Wavelets Interpolatórias como
Ferramenta para Resolução
Adaptativa de EDP's Hiperbólicas

Alice de Jesus Kozakevicius

TESE APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO DE DOUTOR
EM
MATEMÁTICA APLICADA

Área de concentração: Matemática Aplicada
Orientador: Prof. Dr. Luis Carlos de Castro Santos

Durante a elaboração deste trabalho a autora recebeu auxílio financeiro da CAPES-PICDT-UFSP.

São Paulo, outubro de 2002

Wavelets Interpolatórias como Ferramenta para Resolução Adaptativa de EDP's Hiperbólicas

Este exemplar corresponde à redação
final da tese devidamente corrigida
e defendida por Alice de Jesus Kozakevicius
e aprovada pela comissão julgadora.

São Paulo, 17 de novembro de 2002.

Banca examinadora:

- Prof. Dr. Luis Carlos de Castro Santos (Presidente) - IME - USP
- Prof. Dr. Saulo Rabello Maciel de Barros - IME - USP
- Prof. Dr. Ernani Vitillo Volpe - EP - USP
- Prof. Dra. Sônia Maria Gomes - IMECC - UNICAMP
- Prof. Dr. Haroldo Fraga de Campos Velho - INPE

Resumo

Neste trabalho desenvolve-se um esquema original e eficiente para resolução de sistemas hiperbólicos multidimensionais de forma adaptativa através da associação de duas técnicas: transformadas wavelets interpolatórias e esquemas de diferenças finitas essencialmente não oscilatórios (ENO) que utilizam valores pontuais.

O método está baseado na aplicação de uma transformada wavelet interpolatória à solução da equação em cada instante de tempo. A representação multiresolução obtida para a solução é truncada, gerando uma representação esparsa associada a uma malha esparsa. Sobre esta malha faz-se a evolução do operador diferencial, reduzindo-se o custo computacional.

O operador é resolvido por um esquema de alta resolução que utiliza interpolação não linear ENO no cálculo do fluxo numérico de forma conservativa. O método atinge alta ordem de aproximação em regiões onde a solução é suave através da busca do melhor estêncil de interpolação para cada ponto, no qual o fluxo é calculado.

A extensão do método unidimensional para o multidimensional se dá através de uma fatoração por direção com passo no tempo fracionário, fazendo com que a mesma transformada wavelet unidimensional possa ser utilizada e evitando o cálculo dos autovetores da matriz jacobiana do sistema.

Uma série de testes para as equações básicas, como advecção e Burgers; e para o sistema de equações de Euler, foi realizada comprovando a eficiência da técnica.

Abstract

In this work we develop an efficient scheme for solving adaptively multidimensional hyperbolic systems combining two different techniques: Interpolatory wavelet transforms and essentially non-oscillatory (ENO) finite difference schemes for point values.

The method is based on the application of an interpolating wavelet transform using polynomial interpolation on dyadic grids to the solution of the differential equation in each time step. The multiresolution representation obtained for the solution is thresholded and a sparse point representation is obtained. The evolution of the differential operator is done in this sparse representation.

The operator is solved by a high order, non linear finite difference scheme, where the numerical fluxes are evaluated by an ENO approximation, which chooses the locally smoothest stencil among many possibilities for each point of the sparse grid. In this way the method achieves high order accuracy in smooth regions of the solution.

The extension of the one-dimensional case for the multidimensional is done by a time splitting method, where each direction is evaluated separately. This avoids both the calculation of the eigenvectors of the Jacobian, and the definition of a two-dimensional wavelet transform.

A series of tests, for basic equations such as advection and Burgers; and for the Euler systems, is performed confirming the efficiency of the technique.

Dedicatória

Dedico este trabalho às mulheres fortes e guerreiras de minha família.

Em especial, à minha mãe, Marlene, à minha avó, Emília, e à minha prima, Ana Lúcia.
Grandes exemplos de fibra, coragem, irreverência e perseverança em minha vida.

Agradecimentos

Em primeiro lugar, gostaria de agradecer ao Prof. Luis Carlos de Castro Santos, meu orientador e amigo, por ter assumido a orientação deste trabalho com muito otimismo e dedicação, faltando menos de um ano para meu prazo de bolsa se esgotar. Sem a sua experiência, maturidade e firmeza, eu não teria concluído esta etapa...

Ao Prof. Jorge Aragona e à Profa. Roseli Fernandez pela amizade e incentivo desde a minha chegada ao IME. Ao Prof. Saulo de Barros pelas inúmeras dicas durante todo o meu processo de formação, e por me ensinar que o importante é resolver problemas, não criá-los. À Profa. Sônia Gomes pela indicação do artigo que deu rumo à tese e a Profa. Magda Kaibara pela indicação de um dos artigos de Ami Harten que me fez entender o que era uma média celular e pela atenção e carinho com que sempre respondeu a todas as minhas perguntas. Ao Prof. Marco Túlio de Vilhena, meu querido orientador de mestrado, que, mesmo à distância, colaborou com temas para aplicações futuras, me incentivando sempre a seguir em frente.

Aos colegas e amigos do grupo de análise numérica, Cláudia Ines Garcia, Juan Egana, Raul Jimenez e Santos Alberto Remigio pelas inúmeras dicas em FORTRAN e pelo apoio durante as disciplinas. À Maria Sílvia Moura pelas dicas em estatística. À Lurdinha e Osmar pela ajuda na chegada a São Paulo e ao Leandro, pelo convívio durante todos os quatro primeiros anos nesta cidade que tanto me assustou no início.

À amiga Ângela Dewes Moura, pelos conselhos e pela paciência em me escutar nos momentos de crise e desilusão com este trabalho. Aos amigos Maria do Carmo Costa, Maité Kulesza, Rudimar Luiz Nós, Fabio Martinez, Claus Matsushigue, Clélia Terra e sua querida família, pela acolhida em São Paulo, pelo incentivo e por todo o carinho.

À minha tia Magda pelo apoio e à minha madrinha Cleonice por toda a torcida, sofrendo e vibrando comigo, durante todas as etapas deste trabalho.

Ao grupo de pesquisa responsável pelo Laboratório de Análise Numérica e Astrofísica da Universidade Federal de Santa Maria pelo suporte durante minhas idas e vindas de Santa Maria para São Paulo. Aos colegas e amigos do departamento de matemática da UFSM, em especial, à Cláudia Pansonato, Denilson Gomes, Janice Racheli, Alex Schmidt e Maria Inês Copetti por todo o apoio e ajuda com disciplinas, dicas e sugestões para a tese, correções do texto, ajuda com softwares e tantas outras coisas que eu precisei ao longo deste tempo.

Ao Elier, pelo carinho e atenção nesta etapa final.

Quero agradecer ainda a inúmeras outras pessoas que colaboraram para que hoje este trabalho pudesse ser concluído: colegas de curso, colegas de sala, colegas de bate-papo no café, professores, funcionários de secretaria do MAP, da biblioteca, do setor de alunos... enfim, amigos dos quais vou guardar uma recordação muito especial para sempre.

Sumário

Lista de Figuras	ix
Lista de Tabelas	xi
Introdução	1
1 Representação Multiresolução	9
1.1 Representação Multiresolução para Valores Pontuais	10
1.1.1 Interpolação Lagrangeana Centrada	12
1.1.2 Parte Computacional	15
1.1.3 Interpolação ENO	22
1.1.4 Interpolação ENO para Valores Pontuais	24
1.2 Representação Multiresolução para Médias Celulares	25
1.2.1 Interpolação Lagrangeana Centrada	28
1.2.2 Interpolação ENO com Resolução Intra-Célula(Subcell)	29
1.2.3 Parte Computacional	31
2 Compressão de Dados	39
2.1 Truncamento	40
2.2 Estratégias de Compressão	43
2.2.1 Estratégia 1 $\epsilon_k = \frac{\epsilon}{2^k}$	44
2.2.2 Estratégia 2 $\epsilon_k = \frac{\epsilon}{2^{n_0-k}}$	52
2.2.3 Estratégia 3 $\epsilon_k = \frac{\epsilon}{2^{L-k}}$	58
2.2.4 Análise Comparativa	63
2.3 Representação Pontual Esparsa	67
2.3.1 Reconstrução	68

3	Método Adaptativo	75
3.1	Algoritmo Adaptativo	76
3.1.1	Representação Multiresolução	77
3.1.2	Truncamento	77
3.1.3	Extensão e Representação Esparsa	77
3.1.4	Discretização do operador	81
3.1.5	Evolução Temporal	85
3.1.6	Reconstrução	85
3.2	Exemplos Numéricos	88
3.2.1	Resolução da equação de advecção	88
3.2.2	Resolução da equação de Burgers	94
3.3	Atualização da Malha Esparsa	104
3.3.1	Exemplos Numéricos	104
4	Método de Alta Resolução Adaptativo	115
4.1	Método de Alta Resolução	116
4.1.1	Aproximação Conservativa Unidimensional	116
4.1.2	Reconstrução via Primitiva	117
4.1.3	Aproximação ENO para Primitiva	118
4.2	Formulação por Diferenças Finitas	119
4.2.1	Esquema ENO-Roe	120
4.2.2	Esquema ENO-Lax-Friedrich	121
4.2.3	Condições de Contorno	122
4.2.4	Malhas Wavelets	123
4.2.5	Evolução Temporal	124
4.3	Exemplos Unidimensionais	127
4.4	Extensão à Duas Dimensões	148
4.4.1	Exemplos Escalares	151
4.4.2	Exemplo Vetorial - Sistema Hiperbólico	168
	Conclusões	177

A	181
A.1 Cálculo dos coeficientes γ_l	181
Referências Bibliográficas	186

Lista de Figuras

1	Principais Referências da Área e o Trabalho Apresentado	7
1.1	Valores Pontuais de $v(x)$ no nível mais refinado, $n_0 = 7$. $N_0 = 2^{n_0} = 132$ pontos)	15
1.2	Valores Pontuais de $v(x) \times [0, 1]$ nos quatro níveis de fatoração	16
1.3	Detalhes de $v(x) \times [0, 1]$ nos diferentes níveis	17
1.4	Valores Pontuais de $u(x) \times [0, 1]$ no nível mais refinado, $k=0$, com 2^7 pontos	18
1.5	Valores Pontuais de $u(x) \times [0, 1]$ em quatro níveis diferentes	18
1.6	Detalhes da função em cada um dos níveis da fatoração	19
1.7	Valores Pontuais da função $w(x) \times [0, 1]$ no nível mais refinado	20
1.8	Valores Pontuais da função $w(x) \times [0, 1]$ nos níveis $k = 1$ a $k = 4$	20
1.9	Detalhes da função nos níveis $k = 1, 2, 3, 4$	21
1.10	Possibilidades de estêncil contendo x_j e x_{j+1} . $im(j)$ indica o índice do primeiro ponto do estêncil	22
1.11	Interpolações ENO e Lagrangeana Centrada de grau 3	25
1.12	Médias Celulares da função $v(x)$ no nível mais refinado, $n_0 = 7$	32
1.13	Níveis de resolução de $v(x)$, $n_1 = 6$ e $n_2 = 5$	32
1.14	Níveis de resolução de $v(x)$, $n_3 = 4$ e $n_4 = 3$	33
1.15	Detalhes de $v(x)$ nos níveis 1 e 2	33
1.16	Detalhes de $v(x)$ nos níveis 3 e 4	33
1.17	Médias Celulares da função $u(x)$ no nível mais refinado, $n_0 = 7$	34
1.18	Níveis de Resolução da função $u(x)$, $n_1 = 6$ e $n_2 = 5$	35
1.19	Níveis de Resolução da função $u(x)$, $n_3 = 4$ e $n_4 = 3$	35
1.20	Detalhes da função $u(x)$ nos níveis 1 e 2	36
1.21	Detalhes da função $u(x)$ nos níveis 3 e 4	36
1.22	Médias Celulares da função $w(x)$ no nível mais refinado	37
1.23	Fatoração de $w(x)$ em quatro níveis	37

