compressão de 730 vezes.

Inicialmente o problema foi resolvido para o caso não linear com emissividade máxima, ou seja, 1. Neste caso a temperatura de referência para segunda superfície de troca de radiação é de 500K. Ao invés de somente emitir, é de se esperar, como a superfície da direita está submetida a uma temperatura constante de 300K, que a superfície superior receba mais calor por radiação que emita, ficando sua temperatura acima dos 300K e abaixo dos 500K. O coeficiente de filme da superfície superior é de  $10 \text{ W/m}^2/\text{K}$ .

A Figura 124 mostra a solução do problema onde são mostrados o fluxo de calor (acima) e a temperatura (abaixo). O eixo das abscissas, horizontal, representa o contorno do modelo linearizado como mostra a legenda abaixo da solução da temperatura. O eixo parte do canto inferior esquerdo e percorre o modelo em sentido anti-horário passando pelos lados inferior, direito, superior e esquerdo.

O entendimento do fenômeno físico simulado passa pela observação de que a superfície superior absorve energia de uma placa externa de temperatura superior à sua e a superfície direita absorve esta energia. As duas outras superfícies, inferior e esquerda, estão isoladas, isto é, possuem fluxo de calor zero. Estando a superfície superior exposta a uma outra superfície externa que para ela está emitindo radiação, é de se esperar que a temperatura tenda a aumentar quando afasta-se do lado direito, onde está prescrita a 300K, 200K abaixo da superfície emitindo radiação.

Por este motivo o canto superior direito é um ponto de transição entre uma superfície que absorve energia (a direita) e uma superfície que injeta energia no modelo (a superior). Isto causa uma forte descontinuidade na solução, o que é visto claramente pelo gráfico do fluxo de calor na Figura 124.



**Figura 124 Solução do problema de transferência de calor com condição de radiação com emissividade máxima para a superfície superior.**

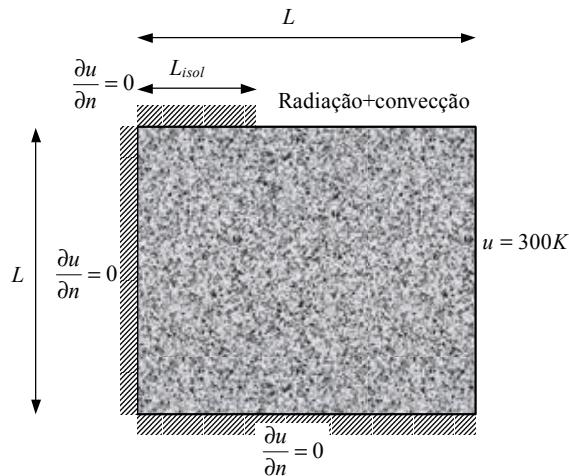
Esta descontinuidade leva à reflexão de que, mesmo em casos onde a geometria é bastante suave, como neste, a solução pode ser bastante mal comportada. Se nestes casos difíceis o modelo não estiver representado com um alto grau de refinamento é possível que algum aspecto importante do fenômeno físico sendo estudado seja mascarado, ou mesmo não seja captado, pela solução. É o caso análogo da elasticidade, onde o ponto de aplicação de uma carga concentrada não raras vezes exige uma atenção redobrada para que o fenômeno físico seja bem representado. Permanecendo-se a carga imóvel, é possível refinar-se a malha somente onde ela está aplicada. Caso contrário, se a carga puder ser aplicada em qualquer ponto do modelo então é necessário que a malha esteja refinada em todos os pontos, o que pode fazer explodir o número de graus de liberdade, e consequentemente o número de equações, do modelo, inviabilizando a sua solução numérica.

Esta é a grande vantagem da super-discretização possibilitada pela compressão das matrizes que se não for utilizada teria-se que recorrer a procedimentos adaptativos em problemas mais complexos. Tendo-se tal liberdade de aplicação de cargas no contorno a partir de um modelo super-refinado possibilita-se a exclusão de um grau a mais de ignorância do fenômeno físico que se quer estudar. São tantos os fatores que podem comprometer a solução significativa de um modelo numérico que ter-se a certeza que

uma falha na discretização da geometria não está comprometendo os resultados obtidos é um grande alívio.

Outra vantagem desta super-representação do modelo é que em casos de análise inversa, onde as condições de contorno mudam rapidamente por causa das soluções diversas tentadas pelo método de otimização, é necessário ter-se um modelo que seja robusto o suficiente para fornecer soluções precisas seja qual for o carregamento aplicado - a princípio, arbitrário. Nos casos de análise inversa onde deseja-se descobrir a melhor posição para a aplicação de um carregamento, como em proteção catódica, a precisão do algoritmo está diretamente ligada ao refinamento adotado.

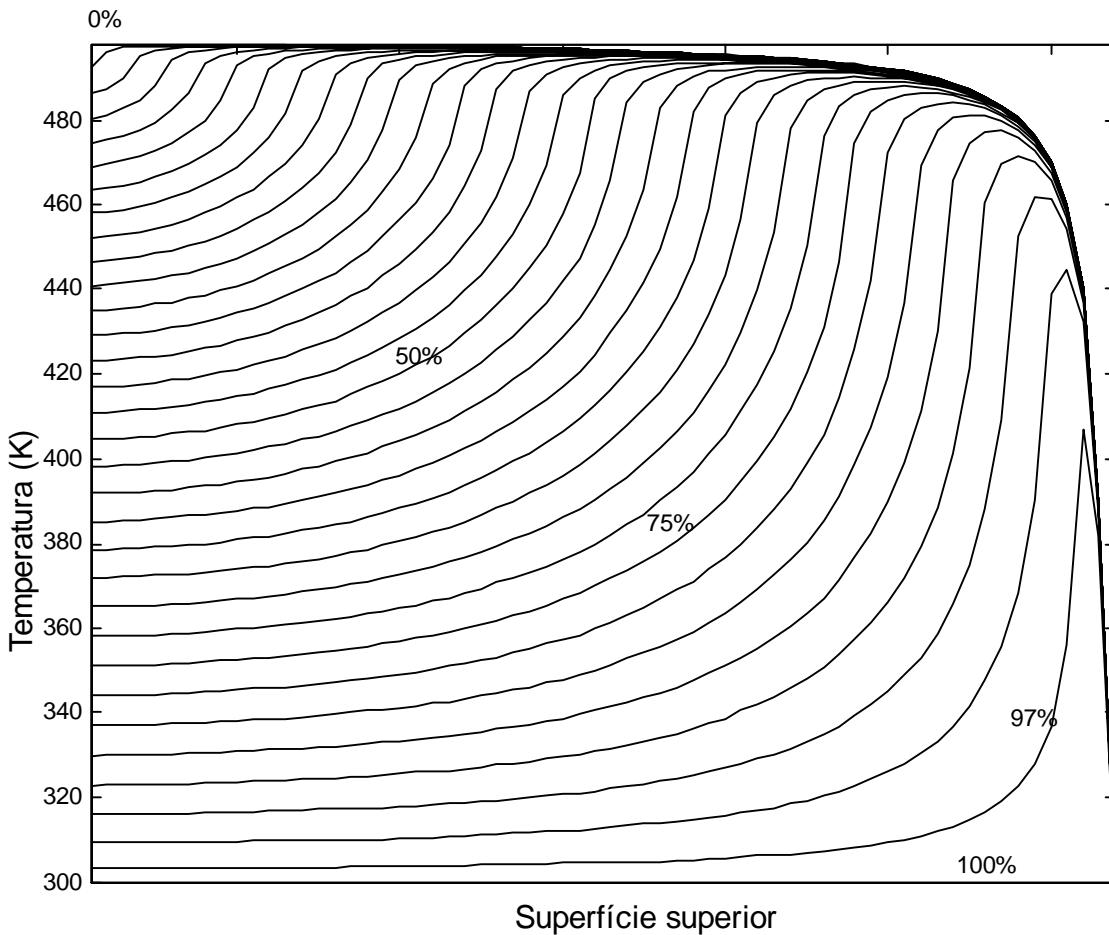
Para ilustrar melhor este ponto, o problema anterior foi submetido a uma série de carregamentos realmente diferentes uns dos outros: cada carregamento difere do outro por um sucessivo isolamento da superfície superior a partir do lado esquerdo, como se ilustra na Figura 125. Define-se o percentual de isolamento como a relação entre o comprimento isolado  $L_{isol}$  da superfície superior sobre seu comprimento total  $L$ . Na superfície superior restante, não isolada, permanecem as condições de convecção e radiação máxima, como anteriormente definido.



**Figura 125 Problema de transferência de calor com radiação e convecção mostrando o isolamento sucessivo da superfície superior**

Discretizou-se o modelo da Figura 125 utilizando-se dois modelos, um com 64 e outro com 2048 elementos por lado, números talvez mais que suficiente para representar os efeitos resultantes do isolamento sucessivo. Cada um dos dois modelos foi resolvido 32 vezes variando-se o percentual de isolamento, como definido anteriormente, de 0 a 100%. A solução da temperatura obtida é mostrada na Figura 126 para o modelo com 64 elementos por lado e na Figura 127 para o modelo discretizado com 2048 elementos

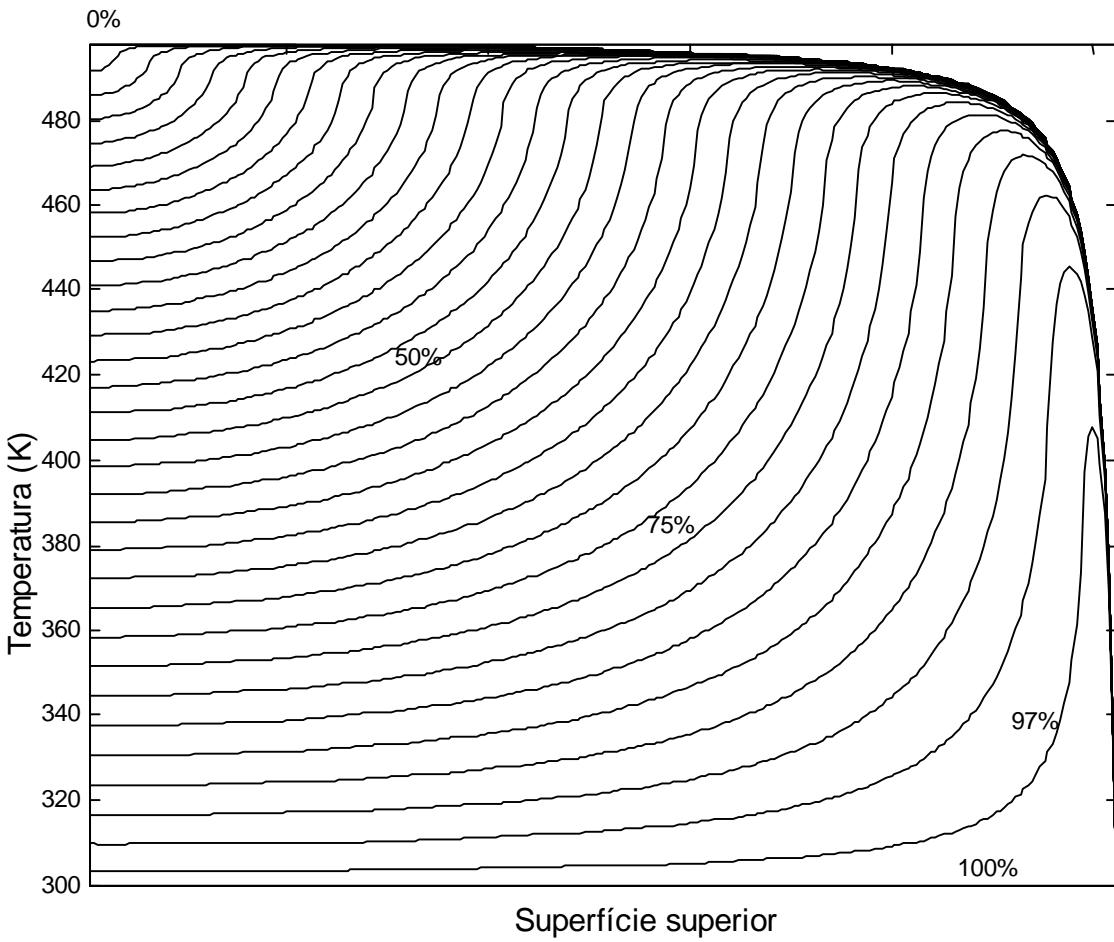
constantes por lado. As percentagens de isolamento estão mostradas ao lado das respectivas curvas.