1.24	Detalhes de $w(x)$ nos quatro níveis de fatoração	38
2.1	Operadores de truncamento	41
2.2	TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$	45
2.3	TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$	45
2.4	TWP3 $v(x)$:Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$	46
2.5	TWP3 $v(x)$:Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$	46
2.6	TWP5 $v(x)$:Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$	47
2.7	TWP5 $v(x)$:Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$	47
2.8	TWP1 $u(x)$: Posições significativas, independente do valor de ε	48
2.9	TWP3 $u(x)$: Posições significativas, independente do valor de ε	48
2.10	TWP5 $u(x)$:Posições significativas, independente do valor de ε	49
2.11	TWP1 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	50
2.12	TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}	50
2.13	TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}	51
2.14	TWP5 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	51
2.15	TWP1 $v(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}	52
2.16	TWP1 $v(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}	53
2.17	TWP3 $v(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	53
2.18	TWP5 $v(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} e 10^{-3}	54
2.19	Posições significativas para $u(x)$ independente do valor de ε : TWP1, TWP3 e TWP5	55
2.20	TWP1 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-3} e 10^{-2}	56
2.21	TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}	56
2.22	TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}	57
2.23	TWP5 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	57
2.24	TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-3}	58
2.25	TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-2}$	59
2.26	TWP3 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-4}	59
2.27	TWP5 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-4}	59
2.28	Posições significativas para $u(x)$ independente do valor de ε : TWP1, TWP3 e TWP5	60
2.29	TWP1 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	61
2.30	TWP3 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}	62
2.31	TWP5 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$ e 10^{-4}	62

2.32	TWP5 $w(x)$: Posições Significativas $\varepsilon=10^{-3}$ e 10^{-2}	63
2.33	Diagrama com os operadores envolvidos na RPE	68
2.34	TWP1 $v(x)$: Coeficientes Significativos e RPE	70
2.35	TWP3 $v(x)$: Coeficientes Significativos e RPE	70
2.36	TWP5 $v(x)$: Coeficientes Significativos e RPE	70
2.37	TWP1 $u(x)$: Coeficientes Significativos e RPE	71
2.38	TWP3 $u(x)$: Coeficientes Significativos e RPE	71
2.39	TWP5 $u(x)$: Coeficientes Significativos e RPE	71
2.40	TWP1 $w(x)$: Coeficientes Significativos e RPE	72
2.41	TWP3 $w(x)$: Coeficientes Significativos e RPE	72
2.42	TWP5 $w(x)$: Coeficientes Significativos e RPE	72
3.1	Diagrama com os operadores para Evolução Temporal na RPE	77
3.2	Pontos de Segurança	80
3.3	Posições Possíveis entre Três Pontos	81
3.4	Uma Interpolação à Esquerda.	82
3.5	Malhas Homogêneas e Malha Esparsa	82
3.6	Vários Níveis à Esquerda.	83
3.7	Reconstrução Não Uniforme	87
3.8	Solução em $[0,1]$ nos casos 1, 2, 3, 4, 5 e 6	89
3.9	Comparação entre soluções em malhas esparsas distintas	93
3.10	Comparação entre soluções em malhas esparsas distintas	94
3.11	Evolução temporal da solução em $[0,1]$: casos 1 a 6	96
3.12	Comparação entre a solução Exata e a solução cuja RPE não é atualizada	98
3.13	Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 1	102
3.14	Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 2	102
3.15	Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 3	102
3.16	Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 4	103
3.17	Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 5	103
3.18	Número de coeficientes significativos de acordo com o parâmetro <i>adpt</i> nas iterações fixadas para medição	106
3.19	Taxas de compressão de acordo com o parâmetro <i>adpt</i> nas iterações fixadas para o teste	106
3.20	Erros na norma 2 para <i>adpt</i> de 1 a 10 nos instantes $(i + 1)\Delta_i$, $i=0,1,\dots,9$	107
3.21	Erros para malhas uniformes e para malha adaptativa, considerando-se <i>adpt</i> =1, nos instantes de medição	108
3.22	Quotas inferior e superior para o erro obtido para diferentes valores fixados para o parâmetro <i>adpt</i> nos instantes de medição	108

3.23	Número de coeficientes significativos por iteração para $adpt$ de 15 a 100 . . .	109
3.24	Taxas de compressão para $adpt$ de 15 a 100 nas respectivas iterações de teste	110
3.25	Erros para $adpt$ de 15 a 100, nos instantes $(i + 1)\Delta_i$, $i=0,1,\dots,9$	110
3.26	Soluções em $T=0,2441$ para malha fina e malhas com $adpt$ diferentes	111
3.27	Erros para $adpt$ 10, 30 e 100 e erros para malhas uniformes com $N_0 = 2^{10}$ e $N_0 = 2^{11}$ pontos. As 20 medidas foram feitas de 5000 em 5000 passos no tempo ao longo das 100000 iterações	112
3.28	Soluções em $T=2,441$ para malha fina e malhas com $adpt$ diferentes	113
3.29	Soluções em $T=2,441$ exata, para malha fina $N_0=2^{10}, 2^{11}, 2^{13}$ com $adpt=10$. .	114
4.1	Vento da Esquerda para Direita	120
4.2	Vento da Direita para Esquerda	120
4.3	Posições Fantasmas para Cálculo de Fluxo nas Fronteiras	122
4.4	Representação Esparsa Unificada	124
4.5	Variáveis ρ , u , p	128
4.6	Variáveis ρ , u , E , p : PVI SOD ₁	130
4.7	RPE em instantes diferentes: PVI SOD ₁	131
4.8	Número de coeficientes significativos e Taxa de Compressão em cada uma das 8000 iterações realizadas: PVI SOD ₁	131
4.9	Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI SOD ₁	132
4.10	Variáveis ρ , u , E , p : PVI SOD ₂	133
4.11	RPE em instantes diferentes: PVI SOD ₂	134
4.12	Número de coeficientes significativos e Taxa de compressão para cada uma das iterações realizadas: PVI SOD ₂	134
4.13	Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI SOD ₁	135
4.14	Variáveis ρ , u , E , p : PVI LAX	136
4.15	RPE em instantes diferentes: PVI LAX	137
4.16	Número de coeficientes significativos e Taxa de compressão para cada uma das iterações realizadas: PVI LAX	137
4.17	Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI LAX	138
4.18	Variáveis ρ , u , E , p : PVI SWSH	139
4.19	RPE em instantes diferentes: PVI SWSH	140
4.20	Número de coeficientes significativos e Taxa de Compressão para cada uma das iterações realizadas: PVI SWSH	140
4.21	Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI SWSH	141
4.22	Variáveis ρ , u , E , p : PVI DCI.	142
4.23	RPE em instantes diferentes: PVI DCI.	143
4.24	Número de coeficientes significativos e Taxa de Compressão para cada uma das iterações realizadas: PVI DCI.	143

4.25	Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI DCI.	144
4.26	Variáveis ρ , u , E , p : PVI DEIB.	145
4.27	RPE em instantes diferentes: PVI DEIB.	146
4.28	Número de coeficientes significativos e Taxas de Compressão para cada iteração realizada: PVI DEIB.	146
4.29	Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI DEIB.	147
4.30	Fatoração por Direção, Evolução no Tempo com Passo Fracionário	150
4.31	Equação de Advecção - RPE na direção x	154
4.32	Malha Esparsa para a equação de Advecção - direção x	155
4.33	Equação de Advecção - RPE na direção y	156
4.34	Malha Esparsa para a equação de Advecção - direção y	157
4.35	Equação de Burger - RPE na direção x	158
4.36	Malha Esparsa para a equação de Burgers - direção x	159
4.37	Equação de Burgers - RPE na direção y	160
4.38	Malha Esparsa para a equação de Burgers - direção y	161
4.39	Malha Esparsa para a Equação de Burgers - direções x e y	162
4.40	Equação de Burgers - RPE na direção x	164
4.41	Malha Esparsa para a equação de Burgers - direção x	165
4.42	Rotação do Cone - RPE na direção y	166
4.43	Malha Esparsa para a rotação do Cone - direção y	167
4.44	Densidade-Configuração 10	170
4.45	Densidade-Configuração 10	171
4.46	Número de significativos e Malha Esparsa em y - Conf. 10	172
4.47	Densidade-Configuração 8	173
4.48	Densidade-Configuração 8	174
4.49	Número de significativos e Malha Esparsa em y - Conf. 8	175

Lista de Tabelas

1.1	Valor máximo dos detalhes de $v(x)$ por nível	16
1.2	Módulo máximo dos detalhes por nível	19
1.3	Módulo máximo dos detalhes por nível	21
1.4	Máximo dos detalhes de $v(x)$ por nível	32
1.5	Valor máximo dos detalhes de $u(x)$ por nível	34
1.6	Valor máximo dos detalhes de $w(x)$ por nível	36
2.1	Número total de coeficientes significativos e Taxas de compressão	48
2.2	Número total de coeficientes significativos e Taxas de compressão	49
2.3	Número total de coeficientes significativos e Taxas de compressão	52
2.4	Número total de coeficientes significativos e Taxas de compressão	54
2.5	Número total de coeficientes significativos e Taxas de compressão	55
2.6	Número total de coeficientes significativos e Taxas de compressão	58
2.7	Número total de coeficientes significativos e Taxas de compressão	60
2.8	Número total de coeficientes significativos e Taxas de compressão	61
2.9	Número total de coeficientes significativos e Taxas de compressão	63
2.10	Número de Significativos, função $v(x)$ -estratégias 1, 2 e 3	64
2.11	Taxas de compressão, função $v(x)$ -estratégias 1, 2 e 3	65
2.12	Número de Significativos, função $w(x)$ -estratégias 1, 2 e 3	65
2.13	Taxas de compressão, função $w(x)$ -estratégias 1, 2 e 3	66
2.14	Número de Significativos, função $u(x)$ -estratégias 1, 2 e 3	66
2.15	Taxa de Compressão, função $u(x)$ -estratégias 1,2 e 3	66
3.1	Taxas de compressão para advecção	90
3.2	Número de Coeficientes Significativos para advecção	90
3.3	Erro na norma 2 para equação de advecção	91
3.4	Erro na norma 2 para equação de advecção: solução com $adpt=0$ acima e com $adpt=1$ abaixo	92
3.5	Número de Coeficientes Significativos para $adpt=0$ e $adpt=1$	93

3.6	Taxas de compressão para cada caso para Burgers	95
3.7	Número de Coeficientes Significativos para cada caso	97
3.8	Erro para cada um dos 6 casos para equação de Burgers	97
3.9	Número de Coeficientes Significativos para $adpt=0$ e $adpt=1$	98
3.10	Erro na norma 2 para equação de advecção: solução com $adpt=0$ acima e com $adpt=1$ abaixo	99
3.11	Módulo máximo do detalhe por tipo de Transformação	100
3.12	Valores de Corte por nível para cada Estratégia	100
3.13	Taxas de Compressão e Quantidade de Significativos para cada uma das estratégias - Preditor Linear	101
3.14	Taxas de Compressão e Quantidade de Significativos para cada uma das estratégias - Preditor Cúbico	101
3.15	Número de coeficientes significativos por iteração para $adpt$ de 1 a 10	105
3.16	Número de coeficientes significativos para $adpt=15,20,25,30,50$ e 100	109
3.17	Número de significativos para $adpt=10,50$ e 100 em 100000 iterações	112
4.1	Tempo de Processamento	148
4.2	Tempo de execução do programa	176

Introdução

Este trabalho concentra-se na resolução numérica de equações diferenciais da forma:

$$\frac{\partial}{\partial t} \vec{u}(\mathbf{x}, t) + \frac{\partial}{\partial x_i} f_i(\vec{u}(\mathbf{x}, t)) = 0, \quad (1)$$

sendo $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$, $\vec{u} : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^m$, vetor m -dimensional das quantidades conservadas, ou variáveis de estado; e $f_i : \mathbb{R}^m \rightarrow \mathbb{R}^m$, $i = 1, 2, 3$. funções de fluxo.

Equações do tipo (1) unidimensionais são ditas *hiperbólicas*, se a matriz jacobiana $m \times m$, $f'_1(u)$, da função de fluxo tiver a seguinte propriedade: para cada valor de u , os autovalores de $f'_1(u)$ são reais e a matriz é diagonalizável, isto é, existe um conjunto de m autovetores linearmente independentes. No caso multidimensional, para que se tenha hiperbolicidade é preciso que qualquer combinação linear dos fluxos jacobianos, $\alpha_1 f'_1(u) + \alpha_2 f'_2(u) + \alpha_3 f'_3(u)$, com $\alpha_i \in \mathbb{R}$, seja diagonalizável com autovalores reais [29].

Matematicamente a característica mais interessante desses sistemas é que eles admitem *ondas de choque* em suas soluções, descontinuidades que podem se formar mesmo a partir de dados iniciais suaves. Estas descontinuidades são, na verdade, grandes variações nos valores das soluções em regiões muito pequenas relativamente à dimensão espacial do problema.

Muitos problemas práticos em física e engenharia envolvem quantidades conservadas e são modelados através de equações diferenciais deste tipo. Um sistema hiperbólico não linear de leis de conservação de grande importância é a equação de Euler da dinâmica dos gases, caso particular da equação de Navier-Stokes, quando desprezados os termos referentes à viscosidade do fluido e à condução de calor. Este sistema em um espaço d dimensional possui $m = d + 2$ equações correspondentes à conservação de massa, energia e quantidade de movimento em cada direção.

As equações de Euler são usadas extensivamente em aerodinâmica, por exemplo na modelagem do escoamento de ar ao redor de uma aeronave ou qualquer outro veículo. Um problema bidimensional típico é o escoamento de ar sobre um aerofólio que é simplesmente uma seção transversal de uma asa. A distribuição de pressão ao longo da superfície do aerofólio é de particular interesse, pois é através dela que são calculadas as componentes horizontal e vertical da força total sobre a asa, dados fundamentais na avaliação de seu desempenho.

Meteorologia e previsão do tempo é outra área de dinâmica dos fluidos, na qual se aplicam leis de conservação. ‘Frentes climáticas’ são essencialmente ondas de choque, pois apresentam grandes variações de pressão e temperatura. No entanto, as escalas envolvidas são muito maiores do que na modelagem do aerofólio e os efeitos dissipativos e viscosos fazem com que estas ondas tenham vários quilômetros.

Na modelagem de problemas em astrofísica [30] são obtidas leis de conservação similares às equações de Euler para a densidade de matéria no espaço. Uma galáxia espiral, por exemplo, é formada por braços de alta densidade que se alternam com braços de baixa densidade, separadas por ‘descontinuidades’ que são novamente ondas de choque que se propagam. Neste contexto o comprimento do choque deve ser de 2 ou 3 anos-luz. Uma vez que o diâmetro da galáxia é da ordem de 10^5 anos-luz, este comprimento é relativamente muito pequeno.

A modelagem da dinâmica de uma única estrela ou do plasma em um reator de fusão também exige leis de conservação. Estas agora envolvem efeitos eletromagnéticos associados a dinâmica do fluido. As equações da magnetohidrodinâmica são um exemplo de sistema hiperbólico utilizado nestes modelos. Sistemas de leis de conservação hiperbólicos surgem ainda naturalmente em uma grande variedade de outras aplicações: no estudo de explosões, na propagação de ondas em meios elásticos, no escoamento multifásico em materiais porosos, no escoamento de geleiras, na separação de elementos químicos por cromatografia, entre outros.

Embora disponha-se de poucas soluções exatas, tem-se um grande volume de teoria sobre a estrutura destas equações e sobre o comportamento de suas soluções. Muito da teoria foi desenvolvido com o objetivo de estudar as equações de Euler e muitos dos métodos numéricos foram desenvolvidos especificamente para este sistema. Existem, no entanto, dificuldades específicas, tanto do ponto de vista teórico quanto numérico, associadas à resolução de sistemas hiperbólicos em geral, que não são encontradas em outras equações, e que devem ser tratadas com muito cuidado no desenvolvimento de métodos numéricos.

Soluções descontínuas, que desenvolvem ondas de choque como mencionado anteriormente, não satisfazem a equação diferencial parcial no sentido clássico em todos os pontos do domínio, apenas no sentido *fraco*, [6]; uma vez que as derivadas não estão definidas nos pontos onde a solução é descontínua. Isto faz com que métodos numéricos, baseados na hipótese de suavidade da solução e que utilizam simples diferenças finitas, produzam resultados desastrosos em regiões onde se encontram descontinuidades da solução.

Uma outra dificuldade é a possibilidade de não unicidade da solução. Frequentemente existe mais de uma solução fraca (não suave) para a lei de conservação com o mesmo dado inicial. O surgimento de uma outra solução é resultado da equação ser apenas um modelo simplificado, no qual vários efeitos físicos foram ignorados. Em particular, leis de conservação excluem difusão e efeitos viscosos. No entanto, em várias situações deve-se usar o conhecimento do que está sendo desprezado, para que se possa escolher a solução fisicamente coerente. Para dinâmica dos gases, por exemplo, pode-se recorrer à segunda lei da termodinâmica, segundo a qual a entropia é uma função não decrescente. Como as moléculas de

um gás passam por um choque, sua entropia deve crescer. Esta condição é suficiente para identificar qual a solução fisicamente correta.

Na tentativa de calcular estas soluções numericamente, surge ainda um novo conjunto de dificuldades. Como discutido em [29], métodos de primeira ordem, do tipo Godunov, por exemplo, carregam naturalmente uma grande quantidade de ‘viscosidade numérica’ que suaviza a solução da mesma forma que a viscosidade física faria, mas em uma extensão muitas vezes superior à escala real do fenômeno físico. Já os métodos de segunda ordem, do tipo MacCormack, eliminam os efeitos dissipativos, mas introduzem efeitos dispersivos que geram grandes oscilações nas soluções numéricas.

O ideal seria obter métodos numéricos que produzissem automaticamente aproximações mais precisas possível nas regiões de grandes variações da solução, sem a necessidade de incluir na equação restrições ou condições específicas para as regiões de descontinuidade. Métodos com esta característica são ditos métodos de captura de choque.

Nos últimos vinte anos têm-se obtido muitos progressos neste sentido e uma grande variedade de métodos com estas características tem sido desenvolvida. Algumas propriedades extras importantes que se gostaria de poder incluir são: que o método fosse no mínimo de segunda ordem para soluções suaves e suaves por partes; que não produzisse oscilações espúrias; que possuísse uma forma apropriada de consistência com a forma fraca da lei de conservação necessária para convergência; que possuísse limites não lineares para estabilidade junto com consistência e uma forma discreta de condição de entropia. Na verdade, os métodos são classificados de acordo com as propriedades que eles conseguem agregar. Neste trabalho tem-se especial interesse nos métodos que reúnem as propriedades de precisão e não produção de oscilações, chamados métodos de alta resolução.

Para sistemas lineares hiperbólicos, métodos envolvendo o conhecimento das curvas características da equação desempenham um papel importante, já para problemas não lineares, a generalização desta teoria, muito usada no desenvolvimento de métodos numéricos, é a resolução de *problemas de Riemann* (2), que no caso unidimensional tem a forma:

$$\begin{aligned} \frac{\partial}{\partial t} u(x, t) + \frac{\partial}{\partial x} f(u(x, t)) &= 0, \\ u(x, 0) &= u_0; \quad x < x_0 \\ u(x, 0) &= u_1; \quad x \geq x_0, \\ u_0, u_1 &\text{ constantes,} \end{aligned} \tag{2}$$

que é simplesmente a lei de conservação associada a um problema de valor inicial, cujos dados são constantes por partes e apresentam uma descontinuidade tipo salto. Muitos métodos resolvem problemas de Riemann em seus passos intermediários, conseguindo com isso uma grande quantidade de informação sobre a estrutura local da solução na vizinhança de determinados pontos do domínio, no entanto as expressões obtidas para a discretização do operador diferencial são bastante complicadas e o custo computacional destes métodos acaba sendo muito grande, pois para cada subintervalo (célula) da malha e para cada instante de tempo, um problema de Riemann (2) deve ser resolvido.

Infelizmente, a obtenção de resultados sobre estabilidade não linear, necessários para provar convergência, é praticamente impossível para os métodos usados na resolução das equações hiperbólicas. Apenas para o caso simples de equações escalares se consegue uma análise mais específica. Em particular, segundo [29], é possível mostrar para muitos métodos que a variação total da solução, $TV(u)$, é não crescente com o tempo. $TV(u)$ é definida por:

$$TV(u) = \sup_{\pi} \sum_{j=1}^N |u(\xi_j) - u(\xi_{j-1})|$$

e o supremo é considerado sobre π , o conjunto de todas as partições possíveis para o domínio espacial da equação. Isto é suficiente para a obtenção de alguns resultados de convergência e para garantir a não geração de oscilações espúrias no caso unidimensional. Métodos com esta propriedade são chamados métodos com variação total decrescente (TVD), $TV(v^n) < TV(v^{n-1})$.

Segundo [14], na prática, extensões destes métodos para os casos vetoriais ou de dimensão mais alta produzem ainda bons resultados, mantendo inclusive as propriedades de alta resolução, embora os resultados de convergência não possam ser estendidos. O que se consegue manter é a estabilidade na norma do máximo para discretizações no espaço de alta ordem.

Os métodos de alta resolução, de variação total decrescente e os do tipo Godunov apresentam formulações bastante elaboradas e muitas vezes complicadas para a discretização do operador diferencial. E quanto mais refinada for a malha utilizada, maior será o custo computacional gasto na resolução do problema. Com o objetivo de diminuir este custo, surge uma outra classe de métodos para resolução de leis de conservação hiperbólicas, ditos métodos adaptativos. Estes métodos exploram fortemente o fato da solução ser suave na maior parte do domínio espacial e apresentar fortes variações apenas em pequenas regiões.

Em termos de métodos que envolvem diferenças finitas, para que o operador seja resolvido com precisão nas regiões de suavidade, a malha não necessita ser tão refinada, quanto nas regiões onde se formam as ondas de choque. Em termos de métodos espectrais, isto corresponde a representar a solução com poucas funções da base na região de suavidade.