**Figura 126 Problema de transferência de calor com sucessivo isolamento da superfície superior – temperaturas obtidas na superfície superior para o modelo discretizado com 64 elementos constantes por lado**

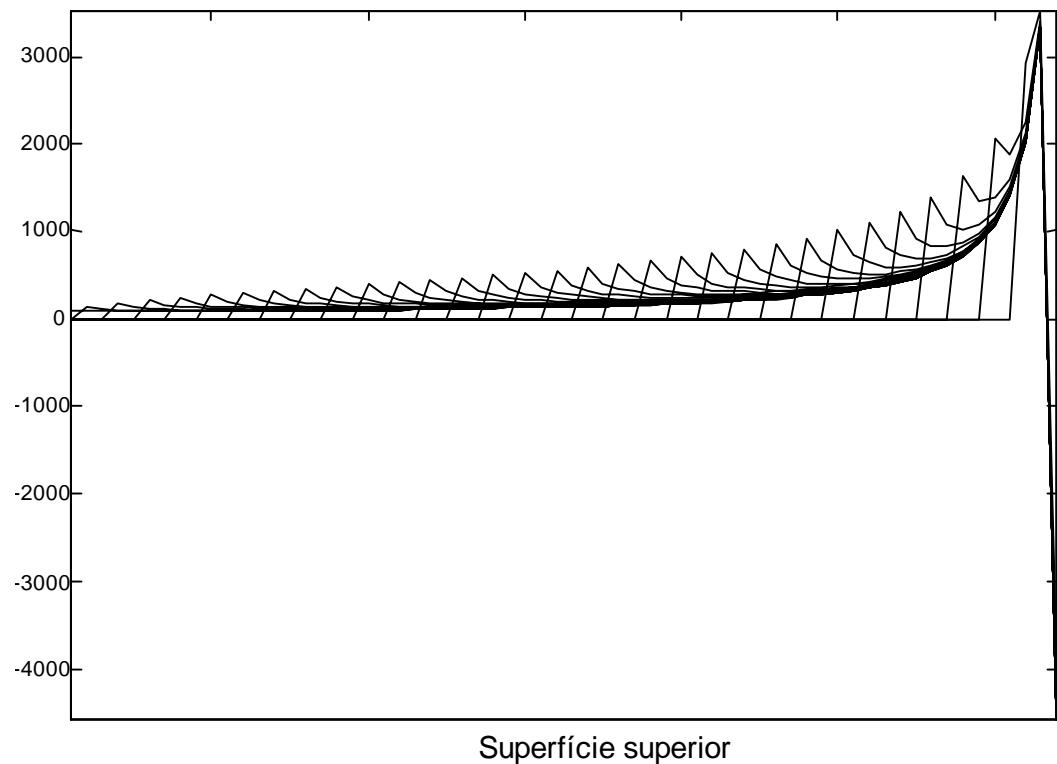
Percebe-se em ambas as figuras que o sucessivo isolamento a partir da esquerda tem o efeito de diminuir a temperatura que, mesmo assim, tende a aproximar-se dos 500K na extremidade direita da superfície superior. Mesmo com um isolamento de 97% a temperatura ainda alcança um pico de cerca de 407K um pouco antes da extremidade direita. No caso limite, 100% de isolamento, a temperatura no lado superior é constante e igual a 300K.

Ambos os modelos foram capazes de captar bem a curva da temperatura que mostrou-se bem comportada por toda a região. Quando o grau de isolamento tende a 100%, no entanto, a melhor discretização do segundo modelo mostra melhor a curva que a temperatura faz até alcançar um pico expressivo. Caso este pico seja importante por algum motivo, é necessário que a malha tenha o grau de discretização adequado.

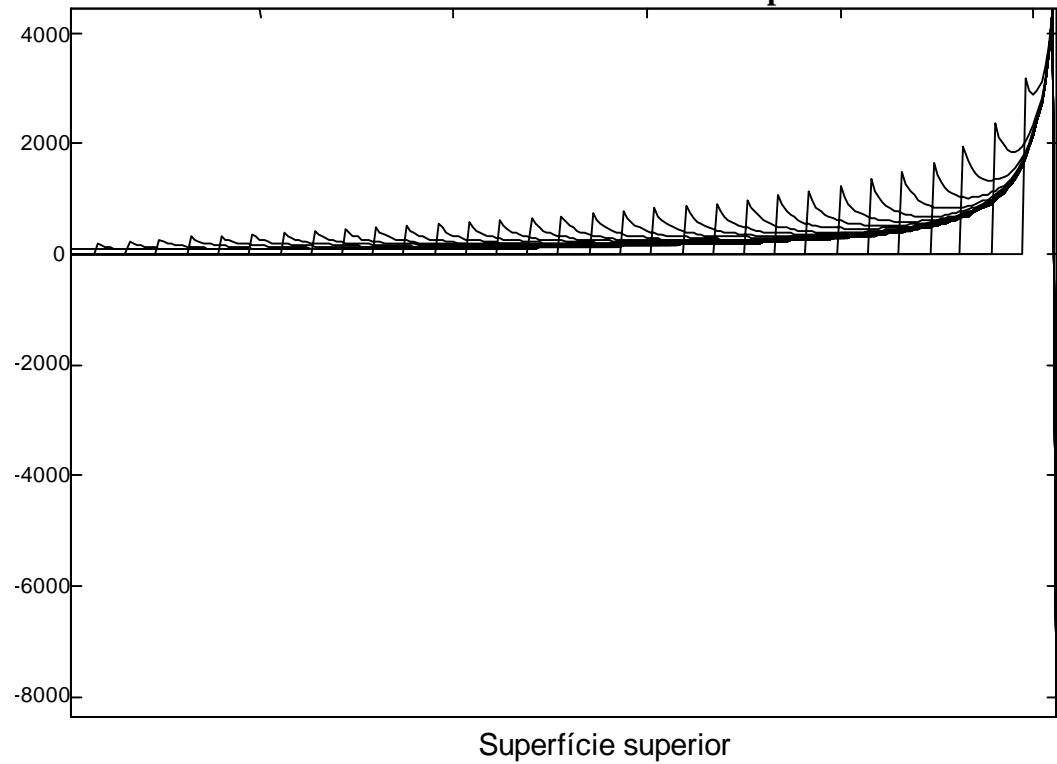


**Figura 127 Problema de transferência de calor com sucessivo isolamento da superfície superior – temperaturas obtidas na superfície superior para o modelo discretizado com 2048 elementos constantes por lado**

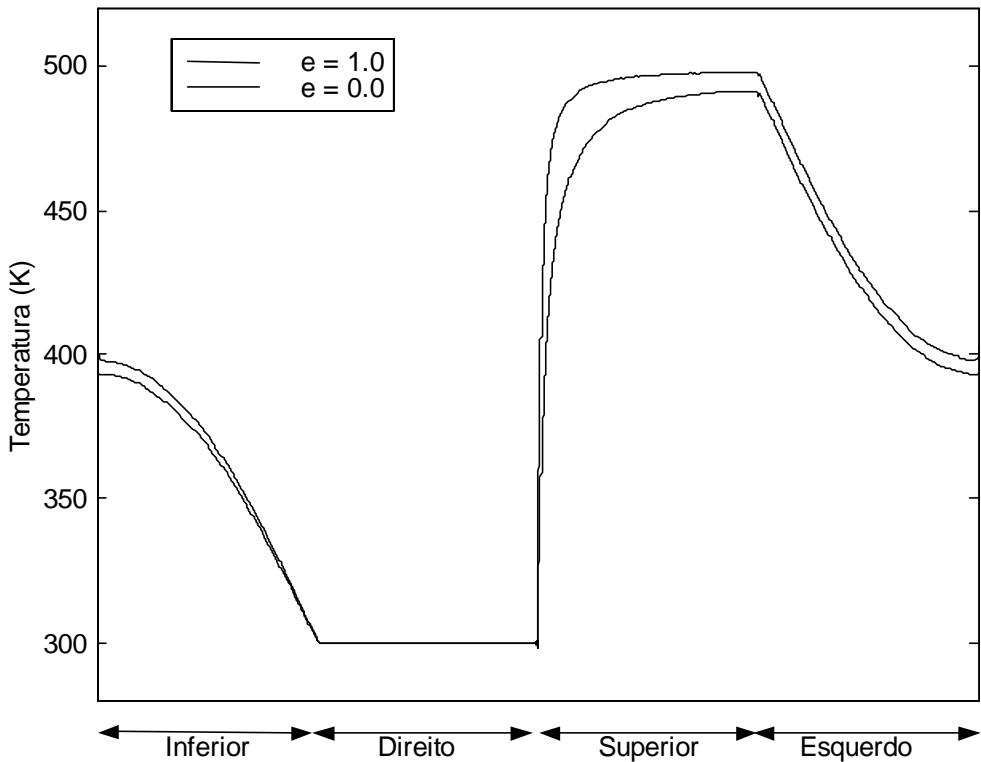
O mesmo comportamento suave não acontece com o fluxo de calor, como pode ser visto na Figura 128 para o modelo com 64 elementos e na Figura 129 para o modelo com 2048 elementos. Percebe-se nestas figuras que o fluxo de calor, como previsto, é sucessivamente anulado na medida que o isolamento avança da esquerda para a direita. As curvas todas têm a mesma tendência: iniciam no zero ao término do isolamento e então têm um salto para um valor significativo e então decrescem seu valor até encontrar uma curva assintótica que então leva o fluxo a um pico muito alto na extremidade direita. Este valor aparenta ser finito e de valor próximo a  $4000 \text{ W/m}^2/\text{K}$ . Observa-se que, neste caso, a observação do pico máximo dependeu bastante do grau de refinamento da malha. Para o modelo de 64 elementos as curvas apresentam-se bastante irregulares e não podem ser usadas para ter-se qualquer certeza a respeito do comportamento do fluxo perto da singularidade.