Assim, além de ajustarem o espaçamento à solução, outra característica dos métodos adaptativos é a existência de algum dispositivo capaz de identificar e selecionar uma região a ser refinada. Na verdade, cada método adaptativo possui sua estratégia para refinar ou modificar o refinamento conforme a solução varia com o tempo.

Neste trabalho, desenvolve-se um método adaptativo explorando-se o fato das *wavelets* serem funções de suporte compacto e das expansões de funções em base wavelet apresentarem coeficientes que dependem localmente da função da base, ou seja, se uma função for perturbada em uma pequena região do domínio, a expansão wavelet desta nova função apresentará modificações em relação à original apenas nos coeficientes associados às wavelets, cujo suporte intersecciona a região onde houve a alteração. Esta propriedade de “localização” das wavelets será utilizada como forma de monitorar as variações da solução da equação diferen-

cial ao longo do tempo e, conseqüentemente, será utilizada como uma forma de selecionar regiões a serem refinadas.

Dada uma representação wavelet de uma função qualquer $u(x)$,

$$u(x) = \sum_k s_{J,k} \varphi_{J,k}(x) + \sum_j \sum_k d_{j,k} \psi_{j,k}(x), \quad (3)$$

$\varphi_{J,k}(x)$ são as *funções escala* no nível menos refinado de representação da função, J ; $\psi_{j,k}(x)$ são as *wavelets* em cada nível mais refinado j ; os coeficientes das funções escala no nível mais grosseiro J , $s_{J,k}$, essencialmente representam a parte mais suave da função $u(x)$, enquanto que os coeficientes wavelet, $d_{j,k}$ contêm informações sobre o comportamento da função em sucessivos níveis mais refinados.

Uma das propriedades mais utilizadas com relação a expansões em bases wavelets é a possibilidade de representar a função $u(x)$ com uma quantidade menor de wavelets, caso os coeficientes $d_{j,k}$ sejam “muito pequenos”. Esta propriedade é denominada compressão. A maneira mais comum de conseguir compressão de uma representação wavelet (3) é truncando a série, extraindo da representação todos os termos cujos coeficientes wavelet possuem valor menor do que algum valor de corte ε fixado. Se o número total de coeficientes na representação for N , obtém-se N_s coeficientes significativos após o truncamento. Através do truncamento da representação wavelet, obtém-se uma maneira automática de se encontrar uma representação esparsa e pode-se utilizar esta representação para calcular valores da função em qualquer ponto.

Esta representação esparsa tem duas vantagens na resolução numérica de equações diferenciais hiperbólicas: é necessário menos memória para guardar a solução, uma vez que são guardados N_s coeficientes ao invés de N , com $N_s < N$; e, caso seja explorada a esparsidade da representação (3), pode-se economizar tempo computacional durante o cálculo do operador.

Muitos enfoques diferentes foram abordados na resolução de equações hiperbólicas dependentes do tempo explorando esparsidade em representações wavelet. Uma maneira é aproximar o operador da equação diferencial em alguma base wavelet. Isto pode ser feito através de métodos de colocação, por exemplo como feito em [3] ou através de métodos de Galerkin, como apresentados em [32], entre outros. O operador será esparsa na base wavelet, mas para que os métodos possam ser eficientes, o número de coeficientes significativos deverá ser muito menor do que N . Para evitar esta questão de eficiência, pode-se usar uma idéia similar à empregada em métodos pseudo-espectrais, nos quais são utilizadas ambas representações: a física e a em base wavelet.

Outra possibilidade é realizar todas as operações na representação física da solução, utilizando as wavelets apenas para construir e atualizar esta representação. Isto promete baixo custo computacional, uma vez que se está trabalhando com os valores na representação física. Nesta linha têm-se os trabalhos de Jameson [22], [23], que utiliza wavelets para descobrir onde refinar a malha e depois evoluir o operador por diferenças finitas na malha irregular

obtida. Harten [15] também utiliza wavelets como um sensor para tomada de decisão quanto ao modo com que os fluxos numéricos serão calculados em uma malha uniforme.

Em [20], Holmström introduz o conceito de *representação pontual esparsa-RPE* baseado em uma transformada wavelet interpolatória que usa interpolação polinomial em malhas diádicas. A característica fundamental da base wavelet escolhida é a correspondência biunívoca entre valores pontuais e coeficientes wavelets.

Neste trabalho dá-se continuidade ao estudo, iniciado em [20], quanto à utilização de transformadas wavelets interpolatórias associadas a esquemas de diferenças finitas. No Capítulo 1 são apresentados os conceitos fundamentais sobre representação multiresolução a partir de transformadas interpolatórias. Algoritmos para as transformadas direta e inversa são apresentados para dados iniciais de dois tipos: valor pontual ou média da função em malhas diádicas. Apresentam-se ainda dois algoritmos para escolha do estêncil de interpolação dependendo dos valores próximos ao ponto a ser interpolado.

No Capítulo 2 discute-se uma das propriedades mais exploradas de representações multiresolução: a compressão de dados. Apresenta-se a definição do operador de truncamento e sugere-se uma estratégia de compressão. A estratégia sugerida e outras apresentadas na literatura são comparadas através de exemplos. Descreve-se ainda a forma com que se obtém uma representação esparsa a partir da representação multiresolução truncada de uma função dada.

No Capítulo 3 são abordados vários aspectos referentes à resolução adaptativa de equações diferenciais hiperbólicas. Inicialmente discute-se cada um dos passos do algoritmo geral de resolução adaptativa, no qual o operador pode ser resolvido por qualquer esquema de diferenças finitas sobre a malha gerada pela representação pontual esparsa da solução. Neste momento esta representação deverá ser estendida, para possibilitar a evolução temporal da solução.

Propõe-se então uma simplificação no algoritmo para determinação dos coeficientes que farão parte da representação esparsa estendida da solução e ainda propõe-se uma estrutura esparsa para armazenamento de dados que leva em consideração a esparsidade da representação multiresolução truncada.

Através de exemplos numéricos, ilustra-se a importância da extensão da RPE para a evolução da solução. Outro aspecto explorado através das simulações é a questão da atualização da RPE ao longo do tempo. Esta atualização é usualmente feita a cada iteração do algoritmo. No entanto, através dos testes numéricos, percebe-se um ganho significativo em tempo de processamento sem afetar a qualidade da solução, se uma mesma RPE for mantida por alguns passos no tempo antes de ser corrigida.

A partir do estudo feito no Capítulo 3 sobre a associação de esquema de diferenças finitas com malhas adaptativas geradas a partir de transformadas wavelet interpolatórias, propõe-se um método de alta-resolução adaptativo para a resolução de leis de conservação hiperbólicas, cuja extensão a problemas multidimensionais se dá de forma natural.

No Capítulo 4, então, obtém-se uma associação eficiente entre um método de alta resolução, dado por [40], baseado em valores pontuais da função e que dispensa decomposições do operador diferencial em autoespaços, com as transformadas wavelet para valores pontuais.

Uma extensa bateria de testes com problemas clássicos amplamente referenciados na literatura é apresentada, entre eles são resolvidos exemplos para o problema do tubo de choque para a equação de Euler unidimensional.

Apresenta-se também uma extensão natural, com grande potencial de paralelização, para resolução de problemas bidimensionais, tanto escalares, quanto vetoriais, através da fatoração do operador por direção. O problema bidimensional resolvido é o problema de Riemann para a equação de Euler.

Na figura (1) apresenta-se um esquema que ilustra a posicionamento do trabalho em relação às referências da área.

Figura 1: Principais Referências da Área e o Trabalho Apresentado

Capítulo 1

Representação Multiresolução

Neste Capítulo apresentam-se os conceitos básicos, conforme Harten [15], sobre Representação Multiresolução Interpolatória de uma função.

Na Seção 1 as transformações são construídas para serem aplicadas sobre valores pontuais de uma função, definidos em uma malha diádica. Os algoritmos para as transformadas direta e inversa são dados em função de filtros interpolatórios obtidos através de interpolação lagrangeana centrada. Apresentam-se exemplos numéricos para as transformações definidas. Discute-se ainda um tipo de interpolação *essencialmente não oscilatória (ENO)*, na qual o conjunto de pontos que forma o interpolador depende do ponto a ser interpolado.

Na Seção 2, as transformadas são construídas para serem aplicadas a médias da função em subintervalos da malha diádica, denominados células. Novamente são apresentados alguns exemplos de funções e suas respectivas decomposições em diferentes níveis de resolução para transformadas direta e inversa para médias celulares.

Considera-se, ao longo deste Capítulo, que a função é periódica de período 1 no intervalo $[0,1]$.

As Malhas

Seja $X_0 = \{x_j^0\}_{j=0}^{N_0}$, $x_j^0 = jh_0$, $0 \leq j \leq N_0$ uma partição uniforme de $[0,1]$ em $N_0 = 2^{n_0}$ subintervalos de tamanho $h_0 = \frac{1}{N_0} = \frac{1}{2^{n_0}}$, $n_0 \in \mathbb{N}$.

X_0 é considerada a malha com maior número de pontos a partir da qual se obtém uma seqüência com L malhas encaixadas da seguinte forma:

$$X_k = \{x_j^k\}_{j=0}^{N_k}, \quad x_j^k = jh_k, \quad 0 \leq j \leq N_k, \quad k = 1, 2, \dots, L, \quad \text{com } N_k = \frac{N_0}{2^k} = 2^{n_0-k} \quad \text{e } h_k = \frac{1}{N_k} = \frac{2^k}{N_0} = \frac{1}{2^{n_0-k}}.$$

Tem-se, então, $X_L \subset X_{L-1} \subset \dots \subset X_1 \subset X_0$, onde X_L é chamada de malha *mais grossa*, pois contém o menor número de pontos, e X_0 é chamada de malha *mais fina*. $k = 0$ é interpretado como sendo o *nível de resolução mais fino (escala mais fina)* no qual as funções são conhecidas.

Para quaisquer duas malhas consecutivas desta seqüência existem duas relações importantes entre seus pontos:

- $X_{k-1} \cap X_k = X_k = \{x_j^k\}_{j=0}^{N_k}$
- $X_{k-1} \setminus X_k = \{x_{2j-1}^{k-1}\}_{j=1}^{N_k}$,

ou seja, a malha mais grossa, X_k , é obtida a partir da mais fina, X_{k-1} , através da remoção dos pontos com índices ímpares em X_{k-1} .

E entre qualquer elemento da seqüência e a malha *mais fina* inicial, X_0 , vale:

- $X_k \cap X_0 = \{x_j^k\}_{j=0}^{N_k} = \{x_{2^k j}^0\}_{j=0}^{N_k}$

1.1 Representação Multiresolução para Valores Pontuais

Agora consideram-se dados $N_0 = 2^{n_0}$ valores $u^0 = \{u_j^0\}_{j=1}^{N_0}$, que são interpretados como sendo valores pontuais de uma função $u(x)$ correspondente a partição X_0 : $u_j^0 = u(x_j^0)$, $1 \leq j \leq N_0$ e $u_0^0 = u_{N_0}^0$, pela periodicidade de $u(x)$.