**Figura 128 Problema de transferência de calor com sucessivo isolamento da superfície superior – fluxo de calor obtido na superfície superior para o modelo discretizado com 64 elementos constantes por lado**



**Figura 129 Problema de transferência de calor com sucessivo isolamento da superfície superior – fluxo de calor obtido na superfície superior para o modelo discretizado com 2048 elementos constantes por lado**

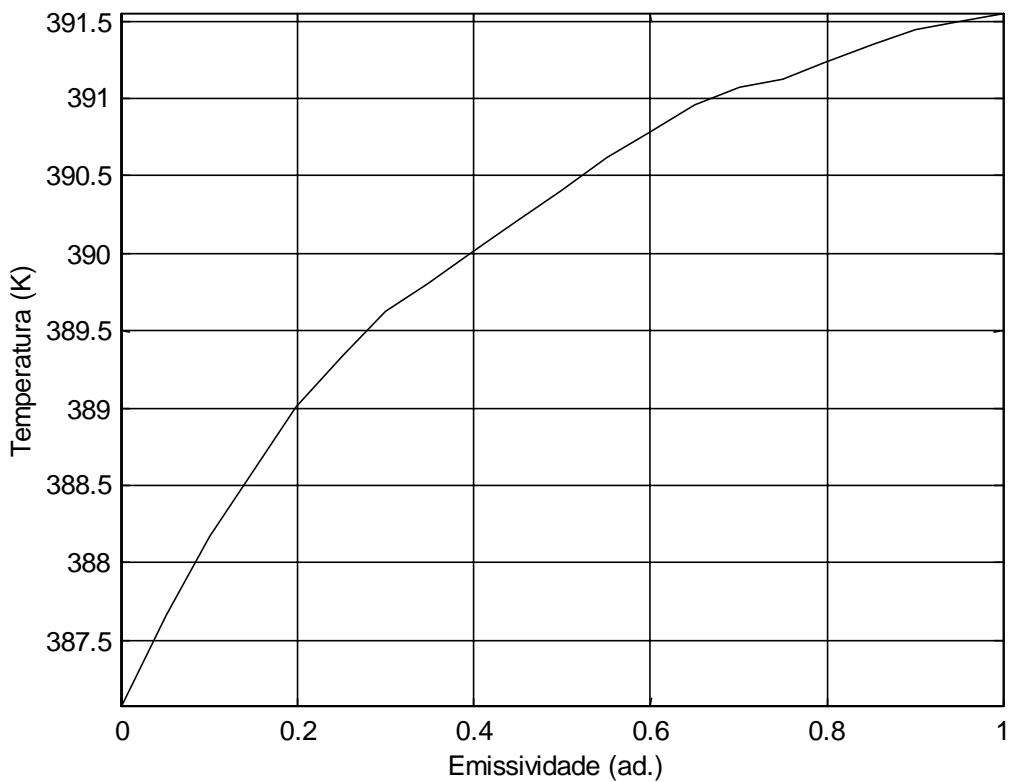


**Figura 130 Solução do problema de transferência de calor com radiação para dois valores extremos de emissividade: nulo e total.**

A compressão das matrizes como proposto, além de possibilitar a solução de tais problemas passíveis de serem resolvidos em tempo finito e aceitável com um método iterativo padrão mas no entanto impossíveis de serem resolvidos pela escassez de memória, possibilita o reaproveitamento das matrizes para outros casos de carga. A Figura 130, por exemplo, mostra a solução do mesmo problema para dois casos extremos de radiação: a emissão e recepção total, como resolvido anteriormente, que é um caso não-linear e a ausência de radiação, quando fica-se somente com a condição de convecção, o que torna o problema linear.

Compreende-se da Figura 130 que a influência da radiação é bastante significativa neste caso em particular. Para entender-se qual a real influência da emissividade sobre a temperatura em um determinado ponto, o ponto superior central, por exemplo, resolveu-se 20 vezes o modelo com 131072 elementos no total, cada caso de carga correspondendo a um valor da emissividade, que fez-se variar de zero a um. A Figura 131 mostra o resultado desta análise.

Percebe-se nesta figura que a introdução da radiação com o aumento da emissividade provoca uma rápida elevação da temperatura que abrange-se com o progressivo aumento de seu valor.



**Figura 131 Variação da temperatura no ponto superior central devido à variação da emissividade da radiação nesta superfície**

O objetivo de tantas análises com diferentes condições de contorno e diferentes tamanhos de modelo não teve a intenção de mostrar que um modelo de 2048 elementos mostra melhor os resultados que um modelo menor com 64 elementos mas, sim, enfatizar como a presente técnica facilita o processo de descoberta que envolve a interação do pesquisador com o fenômeno físico sendo estudado na medida que permite a manipulação e observação exaustiva do modelo sob diversas condições “ambientais”.

## VI CONCLUSÕES

No presente trabalho foram apresentadas as técnicas mais freqüentemente usadas da teoria das transformadas tempo-frequência, concatenando em uma análise única a teoria das wavelets e das distribuições tempo-frequência que usualmente aparecem em publicações distintas. A principal conclusão retirada do presente estudo é certamente a sobre a aplicabilidade das técnicas, isto é, foi descobrir-se onde utilizar uma ou outra técnica para que se obtenham bons resultados.

Foi visto que a transformada wavelet contínua, embora possuindo várias características matemáticas interessantes, não deve ser usada como uma distribuição tempo-frequência como vem sendo feito em vários trabalhos (HAGELBERG, GAMAGE, 1994, FOUFOULA-GEORGIOU, KUMAR, 1994, INOUE *et al*, 1996, WILLIAMS, AMARATUNGA, 1994). Neste caso, tanto o espectrograma como distribuições de melhor qualidade como a *aliasing free* SAF apresentada em tese de mestrado (BUCHER, 1998) a substituem com melhores resultados. O motivo para isto é que a transformada wavelet divide o espaço tempo-frequência em um grid não-uniforme, ficando este mais rarefeito nas altas freqüências e, assim, perdendo qualidade nestas regiões. Nas transformadas bilineares o grid é uniforme, possuindo resolução rigorosamente igual em qualquer faixa de freqüências. A única exceção a esta regra dá-se quando se faz necessária a transformada inversa que, em geral, as transformadas bilineares (espectrograma e Cohen) não possuem.

A aplicabilidade prática da transformada wavelet provém, sim, de sua transformada discreta (DWT) e principalmente da transformada rápida (FWT), sem a qual seu uso estaria restrito a algumas finalidades acadêmicas. A FWT possui velocidade monotonicamente superior à FFT em relação ao aumento do número de pontos do sinal, uma vez que o número de operações ponto-flutuante executados pela FWT tem ordem  $O(n)$  e a FFT apresenta  $O(n \log(n))$ . Esta superioridade, em termos de tempo de execução é bastante maior tendo em vista que o cálculo da FWT não envolve nenhum tipo de função transcendental ou complexa como senos, cossenos ou exponenciais, somente multiplicações e adições simples de números reais. Utilizando-se o esquema de *lifting* é possível executar a transformada wavelet exclusivamente dentro do espaço

dos números inteiros  $Z$  sem qualquer recorrência intermediária ao espaço dos números reais  $\mathfrak{R}$ .

Quanto às distribuições tempo-frequência, existem alguns comentários a serem feitos a respeito do espectrograma, da transformada de Wigner e, mais genericamente, de toda a classe de distribuições de Cohen.

O espectrograma possui qualidade bastante inferior a outras distribuições mais comuns mas é bastante estável, rápido e possui uma forma matemática bastante simples. Devido a isto, geralmente utiliza-se o espectrograma para fornecer uma prova particular a um teorema qualquer e então estende-se o resultado ao resto da família de transformadas tempo-frequência através das propriedades de conservação de energia.

A transformada de Wigner-Ville tem uso exclusivamente acadêmico pois é facilmente contaminada com interferência, também denominada de termos cruzados da transformada bilinear. Esta transformada, bem como qualquer outra distribuição tempo-frequência da classe geral de Cohen, apresenta o efeito de *aliasing* duplo quando não corretamente discretizadas. O *aliasing* duplo impede a análise de sinais que contenham frequências superiores a um quarto da taxa de digitalização.

A classe geral de Cohen consiste em uma moldura genérica para o cálculo das transformadas tempo-frequência, onde podem ser inseridos vários núcleos com características diversas, produzindo assim distribuições com propriedades diversas. A implementação computacional da transformada genérica da classe de Cohen, utilizando-se as discretizações óbvias, leva à inserção de *aliasing* duplo, bem como resulta num algoritmo excessivamente caro. A SAF, uma implementação delineada em trabalho anterior (BUCHER, 1998), fornece um algoritmo rápido e livre de *aliasing* duplo.

Da aplicação de estimativa da taxa de amortecimento pode-se concluir que o método desenvolvido possui as seguintes características:

- Possui baixa sensibilidade ao ruído adicionado, o que se traduz em estabilidade dos resultados. Esta estabilidade provém da utilização de métodos estatísticos (regressão linear) no cerne do método;
- Possui excelente precisão para determinação das taxas de amortecimento quando as taxas são lineares ou mesmo variáveis;

- O algoritmo de cálculo é rápido, simples e independente da introdução de parâmetros externos estimados *a priori* pelo operador;
- Produz um gráfico denominado *gráfico de amortecimento* que permite uma visualização singular do sinal, proporcionando maior entendimento do fenômeno físico estudado;
- O método consegue detectar com facilidade variações das taxas de amortecimento com o tempo quando, por exemplo, não-lineariedades são induzidas por variações estruturais do modelo;
- Utilizando-se técnicas de modulação simples, permite caracterizar a perda de energia em casos extremos como altas taxas de amortecimento e baixas frequências, onde somente poucos ciclos do sinal estão disponíveis para análise.

O uso da técnica de modulação, no entanto, não está exclusivamente comprometida com o método ora desenvolvido. A modulação pode ser pré-aplicada a qualquer sinal antes da sua submissão a um método qualquer. No entanto deve-se ter cuidado ao utilizá-lo pois este insere perturbações no início e ao final do sinal porque a transformada de Hilbert é calculada utilizando-se a FFT.

A principal conclusão que resulta da aplicação de compressão de sinais é que as técnicas tempo-frequência (wavelets) fornecem um contexto teoricamente bem mais adequado à compressão de dados do que a transformada de Fourier tradicional (DCT). Foi mostrado que mesmo utilizando-se um compressor extremamente simples baseado na transformada wavelet pode-se atingir taxas de compressão elevadas, tais como 21:1 para um sinal de apenas 4 pontos até 220:1 num sinal típico de tráfego de veículos com 15400 pontos. Além disso mostra-se que ao comprimir um sinal com wavelets pode-se, ao mesmo tempo, eliminar o ruído indesejável afetando minimamente o sinal.

Quando é qualitativamente comparada com a DCT, a transformada wavelet permite que o sinal restaurado contenha mais detalhes. As descontinuidades são mais bem resolvidas e os valores dos picos tendem a ser mais preservados.

Da aplicação nos sinais de tráfego conclui-se que a compressão wavelet pode ter muito bom rendimento como filtro de ruído. Além disso, como fator de economia apresenta-se como uma solução alternativa que, em um cenário conservador, pode reduzir as

necessidades computacionais (velocidade de transmissão e capacidade de armazenamento) em 50 vezes.