Como as demais L malhas da seqüência são todas obtidas a partir de X_0 considerando-se apenas pontos de X_0 da forma $\{x_{2^k j}^0\}_{j=1}^{N_k}$ para cada nível k , obtém-se desta mesma forma seqüências de valores da função $u(x)$ nestas malhas, que satisfazem a seguinte relação (também denominada *relação de escala*):

$$u_j^k = u(x_j^k) = u(x_{2^k j}^0) = u_{2^k j}^0, \quad 0 \leq j \leq N_k \quad (1.1)$$

Seja $I(x, u^k)$ qualquer função interpoladora de $u(x)$ na malha X_k ,

$$I(x_j^k, u^k) = u_j^k, \quad 0 \leq j \leq N_k. \quad (1.2)$$

I é denominada *Preditor*, uma vez que através desta função, é possível obter uma aproximação para valores de u nos pontos $\{x_{2j-1}^{k-1}\} = X_{k-1} \setminus X_k$, ou seja, pontos que completam o nível k para se obter o nível $k-1$.

$$\tilde{u}_{2j-1}^{k-1} = I(x_{2j-1}^{k-1}, u^k), \quad 1 \leq j \leq N_k. \quad (1.3)$$

Define-se também $D^k = D^k(u) = \{d_j^k\}_{j=1}^{N_k}$ como sendo a seqüência dos erros cometidos na interpolação (1.3); os elementos d_j^k , denominados *coeficientes Wavelet* ou *detalhes*, são dados por:

$$d_j^k = u_{2j-1}^{k-1} - \tilde{u}_{2j-1}^{k-1} = u_{2j-1}^{k-1} - I(x_{2j-1}^{k-1}, u^k), \quad 1 \leq j \leq N_k. \quad (1.4)$$

Observa-se que existe uma maneira *bem definida*, de se passar de um nível de resolução para outro:

- Conhecendo-se u^{k-1} , obtém-se a decomposição em um nível (u^k, D^k) :
Os valores da função na malha mais grossa X_k são:

$$u_j^k = u_{2j}^{k-1}, \quad 1 \leq j \leq N_k, \quad (1.5)$$

pela construção apresentada. Sendo D^k obtida pela relação (1.4).

- Conhecendo-se (u^k, D^k) , recupera-se u^{k-1} no nível fino:
Os valores em posições pares da malha mais fina X_{k-1} são:

$$u_{2j}^{k-1} = u_j^k, \quad 1 \leq j \leq N_k. \quad (1.6)$$

E para valores associados às posições ímpares de X_{k-1} , efetua-se a operação inversa à (1.4):

$$u_{2j-1}^{k-1} = \tilde{u}_{2j-1}^{k-1} + d_j^k = I(x_{2j-1}^{k-1}, u^k) + d_j^k, \quad 1 \leq j \leq N_k. \quad (1.7)$$

Portanto existe uma transformação bijetora entre os conjuntos (u^{k-1}) e (u^k, D^k) , ambos com o mesmo número de elementos $N_{k-1} = 2N_k$. Analogamente, existe uma transformação bijetora entre os níveis k e $k+1$ e assim sucessivamente, produzindo a seguinte fatoração de u^0

$$u^0 \leftrightarrow (D^1, u^1) \leftrightarrow (D^1, (D^2, u^2)) \leftrightarrow (D^1, D^2, (D^3, u^3)) \leftrightarrow \dots \leftrightarrow (D^1, D^2, D^3, \dots, D^L, u^L) \quad (1.8)$$

A fatoração $(D^1, D^2, D^3, \dots, D^L, u^L) = (u_M)^T$ é chamada de *representação multiresolução* de u^0 . Esta seqüência de transformações pode ser representada por:

$$u_M = M u^0. \quad (1.9)$$

e como cada etapa é inversível, tem-se:

$$u^0 = M^{-1} u_M. \quad (1.10)$$

A noção da escala $k-1$ está relacionada com a informação em u^{k-1} que não pode ser prevista por u^k seja qual for o preditor considerado, produzindo assim uma fatoração, na qual os *detalhes* são todos significativos.

Observa-se que a relação (1.8) é válida para qualquer técnica de interpolação utilizada em (1.3). O interpolador considerado pode ser sempre o mesmo para todos os pontos da malha e para todos os níveis de resolução [15]; pode variar apenas de um nível para outro [16], ou ainda, pode ser completamente dependente dos dados [19],[1], o que transforma M em um operador altamente não linear.

A mesma estrutura para o operador M é encontrada para transformações cujos preditor (1.3) e detalhes (1.4) são construídos de diferentes formas. Em [32], Nielsen apresenta as

expressões de cada uma das transformações intermediárias que compõe $\mathbf{M}$ e os algoritmos para a Transformada Rápida no contexto de Wavelets Ortonormais de Daubechies.

Neste trabalho não serão abordados aspectos funcionais relacionados ao operador $\mathbf{M}$, desta forma não serão construídas as funções escala e wavelets interpolatórias associadas [44], [24]. No entanto, apresenta-se a seguir a formulação discreta para $\mathbf{M}$, denominada *Transformada Wavelet Interpolatória*, que depende do tipo de interpolador utilizado na construção do preditor I , dado pela expressão (1.3). Serão apresentados também os algoritmos para construção das transformadas direta e inversa de forma rápida.

1.1.1 Interpolação Lagrangeana Centrada

Uma das preocupações principais na construção de preditores em um esquema multi-resolução é a “qualidade” da predição. Quando se fixa um preditor particular, os erros cometidos por este, e consequentemente, os coeficientes wavelet, d_j^k , incluem uma componente intrínseca à escala e uma componente do erro de aproximação que está relacionado com a precisão do preditor utilizado.

A maneira mais simples de se construir o preditor (1.3) é utilizando interpolação Lagrangeana centrada [9] e [12]. O valor de $u(x)$ no ponto x_{2j-1}^{k-1} é aproximado pelo polinômio de grau $r - 1$, com $r = 2s$ fixado, que interpola $u_{j-s}^k, \dots, u_{j+s-1}^k$.

O conjunto de r pontos utilizados para construir o interpolador polinomial forma o estêncil, S_j , do interpolador. Como os pontos do estêncil são igualmente espaçados e ainda se distribuem de forma homogênea ao redor de x_{2j-1}^{k-1} , isto é, existe a mesma quantidade de pontos tanto à direita quanto à esquerda de x_{2j-1}^{k-1} , tem-se a seguinte expressão:

$$\tilde{u}_{2j-1}^{k-1} = I(x_{2j-1}^{k-1}, u^k) = \sum_{l=1}^s \beta_l (u_{j+l-1}^k + u_{j-l}^k); \quad r = 2s, \quad (1.11)$$

sendo que os coeficientes β_l , que dependem de r , são dados por:

$$\begin{cases} r = 2 & \Rightarrow \beta_1 = 1/2 \\ r = 4 & \Rightarrow \beta_1 = 9/16, \beta_2 = -1/16 \\ r = 6 & \Rightarrow \beta_1 = 150/256, \beta_2 = -25/256, \beta_3 = 3/256 \end{cases} \quad (1.12)$$

Observa-se que os coeficientes β_l não dependem do nível de resolução k .

No caso da função u ser periódica, os coeficientes β_l também não dependem da posição x_{2j-1}^{k-1} . Caso contrário, β_l dependerá se os pontos do estêncil estão ou não todos contidos em X_k .

Para pontos onde $S_j \cap X_k = S_j$, ou seja, todos os pontos do estêncil são da malha, os coeficientes β_l são dados por (1.12). Já nos pontos próximos às fronteiras, para os quais $S_j \not\subset X_k$, os coeficientes β_l deverão ser calculados de outro modo.

Uma forma de se construir o interpolador próximo das fronteiras é considerá-lo não mais centrado [12]. Com isso os coeficientes β_l próximos da fronteira esquerda são dados por:

$$\left\{ \begin{array}{l} r = 2 \Rightarrow \beta_1 = 1/2 \\ r = 4 \Rightarrow \beta_1 = 5/16, \quad \beta_2 = 15/16, \quad \beta_3 = -5/16, \quad \beta_4 = 1/16 \\ r = 6 \Rightarrow \beta_1 = 63/256, \quad \beta_2 = 315/256, \quad \beta_3 = -105/256, \quad \beta_4 = 63/128, \\ \quad \beta_5 = -45/256, \quad \beta_6 = 7/256 \\ r = 6 \Rightarrow \beta_1 = -7/256, \quad \beta_2 = 105/256, \quad \beta_3 = 105/128, \quad \beta_4 = -35/128, \\ \quad \beta_5 = 21/256, \quad \beta_6 = -3/256 \end{array} \right. \quad (1.13)$$

Note que para $r = 6$ existem dois conjuntos de coeficientes β_l . Isso ocorre, pois existem dois pontos próximos da fronteira esquerda que precisam ser calculados por interpoladores não centrados. Quanto maior o número de pontos utilizados no interpolador, maior será o número de posições próximas às fronteiras que deverão ser calculadas separadamente.

Os coeficientes β_l para a fronteira direita são os mesmos dados em (1.13) apenas em ordem invertida.

Desta forma, $\mathbf{M}$ é um operador linear que pode ser representado por uma matriz $N_0 \times N_0$, mas mesmo sem escrever $\mathbf{M}$ explicitamente, pelas relações (1.5), (1.6) e (1.7) têm-se os seguintes algoritmos rápidos para as duas transformadas de u : Direta e Inversa.

Algoritmo: Transformada Direta $u_M = \mathbf{M}u_0$

$$\begin{array}{l} \text{Para } k = 1, 2, \dots, L \\ \left[\begin{array}{l} u_j^k = u_{2j}^{k-1}, \quad 1 \leq j \leq N_k. \\ d_j^k = u_{2j-1}^{k-1} - \sum_{l=1}^s \beta_l (u_{j+l-1}^k + u_{j-l}^k), \quad 1 \leq j \leq N_k. \end{array} \right. \end{array} \quad (1.14)$$

Algoritmo: Transformada Inversa $u_0 = \mathbf{M}^{-1}u_M$

$$\begin{array}{l} \text{Para } k = L, L-1, \dots, 1 \\ \left[\begin{array}{l} u_{2j}^{k-1} = u_j^k, \quad 1 \leq j \leq N_k. \\ u_{2j-1}^{k-1} = \sum_{l=1}^s \beta_l (u_{j+l-1}^k + u_{j-l}^k) + d_j^k, \quad 1 \leq j \leq N_k. \end{array} \right. \end{array} \quad (1.15)$$

Decaimento dos Coeficientes Wavelets

Quando a função inicialmente dada $u(x)$ for suave, então a qualidade do preditor está diretamente relacionada ao grau do polinômio interpolador utilizado:

$$I(x, u^k) = u(x) + O((h_k)^{r-1}), \quad (r-1) = \partial I \quad (1.16)$$

Como consequência, obtém-se uma estimativa para o comportamento dos coeficientes wavelets em cada nível de resolução:

$$|d_j^k| = |u(x_j^k) - I(x_j^k, u^k)| \leq c(h_k)^{r-1} = c \frac{1}{2^{k(r-1)}}, \quad (1.17)$$

com c constante independente do nível k

No caso da função não ser suave, mas possuir $p - 1$ derivadas contínuas e uma descontinuidade tipo salto na p -ésima derivada em $x = x_d$, Harten [15] apresenta a seguinte estimativa:

$$\begin{aligned} |u(x) - I(x, u^k)| &\sim O((h_k)^s), \quad \text{se } s \leq p - 1, \quad s = \partial I, \\ |u(x) - I(x, u^k)| &\sim (h_k)^p [u^{(p)}], \quad \text{se } s \geq p, \quad s = \partial I, \end{aligned} \quad (1.18)$$

ou seja, para todo x_j^k próximo de x_d , vale:

$$\begin{aligned} |d_j^k| &= |u(x_j^k) - I(x_j^k, u^k)| \sim O((h_k)^s), \quad \text{se } s \leq p - 1, \quad s = \partial I, \\ |d_j^k| &= |u(x_j^k) - I(x_j^k, u^k)| \sim (h_k)^p [u^{(p)}], \quad \text{se } s \geq p, \quad s = \partial I, \end{aligned} \quad (1.19)$$

com $[u^{(p)}]$ denotando o salto na p -ésima derivada.