Da aplicação de compressão de matrizes provenientes do método dos elementos de contorno pela técnica das wavelets a primeira grande conclusão é que a expansão da solução do problema em série wavelet como realizado por LAGE e SCHWAB (1999) é uma alternativa muito cara e trabalhosa pois exige o desenvolvimento de todo um arsenal de rotinas de quadratura especiais para o cálculo das novas integrais – tarefa que está longe de ser trivial. De fato, as novas matrizes geradas por este método têm um comportamento totalmente diverso das matrizes geradas a partir do método dos elementos de contorno em sua forma tradicional, o que vem a acrescentar um grau a mais de incerteza na solução de problemas conhecidos, mesmo da teoria do potencial.

Além disso, vê-se que nos trabalhos publicados por estes autores não é informado explicitamente qual o tratamento dado às condições de contorno, tarefa que é tornada bastante mais complexa pois as equações representando as condições de contorno deverão estar em função não dos valores do potencial ou fluxo nos nós funcionais, mas sim dos coeficientes wavelet. A aplicação das condições de contorno constitui-se, portanto, no calcanhar de Aquiles da formulação citada.

Este trabalho, apercebendo-se de tal limitação, procurou uma alternativa híbrida que manteve as boas qualidades de ambos os métodos. Do ponto de vista da quantidade de memória utilizada, mostrou-se que o tamanho das matrizes do sistema pode ser bastante reduzido, isto é, podem ser muito comprimidas. Taxas de compressão acima de 5000/1 foram obtidas com erros aceitáveis sendo introduzidos na solução. A forma de montagem separada das matrizes – denominada de montagem virtual do sistema – possibilita que as matrizes uma vez montadas e comprimidas possam ser utilizadas para solucionar quaisquer casos de carga, incluindo-se casos não-lineares.

Este trabalho mostrou uma forma de montagem, compressão e utilização das matrizes em blocos, o que expande a possibilidade de solução de problemas com centenas de milhares de graus de liberdade. Mostrou-se que a paralelização da montagem e compressão destas matrizes neste esquema em blocos não é difícil e que este procedimento não exige intercomunicação entre os nós. Foi apresentada uma condição necessária e suficiente para a admissibilidade do particionamento.

Da escolha de parâmetros conclui-se que as propriedades de interpolação polinomial da família de Daubechies são suficientes para a compressão das matrizes com resultados satisfatórios. Observou-se que o tempo gasto pela transformada rápida (FWT) é diretamente proporcional ao tamanho do filtro wavelet, que por sua vez é proporcional à ordem da família. Na maioria dos casos a wavelet Daubechies com 10 coeficientes forneceu a melhor relação custo-benefício para os casos estudados. Tempos menores podem ser obtidos mas um filtro com no mínimo 6 coeficientes é altamente recomendado.

Foi visto que o tamanho dos blocos influí diretamente na taxa de compressão obtida mas não influí de forma alguma na taxa de erro, que depende basicamente dos limiares de compressão (*thresholds*) adotados. Mostrou-se que blocos de dimensão 2048 fornecem excelente taxa de compressão mas, se houver memória suficiente (512Mb) recomenda-se o uso de blocos de dimensão 4096 pela sua performance superior quando as taxas de compressão são altas. Além destas dimensões o ganho obtido não pareceu ser significativo.

Quanto ao formato das matrizes esparsas para armazenamento dos coeficientes wavelet, recomenda-se o uso do formato por coordenadas, com números ponto flutuante de 32 bits (*floats*). A adoção de números ponto flutuante de 64 bits (*doubles*) não mostrou significativo aumento no erro introduzido na solução. Se a linguagem adotada permitir a separação dos blocos em unidades distintas, recomenda-se a mudança dos índices de inteiros de 32 para 16 bits como fator de compressão adicional.

Mostrou-se que o formato binário de armazenamento é imprescindível para o armazenamento e reutilização eficiente das matrizes comprimidas. O formato texto, embora o único realmente universal, apresentou performance pífia nas velocidades de escrita e principalmente de leitura a partir do disco rígido. A sugestão de padrão de compatibilidade binário apresentado, um subconjunto simplificado do formato XDR, mostrou-se bastante útil para que diversas máquinas possam compartilhar e reutilizar os modelos comprimidos, o que permitiu as eficientes montagens em paralelo.

Foi mostrado que o esquema de montagem virtual possibilita a solução e implementação de *solvers* não-lineares através da simples substituição das fórmulas das condições de contorno por suas derivadas dentro do algoritmo de multiplicação matriz-vetor. Viu-se que diversos casos não-lineares de grande porte podem ser resolvidos em um tempo razoável.

Do estudo de velocidade conclui-se que o tempo de montagem e compressão das matrizes é o ponto chave da técnica apresentada. Dado que este tempo pode variar de 25% a 200% do tempo de integração das matrizes, vê-se que a técnica apresentada não pode concorrer com os tradicionais *solvers* iterativos para problemas pequenos – até algumas dezenas de milhares de graus de liberdade. Acima disto, no entanto, o fator memória disponível elimina completamente a possibilidade de utilizar-se os solvers tradicionais.

Conclui-se que a presente técnica mostra seu benefício nos seguintes casos: modelos grandes, com geometria suave, super-discretizados, com vários casos de carga ou não-lineares e condições de contorno complexas.

## **VI.1 Futuros desenvolvimentos**

Apesar do método de cálculo do amortecimento já estar em uma forma estável, ele ainda é um método de um grau de liberdade e analisa somente um sinal por vez. Os excelentes resultados obtidos neste trabalho indicam que se nestas condições desfavoráveis – suposição de um único grau de liberdade e análise de uma única resposta – o método já obtem excelentes resultados, a introdução de um modelo com vários graus de liberdade aliado a um método de otimização robusto como o *simulated annealing* deve propiciar a criação de um método ainda mais estável.

A compressão de sinais pode ser bastante otimizada com a utilização da teoria dos pacotes de wavelets (*wavelet packets*), que permite aumentar os índices de compressão, e também do esquema de *lifting*, que permite a criação de transformadores em tempo real, isto é, transformadores wavelet que podem ser aplicados a sinais com fluxo contínuo, virtualmente infinitos. Desta forma a compressão poderia ser realizada concomitantemente com a aquisição dos sinais, sem necessidade de recorte.

Na técnica de compressão de matrizes faltam testes com outros problemas físicos: elasticidade, acústica, eletromagnetismo, entre outros. Faltam testes com outros tipos de elementos que não o constante – lineares e quadráticos – para a teoria do potencial. A técnica da montagem virtual é teoricamente muito assemelhada à forma como está-se, hoje em dia, resolvendo problemas de elementos finitos e esta semelhança não é gratuita. Os pontos de contato entre a montagem virtual e as técnicas de montagem

elemento-por-elemento para elementos finitos fazem antever uma fácil e promissora comunhão entre métodos em um dialeto de alto nível, naturalmente paralelo.

Uma dúvida que não pôde ser esclarecida neste trabalho é como determinar-se *a priori* a relação entre os dois limiares de compressão que minimizem o erro introduzido em determinado problema. Falta investigar se existem soluções analíticas ou limites disponíveis e, em caso negativo, estabelecer metodologias para encontrá-los.

## VII APÊNDICE A

### PADRÃO DE COMPATIBILIDADE BINÁRIA

O formato XDR vem de encontro à necessidade de criar um formato de compatibilidade binária para dados de diversos tipos, sejam eles números ponto flutuantes, inteiros ou texto, em diversos tamanhos. A abordagem adotada neste trabalho foi adaptar o formato XDR às necessidades de portabilidade surgidas durante a pesquisa pois a teoria aqui desenvolvida permite a reutilização das matrizes pré-comprimidas para o cálculo de diversas condições de carga. O formato ora desenvolvido não é a implementação ao pé da letra do formato XDR. O primeiro motivo é que esta norma é extensa e pode ser bastante simplificada para a compatibilidade somente dos seguintes tipos de dados: byte (8 bits), double (64 bits), float (32 bits), inteiro curto (16 bits), inteiro (32 bits) e texto. A maior parte das aplicações é satisfeita com estes tipos de dados. Outros tipos de dados podem ser facilmente adaptados, como o inteiro longo que pode ser armazenado como dois inteiros simples.

A portabilidade foi testada para as seguintes plataformas: Intel PC/Windows, Intel PC/Linux-Gnu, Sun/Solaris, DEC Alpha/OSF. Os compiladores testados foram o GNU GCC v2.95.2 para os sistemas Unix e Microsoft Visual C++ v6 para IntelPC/Windows. A linguagem alvo foi o C++.

A compatibilidade dos números ponto flutuantes – double e float – é trivial pois a maioria dos fabricantes, com raras exceções como a Cray, adere à norma IEEE-754. O maior problema ocorre com os números inteiros, onde podem ocorrer dois tipos de variações principais: o número de bits utilizados e a ordem dos bits.

Em algumas máquinas, como a DEC Alpha, o tamanho dos inteiros e inteiros longos é de 64 bits. No entanto em várias outras plataformas ele inexiste o que resultou na exclusão deste tipo de dados. O inteiro adotado é de 32 bits, sendo portanto desprezados os bits superiores aos 32 bits menos significantes.

O problema principal de compatibilidade é, no entanto, a ordem dos bytes dentro do espaço de memória reservado para cada tipo de dados. Na maioria das plataformas é adotado o formato Little-Endian, isto é, os bytes menos significantes vêm antes (em relação à posição na memória) do que os bytes mais significantes. O número inteiro

(em hexadecimal) 0x99887766 seria armazenado sequencialmente na maioria das máquinas como os bytes 0x66, 0x77, 0x88 e por último 0x99. Nas plataformas Sun, no entanto, os números inteiros são armazenados na ordem Big-Endian, na ordem inversa de bytes, o que no exemplo anterior resultaria na sequência 0x99, 0x88, 0x77 e por último 0x66.

Este comportamento deve ser verificado no início da execução para que posteriores operações de escrita-leitura sejam corrigidas. O formato XDR, talvez por ter sido gerado dentro da Sun Microsystems, segue o formato Big-Endian. Neste ponto abandonamos o formato XDR em favor da performance visto que a maioria das máquinas disponíveis para este trabalho (máquinas Intel e DEC) não armazena os inteiros nesta forma.

A seguinte rotina em linguagem C++ deve ser chamada no início da execução de todos os programas a fim de checar a compatibilidade da máquina ao formato adotado.

```
// These flags are worldwide available
bool XDRCompatible = false;
bool ReverseBytes = false;
```

Estes flags são variáveis que são calculados pela rotina CheckMachine, abaixo, para verificar a compatibilidade da máquina ao formato. Evidentemente não se deseja que uma máquina seja incompatível mas não se deseja entrar nos detalhes de uma conversão entre formatos diferentes em ponto-flutuante, o que pode gerar erros de arredondamento insignificantes mas não desprezíveis. Se o resultado da rotina abaixo for negativo é preferível salvar as matrizes comprimidas em um formato texto com excesso de casas decimais, a favor da segurança.

```
// This routine checks if the machine can adhere to the proposed format
// If so, returns true. Otherwise, false.
bool CheckMachine( ){
```

Esta rotina retornará verdadeiro se esta máquina é compatível com o formato ora adotado e falso em caso contrário.

```
// Return now if already tested
static bool XDRTested = false;
if ( XDRTested ) return XDRCompatible;
// Declares a simple counter
int i;
```

Neste ponto checa-se o fato desta rotina já ter sido chamada anteriormente, para o caso de múltiplas chamadas e nenhum controle externo. A seguir declara-se a variável “i”, um contador que será usado extensivamente adiante.

```

// Check if the int is reversed
int test = 1;
unsigned char* p = (unsigned char*) &test;
if (*p==1)
    ReverseBytes = false; // Here is the PC/DEC Alpha!!
else
    ReverseBytes = true; // Here is the Sun!!