Essas estimativas mostram que longe das descontinuidades os coeficientes wavelets d_j^k diminuem em módulo quando se passa para escalas mais finas com uma taxa que é determinada pela regularidade local da função e pela ordem de precisão do interpolador. Já na vizinhança da singularidade, os coeficientes d_j^k permanecem do mesmo tamanho, independente da escala.

Variando o Grau do Interpolador por Nível de Resolução

Outra possibilidade de se construir um operador $\mathbf{M}$ linear é considerar interpoladores diferentes para cada nível de resolução k [16]. Os interpoladores tanto podem ter o grau diferente a cada nível quanto podem ser construídos por técnicas de interpolação diferentes, desde que para todos os pontos de um mesmo nível, seja utilizado o mesmo interpolador, isto é, ter o mesmo grau e usar sempre um estêncil com a mesma quantidade de pontos e a mesma posição em relação ao ponto de interpolação. Assim denota-se I com o subíndice k para expressar a dependência apenas com o nível de resolução.

$$\begin{aligned} I_k(x_j^k, u^k) &= u_j^k, \quad 0 \leq j \leq N_k. \\ \tilde{u}_{2j-1}^{k-1} &= I_k(x_{2j-1}^{k-1}, u^k), \quad 1 \leq j \leq N_k. \end{aligned} \quad (1.20)$$

Com isso, denota-se o operador por $\mathbf{M}_k$ e os algoritmos para Transformada Direta e Inversa têm a seguinte forma:

Algoritmo: Transformada Direta $u_M = \mathbf{M}_k u_0$

$$\begin{aligned} \text{Para } k &= 1, 2, \dots, L \\ u_j^k &= u_{2j}^{k-1}, & 1 \leq j \leq N_k. \\ d_j^k &= u_{2j-1}^{k-1} - I_k(x_{2j-1}^{k-1}, u^k), & 1 \leq j \leq N_k. \end{aligned} \quad (1.21)$$

Algoritmo: Transformada Inversa $u_0 = M_k^{-1}u_M$

$$\begin{aligned} \text{Para } k = L, L-1, \dots, 1 \\ u_{2j}^{k-1} &= u_j^k, & 1 \leq j \leq N_k. \\ u_{2j-1}^{k-1} &= I_k(x_{2j-1}^{k-1}, u^k) + d_j^k, & 1 \leq j \leq N_k. \end{aligned} \quad (1.22)$$

A seguir são apresentados exemplos numéricos para as transformadas direta e inversa para valores pontuais. São escolhidos três exemplos de funções: uma suave, outra descontínua e a terceira com descontinuidade na derivada.

1.1.2 Parte Computacional

Nesta Seção serão apresentados alguns exemplos numéricos para mostrar os resultados dos algoritmos para Transformada Direta e Inversa e o comportamento dos detalhes para diferentes tipos de funções.

As funções escolhidas são:

$$\begin{aligned} v(x) &= \sin(2\pi x), \quad x \in [0, 1] \\ u(x) &= \begin{cases} 1, & 0.25 \leq x < 0.75 \\ 0, & 0 \leq x < 0.25, \text{ e } 0.75 \leq x \leq 1 \end{cases} \\ w(x) &= |\sin(\frac{3}{2}\pi(x - \frac{1}{2}))|, \quad x \in [0, 1] \end{aligned} \quad (1.23)$$

O interpolador utilizado será de grau 5. Os gráficos abaixo mostram os 4 níveis de resolução e os detalhes associados a cada nível. Como a transformação envolve valores pontuais da função, a relação de escala (1.1) é observada diretamente na figura (1.2) abaixo.

Figura 1.1: Valores Pontuais de $v(x)$ no nível mais refinado, $n_0 = 7$. $N_0 = 2^{n_0} = 132$ pontos)

Figura 1.2: Valores Pontuais de $v(x) \times [0, 1]$ nos quatro níveis de faturação

Os detalhes em cada um dos quatro níveis de resolução estão na figura (1.3) a seguir:

Novamente são apresentados os valores máximos dos módulos dos detalhes por nível de resolução, que seguem a estimativa (1.17) para funções suaves.

nível k	$\max_j d_j^k $
1	4.3600545396316D-09
2	2.7678132519249D-07
3	1.7144954394310D-05
4	9.6145710207263D-04

Tabela 1.1: Valor máximo dos detalhes de $v(x)$ por nível

Caso seja utilizado um preditor de grau 1, o máximo dos módulos dos detalhes no nível mais grosseiro ($k = L = 4$) é 7.0326141918013D-02 e para um preditor de grau 3, 7.8261419180130D-03.

Figura 1.3: Detalhes de $v(x) \times [0, 1]$ nos diferentes níveis

A próxima função a ser decomposta é a função $u(x)$ que apresenta descontinuidades tipo salto.

Figura 1.4: Valores Pontuais de $u(x) \times [0, 1]$ no nível mais refinado, $k=0$, com 2^7 pontos.

Figura 1.5: Valores Pontuais de $u(x) \times [0, 1]$ em quatro níveis diferentes

Os detalhes em cada um dos quatro níveis de resolução estão na figura (1.6) a seguir:

Figura 1.6: Detalhes da função em cada um dos níveis da fatoração

Novamente são apresentados os valores máximos dos módulos dos detalhes por nível de resolução na tabela (1.2), ilustrando o resultado teórico já apresentado na estimativa (1.19).

nível k	$\max_j d_j^k $
1	0.5000000000000000
2	0.5000000000000000
3	0.5000000000000000
4	0.5000000000000000

Tabela 1.2: Módulo máximo dos detalhes por nível

O último exemplo para a transformação envolvendo interpolação de grau 5 para valores pontuais é a função $w(x)$. O nível de resolução mais refinado contém 2^{n_0} pontos, com $n_0 = 7$. São realizados quatro níveis de transformação, analogamente ao realizado para os demais exemplos.

Figura 1.7: Valores Pontuais da função $w(x) \times [0, 1]$ no nível mais refinado

Figura 1.8: Valores Pontuais da função $w(x) \times [0, 1]$ nos níveis $k = 1$ a $k = 4$

Os detalhes em cada um dos quatro níveis de resolução são apresentados na figura (1.9).

Figura 1.9: Detalhes da função nos níveis $k = 1, 2, 3, 4$

E os valores máximos dos módulos dos detalhes por nível de resolução estão na tabela (1.3).

nível k	$\max_j d_j^k $
1	1.0929083361628D-02
2	2.1854752319930D-02
3	4.3675510039484D-02
4	8.6912469233323D-02

Tabela 1.3: Módulo máximo dos detalhes por nível

Tanto para valores pontuais quanto para médias celulares, os algoritmos de reconstrução são *exatos*, pois recuperam exatamente os valores iniciais ou as médias iniciais da função a partir da sequência de detalhes e do sinal no nível mais grosseiro.

1.1.3 Interpolação ENO

As duas formas anteriores de construção de preditores não levam em consideração características locais da função a ser interpolada. Uma vez escolhidos o grau do interpolador e a forma do estêncil, estes permanecem fixados para todos os pontos da malha.

No caso de funções que apresentam singularidades isoladas, a precisão da interpolação nos pontos da região em torno da singularidade depende do estêncil de pontos utilizado para construir o preditor polinomial. Toda vez que o estêncil cruzar a descontinuidade, (1.16) deixa de valer; e uma vez que o número de pontos do estêncil aumenta com o grau do polinômio local, o mesmo ocorre com a extensão da região de baixa precisão.

A idéia básica da técnica de interpolação ENO, [1],[19],[11] é aumentar a região de alta precisão, construindo interpoladores que, sempre que possível, considerem apenas informações das regiões de suavidade da função a ser interpolada.

Seja $I(x, u^k)$ função polinomial por partes em x que interpola u^k nos pontos $\{x_j^k\}$ da malha X_k , isto é:

$$\begin{aligned} I(x_j^k, u^k) &= u_j^k = u(x_j^k), \\ I(x, u^k) &= q_m(x, u^k), \quad x_j^k \leq x \leq x_{j+1}^k \end{aligned} \quad (1.24)$$

q_m é o único polinômio em x de grau m que interpola u^k em $(m+1)$ pontos sucessivos $\{x_i^k\}$, $i_m(j) \leq i \leq i_m(j) + m$ que incluem x_j^k e x_{j+1}^k . Aqui $i_m(j)$ é o primeiro ponto do *estêncil*.

Existem m polinômios possíveis correspondentes às m escolhas diferentes de $i_m(j)$ sujeitas à restrição $j-m+1 \leq i_m(j) \leq j$, como mostra a figura 1.10 para o caso particular de $m=3$.

Figura 1.10: Possibilidades de estêncil contendo x_j e x_{j+1} . $i_m(j)$ indica o índice do primeiro ponto do estêncil

Associa-se a $[x_j^k, x_{j+1}^k]$ um *estêncil* com $(m+1)$ pontos tal que $u(x)$ seja “mais suave” em $[x_{i_m(j)}^k, x_{i_m(j)+m}^k]$ em algum sentido assintótico. A informação sobre suavidade é extraída da tábua de diferenças divididas de $u(x)$.

Definem-se de forma recursiva *diferenças divididas* de uma função qualquer $w(x)$ por:

$$\begin{aligned} w[x_i] &= w(x_i) \\ w[x_i, \dots, x_{i+k}] &= \frac{w[x_{i+1}, \dots, x_{i+k}] - w[x_i, \dots, x_{i+k-1}]}{(x_{i+k} - x_i)} \end{aligned} \quad (1.25)$$

Sabe-se [42] que: Se uma função $w(x)$ for C_∞ em $[x_i, x_{i+k}]$, então

$$w[x_i, \dots, x_{i+k}] = \frac{1}{k!} \frac{d^k}{dx^k} w(\xi_{i,k}), x_i \leq \xi_{i,k} \leq x_{i+k}. \quad (1.26)$$

No entanto, se $w(x)$ tiver uma descontinuidade do tipo salto na p -ésima derivada, $0 \leq p \leq k$, neste intervalo, então:

$$w[x_i, \dots, x_{i+k}] = O(h^{-k+p})[w^{(p)}], \quad (1.27)$$

sendo que $[w^{(p)}]$ denota o salto na p -ésima derivada.

Equações (1.26) e (1.27) mostram que $|w[x_i, \dots, x_{i+k}]|$ fornece uma medida assintótica da suavidade de $w(x)$ em $[x_i, x_{i+k}]$, conforme consta em [11];

Se $w(x)$ for suave em $[x_{i_1}, x_{i_1+k}]$, mas for descontínua em $[x_{i_2}, x_{i_2+k}]$, então para h suficientemente pequeno: $|w[x_{i_1}, \dots, x_{i_1+k}]| < |w[x_{i_2}, \dots, x_{i_2+k}]|$.

Assim o problema de escolher um estêncil de pontos para o qual a função $u(x)$ é **mais suave** é basicamente o mesmo de encontrar um intervalo no qual $u(x)$ tem **a menor diferença dividida**. A característica essencial da interpolação ENO é fornecer um procedimento para determinação deste intervalo.

Procedimento para encontrar $i_m(j)$

1. $i_1(j) = j$, isto é, q_1 é o polinômio que interpola u^k em x_j^k e x_{j+1}^k .
2. Assume-se já definido q_n , polinômio de grau n que interpola u^k em $x_{i_n(j)}^k, \dots, x_{i_n(j)+n}^k$.
3. Consideram-se agora os dois candidatos a q_{n+1} , os dois polinômios de grau $(n+1)$ obtidos quando se agrega um ponto ao estêncil já existente à esquerda ou à direita; o que corresponde a colocar $i_{n+1}(j) = i_n(j) - 1$, se o ponto for agregado à esquerda ou $i_{n+1}(j) = i_n(j)$, se for agregado à direita. Escolhe-se aquele *estêncil* que forneça a diferença dividida com menor módulo.

Este procedimento pode ser escrito ainda como segue:

Algoritmo 1: Busca Hierárquica do Estêncil

```

 $i_0(j) = j$ 
se  $|w[x_{j-1}, x_j, x_{j+1}]| < |w[x_j, x_{j+1}, x_{j+2}]|$ , então
     $i_1(j) = i_0(j) - 1$ 
se não,
     $i_1(j) = i_0(j)$ 
para  $l = 1, m - 2$ 
    se  $|w[x_{i_l(j)-1}, x_{i_l(j)}, \dots, x_{i_l(j)+l+1}]| < |w[x_{i_l(j)}, x_{i_l(j)+1}, \dots, x_{i_l(j)+l+2}]|$ , então
         $i_{l+1}(j) = i_l(j) - 1$ 
    se não,
         $i_{l+1}(j) = i_l(j)$ 
 $i(j) = i_{m-1}(j)$ 

```

Lembrando que m é o grau do polinômio interpolador e j está relacionado ao índice do ponto x_{2j-1}^{k-1} a ser interpolado.

Existe uma segunda possibilidade de se procurar o intervalo de menor diferença dividida, só que de maneira não hierárquica.

Algoritmo 2: Busca Não - Hierárquica do Estêncil

$$i(j) \text{ tal que } |w[x_{i(j)}, \dots, x_{i(j)+m}]| = \min_{j-m+1 \leq l \leq j} |w[x_l, \dots, x_{l+m}]|$$

Após a determinação do primeiro ponto do estêncil, $i(j)$, o polinômio ENO-interpolador tem a seguinte forma, considerando o grau $m = 3$, por exemplo:

$$q_j^{(3)}(x) = \frac{u[x_{i(j)}] + u[x_{i(j)}, x_{i(j)+1}](x - x_{i(j)}) + u[x_{i(j)}, x_{i(j)+1}, x_{i(j)+2}](x - x_{i(j)})(x - x_{i(j)+1}) + u[x_{i(j)}, x_{i(j)+1}, x_{i(j)+2}, x_{i(j)+3}](x - x_{i(j)})(x - x_{i(j)+1})(x - x_{i(j)+2})}{1}$$

Em [11], Donat mostra as vantagens de se usar **Busca Hierárquica** para $i(j)$ e ainda que a interpolação ENO mantém a precisão alta sobre a maior região possível.

1.1.4 Interpolação ENO para Valores Pontuais

O exemplo a seguir ilustra a diferença entre os resultados obtidos por interpoladores de mesmo grau, mas que são construídos de formas diferentes, quando se interpola uma função não suave. No caso da interpolação ENO, o estêncil para construir o interpolador depende dos dados a serem interpolados, caracterizando uma transformação não linear. Já para o interpolador de Lagrange centrado, o estêncil é fixo para todos os pontos da malha.

O gráfico da esquerda da figura (1.11) mostra a função pulso exata e a função obtida por interpolação ENO. O gráfico da direita mostra a função exata e a função obtida por interpolação lagrangeana centrada.

Figura 1.11: Interpolações ENO e Lagrangeana Centrada de grau 3

Agora passa-se a estudar representações multiresolução para dados não mais pontuais e sim para médias celulares de uma função inicialmente dada. As construções a seguir envolvem as deduções feitas para valores pontuais.

1.2 Representação Multiresolução para Médias Celulares

Considera-se agora uma seqüência $\bar{u}^0 = \{\bar{u}_j^0\}_{j=1}^{N_0}$ com N_0 valores interpretados aqui como *médias celulares* de alguma função $u(x)$.

$$\bar{u}_j^0 = \frac{1}{h_0} \int_{x_{j-1}^0}^{x_j^0} u(x) dx, \quad 1 \leq j \leq N_0. \quad (1.28)$$

Considerando a seqüência de malhas encaixadas $\{X_k\}_{k=1}^L$, definem-se os valores

$$\bar{u}_j^k = \frac{1}{h_k} \int_{x_{j-1}^k}^{x_j^k} u(x) dx, \quad 1 \leq j \leq N_k. \quad (1.29)$$

Assim, para se relacionar médias dadas em duas escalas consecutivas, utilizam-se a definição (1.28) e a relação entre as malhas ($h_k = 2h_{k-1}$), obtendo-se a seguinte relação:

$$\bar{u}_j^k = \frac{1}{h_k} \int_{x_{j-1}^k}^{x_j^k} u(x) dx = \frac{1}{2h_{k-1}} \left[\int_{x_{2j-2}^{k-1}}^{x_{2j-1}^{k-1}} u(x) dx + \int_{x_{2j-1}^{k-1}}^{x_{2j}^{k-1}} u(x) dx \right], \quad (1.30)$$

que pode ser escrita como:

$$\bar{u}_j^k = \frac{1}{2}(\bar{u}_{2j-1}^{k-1} + \bar{u}_{2j}^{k-1}). \quad (1.31)$$

No contexto de médias celulares, (1.31) é a *relação de escala*, e através dela, os valores $\{\bar{u}_j^k\}_{j=1}^{N_k}$, $1 \leq k \leq L$, podem ser calculados diretamente a partir dos dados $\{\bar{u}_j^0\}_{j=1}^{N_0}$ fornecidos na malha mais fina, sem a necessidade de se conhecer a função $u(x)$ explicitamente; como mostra o procedimento abaixo:

Algoritmo

$$\begin{aligned} &\text{Para } k = 1, \dots, L \\ &\quad \text{Para } j = 1, \dots, N_k \\ &\quad \quad \bar{u}_j^k = \frac{1}{2}(\bar{u}_{2j-1}^{k-1} + \bar{u}_{2j}^{k-1}) \end{aligned}$$

Existe, portanto, uma transformação entre o nível mais fino e qualquer outro mais grosso. Mas para que se tenha uma representação multiresolução é preciso também que se possa estimar os valores em escalas mais finas através dos valores em escalas mais grossas. Com esse objetivo, consideram-se:

$$U(x) = \int_0^x u(y) dy, \quad (1.32)$$

a função primitiva de $u(x)$, e a seqüência de valores pontuais:

$$U_j^k = U(x_j^k), \quad x_j^k \in X_k. \quad (1.33)$$

Conhecer as médias celulares $\bar{u}^k$ ou conhecer os valores pontuais de sua primitiva é equivalente, pois valem as seguintes relações para $0 \leq k \leq L$, $1 \leq j \leq N_k$:

$$(\Rightarrow) \quad U_j^k = U(x_j^k) = h_k \sum_{i=1}^j \bar{u}_i^k \quad (1.34)$$

$$(\Leftarrow) \quad \bar{u}_j^k = \frac{[U(x_j^k) - U(x_{j-1}^k)]}{h_k} = \frac{[U_j^k - U_{j-1}^k]}{h_k} \quad (1.35)$$

Então, para se produzir aproximações para as médias $\bar{u}_j^{k-1}$ na malha mais fina, X_{k-1} , a partir das médias celulares na malha X_k , são necessários os seguintes passos:

Algoritmo

1. Calcula-se $\{U_j^k\}_{j=1}^{N_k}$ a partir das médias celulares $\{\bar{u}_j^k\}_{j=1}^{N_k}$ através de (1.34).
2. Interpola-se $\{U_j^k\}_{j=1}^{N_k}$ nos pontos $\{x_{2j-1}^{k-1}\}$ por alguma técnica apresentada na sessão anterior para valores pontuais, obtendo-se uma aproximação para a primitiva de $u(x)$ em X_{k-1} ,

$$\tilde{U}_{2j-1}^{k-1} = I(x_{2j-1}^{k-1}, U^k). \quad (1.36)$$

3. Traduz-se a relação (1.35) para a escala $(k-1)$, utilizando-se os valores aproximados dados em (1.36), produzindo desta forma, aproximações para as médias celulares em X_{k-1} :

$$\tilde{u}_{2j}^{k-1} = \frac{U_{2j}^{k-1} - \tilde{U}_{2j-1}^{k-1}}{h_{k-1}}, \quad (1.37)$$

$$\tilde{u}_{2j-1}^{k-1} = \frac{\tilde{U}_{2j-1}^{k-1} - U_{2j-2}^{k-1}}{h_{k-1}}, \quad (1.38)$$

Como os valores da primitiva são pontuais, por (1.1), têm-se $U_{2j}^{k-1} = U_j^k$ e $U_{2j-2}^{k-1} = U_{j-1}^k$ e as expressões (1.37) e (1.38) são reescritas como:

$$\tilde{u}_{2j}^{k-1} = \frac{U_j^k - \tilde{U}_{2j-1}^{k-1}}{h_{k-1}}, \quad (1.39)$$

$$\tilde{u}_{2j-1}^{k-1} = \frac{\tilde{U}_{2j-1}^{k-1} - U_{j-1}^k}{h_{k-1}}. \quad (1.40)$$

Verifica-se ainda que a relação de escala (1.31) permanece válida para as aproximações (1.39) e (1.40), pois:

$$\frac{1}{2}(\tilde{u}_{2j}^{k-1} + \tilde{u}_{2j-1}^{k-1}) = \frac{1}{2h_{k-1}}(U_j^k - \tilde{U}_{2j-1}^{k-1} + \tilde{U}_{2j-1}^{k-1} - U_{j-1}^k) = \frac{1}{h_k}(U_j^k - U_{j-1}^k) = \bar{u}_j^k. \quad (1.41)$$

E com isso, tem-se uma outra alternativa para se obter as médias em X_{k-1} :

- $\tilde{u}_{2j-1}^{k-1}$ continua sendo obtida por (1.38) e
- $\tilde{u}_{2j}^{k-1}$ é determinada, então, pela relação de escala (1.41), ou seja, $\tilde{u}_{2j}^{k-1} = 2\bar{u}_j^k - \tilde{u}_{2j-1}^{k-1}$.