```

Este trecho de código testa se a máquina é Big-Endian (bytes mais significativos antes, como na Sun) ou Little-Endian (bytes menos significativos à frente como no Intel/PC ou DEC Alpha). A resposta é armazenada no flag externo ReverseBytes.

```

// Test integer sizes requirement
if (
    ( sizeof(long)!=8 && sizeof(long)!=4 )
    || sizeof(int)!=4
    || sizeof(short)!=2
    || sizeof(char)!=1
)
{
    std::cerr << "***** Integer sizes failed! " << std::endl;
    std::cerr << "Long : expected minimum 4 got " << sizeof(long) << std::endl;
    std::cerr << "Int : expected 4 got " << sizeof(int) << std::endl;
    std::cerr << "Short: expected 2 got " << sizeof(short) << std::endl;
    std::cerr << "Char : expected 1 got " << sizeof(char) << std::endl;
    XDRCompatible = false;
    return false;
}

```

O trecho de código acima testa a compatibilidade de tamanho de inteiros. Para os inteiros longos é facultado o tamanho de 8 ou 4 bytes. O inteiro simples deve ser 4 bytes, o inteiro curto, 2 bytes e o tipo char (byte) deve ter exatamente 1 byte de comprimento.

```

// Integer compatibility types
long tl = 0x04030201;
int ti = (int) tl;
short ts = (short) tl;
if (sizeof(long)==8)
{
    tl = long(0x08070605) << 32;
    tl += 0x04030201; // Longs with 64 bits (Alpha)
}

```

A seguir declaram-se algumas variáveis “tl”, “ti” e “ts” para testar a exatidão da representação destes tipos. Observe-se que, no caso do tipo inteiro longo ter o tamanho de 8 bytes, é necessário uma operação aritmética para inicializar a respectiva variável. Caso contrário, se fosse inicializada diretamente através da expressão tl=0x0807060504030201L, haveria um erro de compilação nos sistemas onde este tipo tem o tamanho de 4 bytes.

```

// Test long type
p = (unsigned char*) &tl;
if ( ReverseBytes ) p += (sizeof(long)-1);
for ( i=0; i<sizeof(long); i++ )
{
    unsigned char expected = i+1;
    unsigned char got = ReverseBytes ? *p-- : *p++;
    // Test if the byte is as expected
    if ( got != expected )
    {
        std::cerr << "Long Fatal: the compatibility test failed"
                << std::endl;
    }
}

```

```

        std::cerr << "Expected " << expected << " but got " << got
        << " in the " << i << "th byte" << std::endl;
        XDRCompatible = false;
        return false;
    }
}

```

O trecho de código acima testa o tipo inteiro longo byte-a-byte, isto é, dado um valor conhecido, é testado se os bytes contidos em memória estão na ordem esperada conforme previsto pelo flag ReverseBytes. Caso contrário, uma mensagem é mostrada na saída console padrão e a rotina retorna falso. Por simplicidade, foram omitidos os trechos de código que verificam os tipos inteiro e inteiro curto por serem completamente similares ao trecho de código acima listado.

```

// Test size compatibility of floating point numbers
if ( sizeof(float) != 4 || sizeof(double) != 8 )
{
    std::cerr << "Double (8 bytes) or Float(4 bytes) sizes, Fatal: "
    << "this machine is not IEEE-754 compliant!"
    << std::endl;
    XDRCompatible = false;
    return false;
}

```

Primeiro teste de compatibilidade em ponto flutuante: verifica-se o tamanho dos tipos float (32 bits) e double (64 bits) que devem possuir, respectivamente, 4 e 8 bytes de comprimento.

```

// Expected values (IEEE-754)
unsigned char fex[] = { 0xc3, 0xf5, 0x48, 0x40 };
unsigned char dex[] = { 0x1f, 0x85, 0xeb, 0x51, 0xb8, 0x1e, 0x09, 0x40 };
// Test floating-point compatibility
float f = (float) 3.14;
double d = (double) 3.14;

```

As linhas acima definem duas variáveis em ponto flutuante, “f” e “d”, ambas representando o valor 3.14 que deverão estar armazenadas em memória conforme os vetores “fex” e “dex”, os valores esperados.

```

// Test float 32-bits
p = (unsigned char*) &f;
for (i=0; i<4; i++, p++)
{
    unsigned char expected = fex[ ReverseBytes ? 3-i : i ];
    if ( *p != expected )
    {
        std::cerr << "Float fatal: this machine is "
        << "not IEEE-754 compliant!"
        << std::endl;
        std::cerr << "Expected " << (int) expected
        << " got " << (int) *p
        << " on " << i << "th element" << std::endl;
        XDRCompatible = false;
        return false;
    }
}

```

O código acima testa a variável “f”, um número ponto flutuante de 32 bits, contra os valores esperados, dados pelo vetor “fex”. A checagem para o ponto flutuante de 64 bits é análoga e foi omitida por simplicidade.

```

    // Everything appears to be ok
    XDRTested = true;
    XDRCompatible = true;
    return XDRCompatible;
}

```

Se todas as checagens estiverem corretas, o fluxo da rotina deverá chegar a este ponto, onde será retornado o valor lógico verdadeiro indicando a compatibilidade da máquina. Deve-se atentar para o fato de que esta rotina é de checagem rápida, isto é, não se presume ser uma exaustiva checagem de compatibilidade, que deverá ser cautelosamente checada a partir da documentação do equipamento, mas, sim, um teste para fornecer ao programa os dois sinalizadores globais XDRCompatible e ReverseBytes.

Deve-se atentar para o fato de que se o sinalizador XDRCompatible for falso então é impossível utilizar-se em outras máquinas os dados salvos em formato binário a partir desta. No entanto, se o sinalizador ReverseBytes indicar uma reversão então o programa deverá responsabilizar-se por inverter a ordem dos bytes durante a gravação e leitura dos tipos inteiros, o que consiste em uma tarefa trivial.

Tomando-se estas cautelas e utilizando-se a rotina descrita é possível construir-se um programa portável que rode em compatibilidade binária completa com os dados salvos em qualquer plataforma que passe o teste acima e implemente corretamente a norma IEEE-754.

# VIII APÊNDICE B

## INTERFACES BÁSICAS PARA O MÉTODO DOS ELEMENTOS DE CONTORNO

A abstração a nível de componentes para a implementação do solver é baseado em apenas três interfaces básicas: vetor, operador e solver. A seguir está listado o arquivo cabeçalho para estas três interfaces. Perceba-se que todas as declarações de funções terminam com “=0”, o que força as classes derivadas a implementarem estas funções.

```
*****  
// VECTOR  
// A generic, double, vector  
*****  
class Vector {  
public:  
    virtual double* GetData() const=0;  
    virtual int     GetSize() const=0;  
    virtual void    SetSize( int Size ) =0;  
    virtual void    SetData( double* ptr, int Size )=0;  
};
```

As classes que implementam a interface “Vector” apenas precisam oferecer aos clientes um meio de acesso ao seu conteúdo, realizado pelas funções GetData e GetSize e também um meio de modificar seu conteúdo, realizado pelas funções SetData e SetSize.

```
*****  
// OPERATOR  
// A generic, maybe nonlinear, operator  
*****  
class Operator {  
public:  
    // Info functions  
    virtual int Dimension() const =0;  
    virtual bool IsLinear() const =0;  
    virtual bool GetDiagonal( const Vector& CurrentSolution,  
        Vector& diagonal ) const=0;  
    // Algebra  
    virtual void Apply( const SimVector& AVector, SimVector& Result ) const=0;  
    virtual SimOperator* GetJacobian( const Vector& AtPoint ) const=0;  
};
```

Classes implementando a interface “Operator” necessitam saber mais: uma forma de saber a dimensão espacial do operador, obtido pela função Dimension, uma forma de saber se o operador é linear ou não, o que é obtido pela função IsLinear, e uma forma de fornecer ao pré-condicionador diagonal a diagonal necessária para sua inicialização, realizado pela função GetDiagonal. Além disso estas classes precisam fornecer uma forma de aplicar-se a um vetor arbitrário, operação realizada através da rotina Apply (e posteriormente expandida para o símbolo de multiplicação ‘\*’ através de uma expansão *inline*). Para resolver-se operadores não-lineares é necessário também que se implemente a função GetJacobian.

```

//*****
// SOLVER
// A generic solver.
//*****
class Solver {
public:
    virtual bool Solve( const Operator& Op,
                        const Vector& Force, Vector& Solution ) = 0;
};

```

As classes que implementam a interface “Solver” necessitam somente implementar a rotina Solve, que se responsabiliza por tentar resolver o operador passado como argumento.

Esta simples divisão de tarefas entre três classes possibilita uma enorme flexibilidade à programação como ilustrado pela simplicidade de programação do solver não linear implementando o método de Newton-Raphson. A seguinte rotina em C++ resolve o operador supostamente não-linear “Op” para o vetor de força “Force” retornando falso caso o número máximo de iterações sejam atingidos e/ou o cálculo da correção “dx” seja impossível – no caso de singularidade. Caso a solução seja atingida conforme a tolerância indicada, o vetor resultante é retornado na variável “Solution”.

```

// Solves the nonlinear operator "Op" for the force vector "Force" and returns
// the resulting vector in the variable "Result"
bool NRSolver::Solve( const Operator& Op, const Vector& Force, Vector& Solution ) {
    // Get the operator's number of rows
    unsigned int N = Op.Dimension();

    // The residual vector
    Vector r = Op*Solution - Force;
    // The solution correction
    Vector dx( N, 0.0 );
    // Get the target norm = Tolerance * Initial residual
    double TargetNorm = TOL * r.norm2();
    // Set completion flag initially to false
    bool bComplete = false;

    // Start NR-iterations
    for ( unsigned int nit=0; nit<MAXIT; nit++ ) {
        // Retrieve the operator's Jacobian at the actual solution point
        Operator* Jac = Op.GetJacobian( Solution );
        // Calculate the correction DX = inv(Jacobian)*Residual
        if ( !StaticSolver.Solve( Jac, r, dx ) ) return false;
        // Correct solution
        Solution -= dx;
        // Evaluate Residual = Ax-b
        r = Op*Solution - Force;
        // if objective was reached, get out of this loop
        if ( r.norm2()<TargetNorm ) { bComplete = true; break; }
    }
    return bComplete;
}

```

Pode-se observar no código da rotina de Newton-Raphson que o operador “Op” é do tipo “Operator”, o qual é definido como abstrato, isto é, não possui implementação (código), apenas a interface. Em outro ponto do código as classes “Gemini”, a qual representa a matriz do sistema através do método da montagem virtual mostrado anteriormente, e a classe “DenseMatrix”, a qual representa uma matriz simples e densa, irão implementar o respectivo operador multiplicação. Desta forma ambas as matrizes

original e comprimida poderão utilizar-se do mesmo solver não linear sem precisar-se escrever uma rotina de Newton-Raphson para cada tipo de matriz. Evidentemente a matriz densa encarrega-se de recalcular-se utilizando o carregamento atual a cada vez que “GetJacobian” é chamado pois uma matriz densa é, por definição, linear.

### VIII.1.1 O montador universal para o método dos elementos de contorno

Uma grande flexibilidade pode ser obtida com a definição de interfaces para regiões (subdomínios) e elementos, o que possibilita a construção de um montador universal para qualquer domínio e qualquer elemento para o método dos elementos de contorno.

A interface para o elemento de contorno, definida pelo código C++ a seguir, é uma sugestão desenvolvida no presente trabalho de forma a orientar futuros trabalhos que pretendam usufruir desta vantagem. Pode-se dizer que, entre a definição de uma interface consistente como a apresentada a seguir e a sua implementação em código fonte, a primeira exigiu mais esforço do que a última.

```
class BEMElement {
public:
    // 1. Problem definition
    virtual unsigned int nDimensions() const = 0;
    virtual unsigned int nDOFPerNode() const = 0;
    virtual bool isHyperSingularFormulationAvailable() const = 0;
```

Em primeiro lugar, objetos desejando comunicar-se com o montador universal, em outras palavras, que implementem a interface BEMElement, devem responder qual é o domínio para o qual foram preparados, isto é, 1, 2 ou 3 dimensões, o que é feito pela função nDimensions. Nada impede, entretanto, que o montador universal opere com domínios de 4 ou mais dimensões. Outra característica do domínio que o elemento deve responder é o número de graus de liberdade por nó, não contando sua representação dual, respondida pela função nDOFPerNode. Um problema de potencial, bi ou tridimensional, deverá responder 1 a esta pergunta. Um elemento de elasticidade tridimensional deverá responder 3. Neste último caso tensões e deslocamentos não se contam separadamente pois um é a representação dual do outro. Outra pergunta que o elemento deverá responder é se este elemento suporta a integração hipersingular, respondida pela função isHyperSingularFormulationAvailable, que retorna um valor verdadeiro ou falso.

```
// 2. Basic questions about this element
virtual unsigned int nFunctionalNodes() const = 0;
virtual unsigned int nGeometricNodes() const = 0;
```

O grupo seguinte de funções responde ao montador as características deste elemento em particular. A função nFunctionalNodes responde qual o número de nós funcionais o

elemento possui, isto é, o número de nós que possuem graus de liberdade atachados. O número total de nós do elemento é o resultado da multiplicação de `nFunctionalNodes` pelo resultado da função `nDOFPerNode`. A função `nGeometricNodes` retorna o número de nós “geométricos” que este elemento possui. Um nó geométrico é um nó que existe apenas para definir a geometria do elemento, não possuindo qualquer grau de liberdade anexado, o que não impede que existam nós geométricos coincidentes com nós funcionais no elemento.

```
// 3. Node-related functions
virtual void GetFunctionalNodePosition(BEMDomain* Domain,
                                         unsigned int Index, double* Coordinates) const =0;
virtual void GetFunctionalNodeNormal(BEMDomain* Domain,
                                         unsigned int Index, double* Normal) const =0;
```

O grupo de duas funções acima responde questões acerca de cada nó funcional do elemento. `GetFunctionalNodePosition` retorna a posição de determinado nó e `GetFunctionalNodeNormal` retorna a normal do elemento neste determinado nó. Perceba-se que além do índice do nó, que deve seguir uma numeração interna, passa-se também uma referência ao domínio ao qual este elemento pertence. Isto acontece por uma razão simples: se cada elemento fosse guardar em seu conjunto de dados as coordenadas de cada nó ao qual está conectado, a quantidade de memória gasta para armazenar um número não desprezível de elementos poderia ser surpreendentemente grande. Desta forma, os elementos somente precisam armazenar o número dos nós ao qual estão conectados e, quando necessário, perguntar ao domínio qual a posição geométrica do nó dado seu número. Um elemento bidimensional retilíneo com um nó funcional no meio, por exemplo, pergunta ao domínio a posição dos seus nós extremos e depois retorna a média entre as duas coordenadas retornadas.

```
// 4. Cache-related functions
virtual unsigned long nSizeOfTemporarySpace() const { return 0; };
virtual void InitTemporarySpace( char* temp, BEMDomain* pDomain ) {};
};
```

Este grupo de duas funções acima prepara o elemento para o grupo de funções que virá a seguir, relacionadas à integração do elemento. É de longo sabido que a integração das matrizes de elemento de contorno deve ser realizada por colunas, o que é equivalente a fixar um determinado grau de liberdade de um determinado elemento e integrar-se sobre todos os pontos fonte do domínio. Estas funções fazem justamente preparar o elemento para esta maratona de integrações de forma que estas sejam realizadas o mais rápido possível. A função `nSizeOfTemporarySpace` retorna para o montador o tamanho do espaço de memória que o elemento irá precisar para armazenar as variáveis adicionais

calculadas para agilizar os cálculos. O montador irá reservar uma área de memória do tamanho especificado e chamar a rotina InitTemporarySpace para que o elemento tenha uma chance de inicializar este espaço com as variáveis calculadas antes que a rotina de integração principal comece a chamar a rotina de integração do elemento. Um elemento potencial poderia, por exemplo, aproveitar a execução da rotina InitTemporarySpace para calcular e armazenar a área do elemento, os cossenos diretores, normais e todas as variáveis que se fizerem necessárias. Durante a integração dos pontos fonte uma referência a esta área de armazenamento temporário será passada ao elemento para que este possa usá-la.

```
// 5. Element assembly
virtual bool IsRigidBodyMotionRequiredForG() const =0;
virtual bool IsRigidBodyMotionRequiredForH() const =0;
virtual void Integrate( char* tempSpace, unsigned int uDofType,
    double* pSourceCoordinates, double* G, double* H, bool IsInternal ) const =0;
virtual void IntegrateHS( char* tempSpace, unsigned int uDofType,
    double* pSourceCoordinates, double* pSourceNormals,
    double* G, double* H, bool IsInternal ) const = 0;
```

O grupo de funções acima relaciona-se à integração do elemento. IsRigidBodyMotionRequiredForG e IsRigidBodyMotionRequiredForH retornam verdadeiro ou falso caso o movimento de corpo rígido seja necessário para calcular as integrações singulares dentro do elemento. A função Integrate retorna dois vetores, um para cada matriz G e H, contendo as integrais calculadas para cada grau de liberdade do elemento. No caso da variável IsInternal for verdadeira, as coordenadas do ponto passadas referem-se a um ponto interno e, portanto, uma integração singular deverá ser realizada. Neste caso a variável uDofType contém o número do nó funcional sobre o qual a integração está sendo realizada. Caso a variável IsInternal seja falsa, uma integração externa poderá ser executada. Neste caso a variável uDofType contém o tipo de grau de liberdade sendo integrado. No caso de elasticidade bidimensional, zero referiria-se ao grau de liberdade na direção X e o valor um referiria-se à direção Y. A integração deve ser realizada em relação ao ponto fonte passado pela variável pSourceCoordinates.

A rotina de integração hipersingular IntegrateHS é similar a rotina de integração normal exceto que um parâmetro adicional pSourceNormals, contendo as normais do ponto fonte é passado à rotina. Esta é a única rotina de elemento que não necessita ser implementada desde que a função IsHyperSingularFormulationAvailable retorne falso.

Deve-se perceber que o primeiro parâmetro passado às rotinas de integração, tanto normal como hipersingular, é exatamente uma referência ao espaço de armazenamento

temporário inicializado anteriormente. A rotina de montagem irá encarregar-se de liberar este espaço de memória ao final da integração de todos os pontos fonte.

A rotina de montagem é um pouco mais envolvente, pois existe a necessidade de contemplar a montagem em blocos. A listagem a seguir, em pseudo-código, representa o montador universal de acordo como foi implementado. Ao invés de apresentar um pseudo-código enxuto e após comentar-se o que foi feito, preferiu-se fazer os comentários diretamente dentro do pseudo-código, aumentando um pouco mais seu tamanho mas acreditando-se que esta abordagem aumente a clareza desta operação fundamental para o funcionamento do esquema de montagem em blocos.

#### ***ROTTINA MONTABLOCO***

##### ***Entrada:***

Inteiro **Nbloco**: o tamanho do bloco a ser integrado  
 Inteiros **IP** e **JP**: a posição do bloco dentro da matriz original  
 Inteiro **Ndim**: o número de dimensões do domínio (2D, 3D,...)

##### ***Saída:***

matrizes **G** e **H**: as matrizes onde vão ser guardados os resultados da integração, o bloco em si.  
 vetor **F**: o vetor de força

##### ***Variáveis locais:***

Vetor booleano **DOFCalculado(Nbloco)**: dirá se um grau de liberdade já foi calculado ou não  
 matriz double **Coordenadas(Nbloco,Ndim)**: para conter as coordenadas dos pontos-fonte  
 matriz double **Normais(Nbloco,Ndim)**: para conter as normais dos pontos-fonte (exigido pela formulação hipersingular)  
 matriz double **ElemNúmero(Nbloco)**: contém a numeração dos elementos a serem integrados no bloco  
 Inteiro **dof**: grau de liberdade sendo integrado  
 Ponteiro à interface **BEMElement pElemento**: elemento sendo integrado  
 Vetores **Ge** e **He**: conterão o resultado da integração do elemento.  
 Inicialmente têm comprimento zero mas deverão ser dimensionados posteriormente. Alternativamente podem ser dimensionados com o valor do máximo número de graus de liberdade possível contido em um único elemento.  
 Vetor genérico **Temp**: conterá as variáveis locais requisitadas pelo elemento para sua integração  
 Vetor inteiro **DofIndex**: conterá a numeração (local ao elemento) dos Graus de liberdade de determinado elemento que pertençam ao bloco  
 Vetor inteiro **BkIndex**: posição, referente ao bloco, dos graus de liberdade armazenados em **DofIndex**  
 Inteiro **nIndices**: contém o número de posições em **DofIndex** e **BkIndex**

##### ***Início:***

Preencha as matrizes **G** e **H** inicialmente com zeros  
 Preencha o vetor **DOFCalculado** inicialmente com falso  
 Preencha as matrizes **Coordenadas** e **Normais** com as posições e normais de todos os elementos contidos no bloco.  
 Loop **dof=1..Nbloco**, através dos graus de liberdade (elementos)  
 Cheque se este grau de liberdade já foi calculado (veja o vetor **DOFCalculado**). Se foi, retorne ao início do loop para o próximo grau de liberdade  
 Encontre o elemento correspondente ao grau de liberdade na posição da coluna **JP+dof** e armazene o ponteiro na variável **pElemento**  
 Dimensione os vetores **Ge** e **He** para conter o número de posições retornada por **pElemento->nDOFTotal()**  
 Dimensione o vetor **Temp** para conter, em bytes, o espaço retornado pela função do elemento **pElemento->nSizeOfTemporarySpace()**  
 Inicialize o espaço temporário recém alocado chamando a rotina **pElemento->InitTemporarySpace(temp,this)**;  
 Preencha o vetor **DofIndex** com o número do grau de liberdade (numeração local ao elemento) de todos os graus de liberdade que estejam contidos neste bloco  
 Preencha o vetor **BkIndex** com a posição, no bloco, correspondente aos graus de liberdade contidos no vetor **DofIndex**. O inteiro **nIndices** conterá o número de posições nestes vetores  
 Loop **sp = 1** até **Nbloco** (os pontos fonte)

```

Se a formulação é hipersingular chame a rotina
pElemento->IntegrateHS() passando as coordenadas
(vetor Coordenadas) e normais (vetor Normais)
calculadas anteriormente. O resultado da
integrações são retornados nos vetores Ge e He.
Se a formulação for normal, chame a rotina
pElemento->Integrate()
Loop k = 1 até nIndices, faça a transferência
    H(sp,BkIndex(k))=He(DofIndex(k))
    G(sp,BkIndex(k))=Ge(DofIndex(k))
Fim loop
Fim loop dos pontos-fonte
Loop k = 1 até nIndices, faça
    DofCalculado(BkIndex(k))=verdadeiro
Fim loop
Fim loop dos graus de liberdade
Fim da rotina

```

Deve-se observar a função exercida pelo vetor **DofCalculado** que não permite que um determinado elemento seja calculado duas vezes, isto é, uma vez que o elemento integre um ponto fonte, todos os seus graus de liberdade não precisarão ser recalculados.

Esta rotina, como está mostrada, é geral do ponto de vista que qualquer elemento poderá ser implementado desde que este elemento implemente a interface BEMElement, mostrada e comentada anteriormente.

## IX REFERÊNCIAS BIBLIOGRÁFICAS

- ALLEMANG, R.J., BROWN, D., *Experimental modal analysis and dynamic component synthesis*, v. III, University of Cincinnati, Ohio, 1987.
- AOKI, S., KISHIMOTO, K., SAKATA, M., “Boundary element analysis of galvanic corrosion”, in: *Boundary Elements VII*, v. 1, pp. 63-71, Springer-Verlag, Berlin, 1985.
- ARRUDA, J.R.F., BERTOLINI, P.U., *Métodos para estimação de parâmetros modais*, UNICAMP/Campinas, SP, 1994.
- ATLAS, L. E., LOUGHLIN, P. J., PITTON, J. W., “Signal analysis with cone kernel time-frequency representations and their application to speech”, In: Boashash, B., *Time-Frequency Signal Analysis*, pp. 375-388, Longman Cheshire, 1992.
- AZEVEDO, J.P.S., *Análise de Problemas Não-lineares de Transferência de Calor pelo Método dos Elementos de Contorno*, M.Sc., COPPE/Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1985.
- BARRA, L.P.S., COUTINHO, A.L.G.A., MANSUR, W.J., TELLES, J.C.F., “Iterative solution of BEM equations by GMRES”, *Computers and Structures*, n. 111, pp. 335-355, 1994.
- BARRET, R., *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 1a. ed., Philadelphia, SIAM Press, 1994.
- BARSCH, T., KUNOTH, A., URBAN, K., “Towards object-oriented software tools for numerical multiscale methods for PDEs using wavelets”, In: *Multiscale Wavelet Methods for Partial Differential Equations*, v. 6, *Wavelet Analysis and its Applications*, Academic Press, pp. 383-412, 1997.
- BEDROSIAN, E., “The analytic signal representation of modulated waveforms”, *Proc. IRE*, v. 50, pp. 2071-2076, 1962.
- BEDROSIAN, E., “A product theorem for Hilbert transforms”, *Proc. IEEE*, v. 51, pp. 868-869, 1963.
- BEYLKIN, G., COIFMAN, R., ROKHLIN, V., “Fast wavelet transforms and numerical algorithms”, *I. Comm. Pure Appl. Math.*, v. 2, n. 44, pp. 141-183, 1991.

BEYLKIN, G., “Wavelets and fast numerical algorithms”, In: I.Daubechies, ed., *Different perspectives on wavelets (San Antonio, TX, 1993)*, v. 47, *Proc. Sympos. Appl. Math.*, pp. 89-117, Amer. Math. Soc., Providence, RI, 1993.

BOASHASH, B., “Estimating and interpreting the instantaneous frequency of a signal”, *Proc IEEE*, v. 80, pp. 520-538, 1992.

BOASHASH, B., O'SHEA, P., “Polynomial Wigner-Ville Distributions and Their Relationship to Time-Varying Higher Order Spectra”, *IEEE Trans. on Signal Processing*, v. 42, n. 1, 1994.

BOND, D.M., VAVASIS, S.A., “Fast wavelet transforms for matrices arising from boundary element methods”, Technical Report, Center for Applied Mathematics, v. 174, Eng. and Theory Center, Cornell University, New York, 1994.

BOVA, S.W., CAREY, G.F., “A distributed memory parallel element-by-element scheme for semiconductor device simulation”, *Comput. Methods Appl. Mech. Engrg.*, n. 181, pp. 403-423, 2000.

BRADLEY, J.N., BRISLAWN, C.M., HOPPER, T., “The FBI wavelet/scalar quantization standard for grayscale fingerprint image compression”, *Proc. of SPIE Visual Info. Proc. II*, pp. 293-304, Orlando, Florida, 1992.

BRAY, T., PAOLI, J., SPERBERG-MCQUEEN, C., “Extensible Markup Language (XML) 1.0”, W3C Recommendation, February 1998. Disponível para download em <http://www.w3.org/TR/1998/REC-xml-19980210>

BREBBIA, C.A., TELLES, J.C.F., WROBEL, L.C., *Boundary Element Techniques*, Berlin, Springer-Verlag, 1984.

BRUN, G., *Modulations Analogiques*, Paris, Eyrolles, 1991.

BUCHER, H.F., *Estudo de técnicas tempo-frequência e suas aplicações em engenharia civil*, M.Sc., COPPE/UFRJ, Rio de Janeiro, 1998.

CAREY, G.F., *Computational Grids: generation, adaptation and solution strategies*, 1<sup>a</sup> ed., Austin, Taylor&Francis, 1997.

CARPENTER, L.A., MUMFORD, A .M., *File Formats for Computer Graphics: Unravelling the Confusion*, In: AGOCG Technical Report Series, v. 10, 1992.

- CETIN, C.E., GEREK, O.N., ULUKUS, S., "Block wavelet transforms for image coding", *IEEE Trans. Circ. and Syst. for Video Tech.*, n. 3, pp. 433-435, 1993.
- CHOI, H. I., WILLIAMS, W. J., "Improved time-frequency representation of multicomponent signals using exponential kernels", *IEEE Trans. On Acoust., Speech, Signal Processing*, v. 37, pp. 862-871, 1989.
- CHRISTOPOULOS, C.A., EBRAHIMI, T., SKODRAS, A.N., "JPEG 2000: The new still Picture Compression Standard", In: anais do congresso *ACM Multimedia 2000*, Electronic proceedings, Los Angeles, October, 2000. Disponível para download no em <http://www.acm.org/sigs/sigmm/MM2000/ep/christopoulos/>
- CHUI, C. K., *An Introduction to Wavelets*, Academic Press, New York, 1992.
- COHEN, L., "Generalized phase-space distribution functions", *Jour. Math. Phys.*, v. 7, pp. 781-786, 1966.
- COHEN, L., "Quantization Problem and Variational Principle in the Phase Space Formulation of Quantum Mechanics", *Jour. Math. Phys.*, v. 17, pp. 1863-1866, 1976.
- COHEN, L., *Time-frequency Analysis*, 1 ed., New Jersey, Prentice Hall, 1995.
- COHEN, L., "A general Approach for Obtaining Joint Representations in Signal Analysis - Part I: Characteristic Function Operator Method", *IEEE Trans. on Signal Processing*, v. 44, n. 5, 1996.
- COOLEY, J.W., TUKEY, J.W., "An algorithm for the Machine Computation of Complex Fourier Series", *Mathematics of Computation*, v. 19, pp. 297-301, April, 1965.
- CUNNINGHAM, G. S., WILLIAMS, J. W., "High-Resolution Signal Syntesis for Time-Frequency Distributions", *Proc. IEEE ICASSP-93*, v. 4, pp. 400-403, 1993.
- CUNNINGHAM, G. S., WILLIAMS, J. W., "Vector-Valued Time-Frequency Representations", *IEEE Trans. on Signal Processing*, n. 7, v. 44, pp 1642-1656, 1996.
- DAHMEN, W. AND MICCHELLI, C., "Using the refinement equation for evaluating integrals of wavelets", *SIAM J. Numerical Analysis*, v. 30, pp. 507-537, 1993.
- DAUBECHIES, I., "Orthonormal bases of compactly supported wavelets", *Commun. Pure Appl. Math.*, v. 41, pp. 909-996, 1988.
- DAUBECHIES, I., *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992.

DONGARRA, J., LUMSDAINE, A., POZO, R., REMINGTON, A., *IML++ Iterative Methods Library v.1.2, Reference Guide*, 1996. Disponível para download no endereço <http://math.nist.gov/iml++>.

DONOHO, D. L., “Nonlinear Wavelet Methods for Recovery of Signals, Densities, and Spectra from Indirect and Noisy Data”, In: *Proceedings of Symposia in Applied Mathematics*, v. 00, pp. 173-205, Stanford, 1993.

DUGUNDJI, J., “Envelopes and preenvelopes of real waveforms”, *IRE Trans. Inform. Theory*, v. 4, pp. 53-57, 1958.

FLANDRIN, P., “A time-frequency formulation of optimum detection”, *IEEE Trans. On Acoust. Speech Signal Processing*, v. 36, pp.1377-1384, 1988.

FOUFOULA-GEORGIOU, E., KUMAR, P., “Wavelet Analysis in Geophysics: An Introduction”, In: *Wavelets in Geophysics*, v. 4, *Wavelet Analysis and its Applications*, Academic Press, pp. 1-43, 1994.

FROMMEN, J., MALLAT, S., “Second generation compact image coding with wavelets”, in *Wavelets: A tutorial in theory and applications*, C.K.Chui, San Diego, Academic Press, 1992.

GABOR, D., “Theory of communication”, *Journal of the IEE*, v. 93, pp. 429-457, 1946.

GENDRIN, R., ROBERT, P., “Temps de groupe et largeur de bande de signaux modulés simultanément en amplitude et en fréquence”, *Ann. Télécommun.*, v. 37, pp. 289-297, 1982.

GERSHO, A., *Vector quantization and signal compression*, Academic Publishers, Boston, 1995.

GOLUB, G.H., VAN LOAN, C.F., *Matrix Computations*, 2. ed., Baltimore, The Johns Hopkins University Press, 1989.

GONZALEZ, P., CABALEIRO, J.C., PENA, T.F., “Parallel iterative scheme for solving BEM systems using fast wavelet transforms”, *Developments in Engineering Computational Technology*, Edinburgh, Civil-Comp Press, 2000.

GRATTAN-GUINNESS, I., *Joseph Fourier 1768-1830*, Cambridge, Mass., MIT Press, 1972.

HAAR, A., "Zur Theorie der orthogonalen Funktionensysteme", *Mathematische Annalen*, v. 69, pp.331-371, 1910.

HACKBUSCH, W., NOWAK, Z.P., "On the fast matrix multiplication in the boundary element method by panel clustering", *Numerische Mathematik*, v. 54, pp. 463-491, 1989.

HAGELBERG, C. R., GAMAGE, N. K. K., "Applications of Structure Preserving Wavelet Decompositions to Intermittent Turbulence: A Case Study", In: *Wavelets in Geophysics*, v. 4, *Wavelet Analysis and its Applications*, Academic Press, pp. 45-80, 1994.

HORNE, R., "QuickTime: Making Movies With Your Macintosh", ISBN 1-55958-242-1.

HUH, Y., HWANG, J.J., RAO, K.R., "Block wavelet transform coding of images using classified vector quantization", *IEEE Trans. Circ. and Syst. for Video Tech.*, v. 5, pp. 63-67, 1995.

*IEEE Standard for Binary Floating-Point Arithmetic*, ANSI/IEEE Standard 754-1985, New York, Institute of Electrical and Electronic Engineers, 1985.

INOUE, H., KISHIMOTO, K., SHIBUYA, T., "Experimental Wavelet Analysis of Flexural Waves in Beams", *Experimental Mechanics*, v. 36, n. 3, pp. 212-217, 1996.

JEONG, J., WILLIAMS, W. J., "A new formulation of generalizes discrete-time time-frequency distributions", *Proc. IEEE ICASSP-91*, pp. 3189-3192, 1992.

KANE, J.H., KEYES, D.E., PRASAD, K.G., "Iterative equation solution techniques in boundary element analysis", *Int. J. Num. Meth. Eng.*, **31**, 1511-1536, 1991.

KIRCHHOFF, G., BUNSEN, R., *Annalen der Physik und der Chemie* (Poggendorff), v. 110, pp. 161-189, 1860.

KREIDER, D.L., KULLER, R.G, OSTBERG, D.R., PERKINS, F.W., *Introdução à Análise Linear*, v.1, Rio de Janeiro, Ao Livro Técnico, 1972.

LAGE, C., "Concept-oriented design of numerical software", In: *14th GAMM Seminar on Concepts of Numerical Software*, Christian-Albrechts-Universitat Kiel, 1998.

LATTO, A., RESNIKOFF, H.L., TENENBAUM, E., *The evaluation of connection coefficients of compactly supported wavelets*, Aware Technical Report AD910708,

1991 e *Proceedings of the French - USA Workshop on Wavelets and Turbulence*, New York, Springer-Verlag, 1992.

LE GALL, D. J., "MPEG: A video compression standard for multimedia applications", *Communications of ACM*, v. 34, pp. 47-58, 1991.

LOUGHLIN, P., TACER, B., "On the amplitude and frequency modulation decomposition of signals", *J. Acoust. Soc. Amer.*, v. 100, pp. 1594-1601, 1996.

MACROMEDIA INC, "Indeo Video Technology Backgrounder", *Macromedia European Technical Newsletter*, v. 2, Macromedia Europe, 1993.

MALLAT, S. G., *Multiresolution approximation and wavelets*, GRASP Lab., Department of Computer and Information Science, University of Pennsylvania, 1986.

MALLAT, S. G., "A theory for multiresolution signal decomposition: the wavelet representation", *Comm. Math. Phys.*, v. 110, pp. 601-615, 1987.

MANDAL, M.K., PANCHANATHAN, S., ABOULNASR, T., "Choice of wavelets for image compression", *Lecture notes in Computer Science*, v. 1133, pp. 239-249, 1993.  
(email mandalm@marge.genie.uottawa.ca)

MANSUR, W.J., SILVA, W.L., "Efficient Time Truncation in Two-Dimensional BEM Analysis of Transient Wave Propagation Problems", *Earthquake Engineering and Structural Dynamics*, v. 21, n. 1, pp. 51-63, 1992.

MANSUR, W.J., ARAUJO, F.C., MALAGHINI, J.E.B., "Solution of BEM systems of equations via iterative techniques", *Int. J. Num. Meth. Eng.*, v. 33, pp. 1823-1841, 1992.

MARGENOU, H., HILL, R. N., "Correlation between measurements in quantum theory", *Prog. Theoret. Phys.*, v. 26, pp. 722-738, 1961.

MEYER, Y., *Wavelets: algorithms and applications*, 1 ed., Philadelphia, SIAM, 1993.

MEYER, Y., "Principe d'incertitude, bases hilbertiennes et algébres d'opérateurs", *Séminaire Bourbaki*, n. 662, Paris, 1985.

MICROSOFT CORPORATION, *Microsoft Multimedia Developers Kit: Programmers Reference*, Redmond, Microsoft Press, 1991.

MORDANI, R., DAVIDSON, J., BOAG, S., *Java API for XML processing - version 1.1 Final Release*, Sun Microsystems, 2001.

- NABORS, K., KORSMEYER, F.T., LEIGHTON, F.T., WHITE, J., "Preconditioned, adaptive, multipole-accelerated iterative methods for three-dimensional potential integral equations of the first kind", *SIAM J. Sci. Comp.*, v. 15, pp.713-735, 1994.
- OPPENHEIM, A.V., WILLSKY, A.S., *Signals and Systems*, 1 ed., New Jersey, Prentice-Hall, 1983.
- OPPENHEIM, A. V., SCHAFER, R. W., *Discrete-Time Signal Processing*, 1 ed., New Jersey, Prentice Hall, 1989.
- OJANEN, H., *Orthonormal Compactly Supported Wavelets with Optimal Sobolev Regularity*, Rutgers Univ. Math. Dept., 1998.
- OJANEN, H., *Remarks on the Sobolev regularity of wavelets and interpolation schemes*. Research Reports A305, Helsinki Univ. of Technology, Inst. of Math., 1991.
- OSWALD, J., "The Theory of analytic band-limited signals applied to carrier systems", *IRE Trans. Commun. Theory*, v. CT-3, pp. 244-251, 1956.
- PAGE, C. H., "Instantaneous power spectra", *Jour. Appl. Phys.*, v. 23, pp. 103-106, 1952.
- PANTER, P., *Modulation, Noise and Spectral Analysis*, New York, McGraw-Hill, 1965.
- PATTISON, T., *Programming Distributed Applications with COM+ and Microsoft Visual Basic 6.0*, 2. ed., Redmond, Microsoft Press, 2000.
- PETERSDORFF, T., SCHWAB, C., "Fully discrete multiscale Galerkin BEM", In: *Multiscale Wavelet Methods for Partial Differential Equations*, v. 6, *Wavelet Analysis and its Applications*, pp. 287-346, Academic Press, 1997.
- PICINBONO, "Représentation des signaux par amplitude et phase instantanées", *Ann. Télécommun.*, v. 38, pp. 179-190, 1983.
- PICINBONO, B., *Principles of Signals and Systems*, London, Artech House, 1988.
- PRASAD, K.G., KANE, J.H., KEYES, D.E., BALAKRISHNA, C., "Preconditioned Krylov solvers for BEA", *Int. J. Num. Meth. Eng.*, v. 37, pp. 1651-1672, 1994.
- PRITCHARD, J., *COM and CORBA side by side: Architectures, strategies and implementations*, 1. ed., Reading, Massachussets, Addison-Wesley, 1997.

QIAN,S., CHEN, D., *Joint Time-Frequency Analysis: methods and applications*, 1 ed., New Jersey, Prentice-Hall, 1996.

RAO, S., *Mechanical Vibrations*, 3. ed., Reading, Massachussets, Addison Wesley, 1995.

RIHACZEK, K. S., "Signal energy distribution in time and frequency", *IEEE Trans. Info Theory*, v. 14, pp. 368-374, 1968.

RIOUL, O., "On the choice of wavelet filters for still image compression", *Proceedings of ICASSP'93*, v. 5, pp. 550-553, 1993.

ROKHLIN, V., "Rapid solution of integral equations of classical potential theory", *J. Comp. Phys.*, v. 60, pp. 187-207, 1983.

RUBIN, W.L., DI FRANCO, J.V., "Analytic representation of wide-band radio frequency signals", *J. Franklin Inst.*, v. 275, pp. 197-204, 1966.

SAAD, Y., SCHULTZ, M.H., "GMRES: A Generalized Minimal Residual Method for Solving Nonsymmetric Linear Systems." *SIAM J. Sci. Stat. Comp.*, n. 7, v. 3, pp. 856-869, 1986.

SAAD, Y., *Iterative Methods for Sparse Linear Systems*, New York, PWS Publishing, 1996.

SCHWARTZ, M., *Information Transmission Modulation and Noise*, New York, McGraw-Hill, 1987.

SHAPIRO, J.M., "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Transactions on Image Processing*, v. 41, n. 12, pp. 3445-3462, 1993.

SRINIVASAN, R., *XDR: External Data Representation standard - RFC 1832*, Sun Microsystems, Inc., August 1995. Disponível para download no site <http://community.roxen.com/developers/idocs/rfc/>

STANKOVIC, L. J., "Highly concentrated time-frequency distributions: pseudo quantum signal representation", *IEEE Trans. on signal processing*, v. 45, n. 3, 1997.

TAUB, H., SCHILLING, D.L., *Principles of Communication Systems*, New York, McGraw-Hill, 1986.

VAKMAN, D., "On the definition of concepts of amplitude, phase and instantaneous frequency of a signal", *Radio Eng. Electron. Phys.*, v. 39, pp. 754-759, 1972.

VAKMAN, D., "On the analytic signal, the Teager-Kaiser energy algorithm and other methods for defining amplitude and frequency", *IEEE Trans. Signal Processing*, v. 44, pp. 791-797, 1996.

VILLE, J., "Theorie et applications de la notion de signal analytique", *Cables et Transmission*, v. 2, n. 1, pp. 61-74, 1946.

VILLE, J., *Théorie et applications de la notion de signal analytique*, C&T, Laboratoire de Télécommunications de la Société Alsacienne de Construction Mécanique, 1948.

VOELCKER, H., "Toward a unified theory of modulation, Part 2, Zero manipulation", In: *Proc. IEEE*, v. 54, pp. 737-755, 1966.

WALLACE, G.K., "The JPEG still picture compression standard", *Communications of ACM*, v. 34, pp. 31-44, 1991.

WHIPPLE, E., ROSS, R., BUMPUS, F., *Kylix Development*, 1. ed. , New York, Wordware Publishing, 2001.

WICKERHAUSER, M.V., *Lectures on wavelet packets algorithms*, Dept. of Math., Washington University, 1991.

WICKERHAUSER, M.V., *Adapted wavelet analysis from theory to software*, A. K. Peters, 1994.

WIGNER, E. P., "On the quantum correction for thermodynamic equilibrium", *Phys. Ver.*, v. 40, pp. 749-759, 1932.

WIGNER, E. P., "Quantum-mechanical distribution functions revisited", In: W. Yourgrau and A. van der Merwe, *Perspectives in Quantum Theory*, pp. 25-36, MIT Press, 1971.

WILLIAMS, J. R., AMARATUNGA, K., "Introduction to wavelets in engineering", *International Journal for Numerical Methods in Engineering*, v.37, pp. 2365-2388, 1994.

WITTEN, I.H., NEAL, R.M., CLEARY, J.G., "Arithmetic coding for data compression", *Communications of ACM*, v. 30, pp. 520-539, 1987.

WOZENCRAFT, J. M., JACOBS, I. M., *Principles of Communication Engineering*, New York, Wiley, 1965.

ZHAO, Y., ATLAS, L. E., MARKS, R. J., "The use of cone-shaped kernels for generalizes time-frequency representations of nonstationary signals", *IEEE Trans. Acoust., Speech, Signal Processing*, v. 38, pp. 1084-1091, 1990.

ZIV, J., LEMPEL, "A universal algorithm for sequential data compression", *IEEE Transactions on Information Theory*, v. IT-23, pp. 337-343, 1977.