A seguir, denota-se por $d^k = d^k(u^0) = \{d_j^k\}_{j=1}^{N_k}$ a seqüência dos erros de aproximação na previsão dos valores $\{\bar{u}_{2j-1}^k\}_{j=1}^{N_k}$ a partir das médias celulares $\bar{u}^k$ no nível mais grosso.

$$d_j^k = \bar{u}_{2j-1}^{k-1} - \tilde{u}_{2j-1}^{k-1} = \bar{u}_{2j-1}^{k-1} - \left[\frac{I(x_{2j-1}^{k-1}) - U_{j-1}^k}{h_{k-1}} \right], \quad 1 \leq j \leq N_k. \quad (1.42)$$

Assim como existe uma equivalência entre a representação multiresolução para valores pontuais e seu nível mais fino inicial, obtém-se também uma equivalência no contexto de médias celulares, uma vez que conhecer $(d^k, \bar{u}^k)$ equivale a conhecer $\bar{u}^{k-1}$:

$$(\Rightarrow) \quad \bar{u}^{k-1} \rightarrow (d^k, \bar{u}^k)$$

- Pela relação de escala (1.31), tem-se $\bar{u}_j^k = \frac{1}{2}(\bar{u}_{2j-1}^{k-1} + \bar{u}_{2j}^{k-1})$.
- d_j^k é obtido pela definição (1.42) acima.

$$(\Leftarrow) \quad (d^k, \bar{u}^k) \rightarrow \bar{u}^{k-1}$$

- $\bar{u}_{2j-1}^{k-1} = d_j^k + \left[\frac{I(x_{2j-1}^{k-1}) - U_{j-1}^k}{h_{k-1}} \right]$
- $\bar{u}_{2j}^{k-1} = 2\bar{u}_j^k - \bar{u}_{2j-1}^{k-1}$

De forma análoga ao desenvolvimento para valores pontuais, obtém-se uma transformação bijetora entre $\bar{u}^0$ e sua representação multiresolução $\bar{u}_M = (d^1, d^2, \dots, d^L, \bar{u}^L)^T$, representada pelo operador $\bar{M}$.

$$\bar{u}_M = \bar{M}\bar{u}_0, \quad \bar{u}_0 = \bar{M}^{-1}\bar{u}_M \quad (1.43)$$

Outra equivalência interessante é entre as representações multiresolução $\bar{u}_M$ e U_M . Uma vez que por (1.34) e (1.35), $\bar{u}^0 \leftrightarrow U^0$, segue imediatamente que $\bar{u}^L \leftrightarrow U^L$. E, por (1.42), (1.35) e a relação de escala para valores pontuais, tem-se:

$$\begin{aligned} d_j^k(\bar{u}^0) &= \frac{1}{h_{k-1}} [U_{2j-1}^{k-1} - U_{2j-2}^{k-1}] - \frac{1}{h_{k-1}} [I(x_{2j-1}^{k-1}, U^k) - U_{2j-2}^{k-1}] = \\ &= \frac{1}{h_{k-1}} [U_{2j-1}^{k-1} - I(x_{2j-1}^{k-1}, U^k)] = \frac{1}{h_{k-1}} D_j^k(U^0), \end{aligned} \quad (1.44)$$

caracterizando a transformação entre $d_j^k(\bar{u}^0)$ e $D_j^k(U^0)$. Logo,

$$(d^1, d^2, \dots, d^L, \bar{u}^L)^t \Leftrightarrow \bar{u}_M \Leftrightarrow U_M \Leftrightarrow (D^1, D^2, \dots, D^L, U^L)^t$$

Passa-se agora a apresentar os algoritmos rápidos para as transformadas direta e inversa para médias celulares, dependendo do tipo de interpolador utilizado para prever os valores de função primitiva de $u(x)$.

1.2.1 Interpolação Lagrangeana Centrada

Quando se utiliza interpolação Lagrangeana centrada (1.11) em (1.36), conseqüentemente obtém-se $\bar{M}$ como um operador linear que pode ser expresso por uma matriz $N_0 \times N_0$ e as transformações em (1.43) podem ser obtidas pelos seguintes algoritmos:

Algoritmo: Transformada Direta $u_M = \overline{M}u^0$

$$\begin{aligned} \text{Para } k = 1, 2, \dots, L \\ \left[\begin{aligned} \overline{u}_j^k &= \frac{1}{2}(\overline{u}_{2j-1}^{k-1} + \overline{u}_{2j}^{k-1}), \quad 1 \leq j \leq N_k. \\ d_j^k &= \overline{u}_{2j-1}^{k-1} - \overline{u}_j^k - \sum_{l=1}^{s-1} \gamma_l(\overline{u}_{j+l}^k - \overline{u}_{j-l}^k), \quad 1 \leq j \leq N_k; \end{aligned} \right. \end{aligned} \quad (1.45)$$

sendo que os coeficientes γ_l dependem de escolha feita para o grau do polinômio interpolador centrado $I(x, U^k)$, e portanto dependem dos coeficientes β_l dados em (1.12). No Apêndice (A) mostra-se o cálculo de γ_l para o caso particular $\partial I = 3 = \text{grau de } I(x, U^k)$ e apresentam-se os valores de γ_l para $\partial I = \text{grau de } I(x, U^k) = 5$. A precisão do predictor neste caso é dada por $\bar{r} = 2s - 1 = r - 1$, com r sendo o grau do interpolador da primitiva.

O algoritmo para Transformada Inversa das médias celulares é então dado por:

Algoritmo: Transformada Inversa $u_0 = \overline{M}^{-1}u_M$

$$\begin{aligned} \text{Para } k = L, L-1, \dots, 1 \\ \left[\begin{aligned} \Delta &= \sum_{l=1}^{s-1} \gamma_l(\overline{u}_{j+l}^k - \overline{u}_{j-l}^k) + d_j^k, \quad 1 \leq j \leq N_k. \\ \overline{u}_{2j-1}^{k-1} &= \overline{u}_j^k + \Delta, \quad \overline{u}_{2j}^{k-1} = \overline{u}_j^k - \Delta, \quad 1 \leq j \leq N_k. \end{aligned} \right. \end{aligned} \quad (1.46)$$

Usando as estimativas (1.16) e (1.18), o fato de que a primitiva $U(x)$ possui uma derivada contínua a mais do que $u(x)$ e utilizando a relação (1.44), Harten, [15], apresenta a seguinte estimativa para o comportamento dos coeficientes wavelet no caso de $u(x)$ possuir $p - 1$ derivadas contínuas e uma descontinuidade tipo salto na p -ésima derivada em $x = x_d$:

$$d_j^k(\overline{u}^0) \sim \begin{cases} (h_k)^{\bar{r}} u^{(\bar{r})}, & \text{se } \bar{r} \leq p \\ (h_k)^p [u^{(p)}], & \text{se } 0 \leq p \leq \bar{r} \end{cases} \quad (1.47)$$

com $[u^{(p)}]$ denotando o salto na p -ésima derivada.

1.2.2 Interpolação ENO com Resolução Intra-Célula(Subcell)

Outra possibilidade é a utilização da interpolação ENO na construção do interpolador da primitiva de $u(x)$, (1.36).

No caso da função $U(x)$ ser contínua, suave por partes e $U'(x)$ com uma descontinuidade tipo salto em $x_d \in [x_{j-1}, x_j]$, isto é, $[U']_{x_d} = U'(x_d + 0) - U'(x_d - 0) = O(1)$, Donat [11] analisou os dois procedimentos de escolha do estêncil, S_j , e observou que ambos algoritmos 1 e 2 levam a escolhas tais que $S_{j-1} \cap S_j = \emptyset$, ou seja, ambos algoritmos geram polinômios interpolatórios que satisfazem: $U(x) = q_l(x, U^k) + O(h^r ||U^r||)$, $x \in [x_{l-1}, x_l]$, $l \leq j-1$, $l \geq j+1$.

No entanto, a parte polinomial suave $q_j(x, U^k)$ só pode ser de primeira ordem na célula $[x_{j-1}, x_j]$ que contém a singularidade.

Caso a localização da descontinuidade x_d seja conhecida dentro da célula, ou uma aproximação para ela, a definição do interpolador $I(x, U^k)$ na célula que contém a descontinuidade pode ser modificada para manter a relação $U(x) = I(x, U^k) + O(h^r)$ válida sobre uma região ainda maior. Esta é a idéia central do esquema **ENO com Resolução Intra-Célula**, [1], sendo que Harten et al. denominam esta técnica de **Subcell Resolution**.

Aproximação para a posição x_d

Pelo esquema ENO, (1.24), obtém-se interpoladores tais que:

$$\begin{aligned} U(x) &= q_{j-1}(x, U^k) + O(h^r), & x \in [x_{j-2}, x_{j-1}], \\ U(x) &= q_{j+1}(x, U^k) + O(h^r), & x \in [x_j, x_{j+1}], \end{aligned} \quad (1.48)$$

A localização da descontinuidade pode ser recuperada através da seguinte função:

$$G_j(x) = q_{j+1}(x, U^k) - q_{j-1}(x, U^k). \quad (1.49)$$

Usando expansões em Taylor nas regiões de suavidade, [1], mostra-se que:

$$G_j(x_{j-1})G_j(x_j) = a(a-1)[U']_{x_d}^2 h^2 + O(h^3),$$

com $x_d = x_j - ah$, $0 < a < 1$. Assim, se h for suficientemente pequeno, existirá uma raiz de G_j em (x_{j-1}, x_j) . Seja $\theta_j \in (x_{j-1}, x_j)$ tal que $G_j(\theta_j) = 0$. Considera-se θ_j uma aproximação para x_d . Em [11] mostra-se que $|\theta_j - x_d| = O(h^r)$.

Modificação do Interpolador

Esta informação sobre a localização aproximada de x_d pode ser utilizada agora para modificar a definição da parte polinomial correspondente a célula $[x_{j-1}, x_j]$, da seguinte forma: ao invés de considerar $q_j(x, U^k)$ como aproximação para $U(x)$ nesta célula, estendem-se as partes polinomiais dos intervalos esquerdo e direito até o ponto θ_j , onde eles se intersectam. Com isso, a nova parte polinomial que interpola U na célula que contém a descontinuidade tem a seguinte forma:

$$I^{SR}(x, U^k) = \begin{cases} q_l(x, U^k), & x \in [x_{l-1}, x_l], \quad l \neq j \\ q_{j-1}(x, U^k), & x \in [x_{j-1}, \theta_j], \\ q_{j+1}(x, U^k), & x \in [\theta_j, x_j], \end{cases} \quad (1.50)$$

Claramente,

$$I^{SR}(x_j^k, U^k) = U_j^k, \quad x_j^k \in X_k. \quad (1.51)$$

E por [1], tem-se também que:

$$\frac{d^m}{dx^m} I^{SR}(x \pm 0, U^k) = \frac{d^m}{dx^m} U(x) + O(h^{r-m}), \quad 0 \leq m \leq r-1, \quad (1.52)$$

em todos os pontos exceto para uma faixa de $O(h^r)$ em torno de x_d , que agora é a única região na qual a precisão ficou degenerada, ao invés de toda célula $[x_{j-1}, x_j]$.

Em [1] ainda são discutidas outras modificações no modo de construção do interpolador dependendo do tipo de descontinuidade apresentada pela função U .

Algoritmo SR

1. Aplicar interpolação ENO em todas as células do Domínio usando **Algoritmo 1** para escolha do estêncil.
2. Usar o critério $S_{j-1} \cap S_j = \emptyset$ para selecionar células suspeitas de conterem singularidades. Estas células serão chamadas de células *singulares*.
3. Localizar a descontinuidade dentro das células singulares usando a função G_j .
4. Modificar a definição do interpolador nas células singulares usando (1.50).

Em [1], os autores enfatizam que, na prática, quando uma célula for classificada como *singular* em algum nível de resolução k , por exemplo, não é necessário saber explicitamente a posição θ_j^k , apenas sua posição relativa em relação aos pontos x_{2j-1}^{k-1} . Esta pode ser encontrada apenas pela avaliação do sinal de $G_j^k(x_{2j-2}^{k-1})G_j^k(x_{2j-1}^{k-1})$.

Para as mesmas funções escolhidas (1.23), aplicam-se agora as transformadas interpolatórias utilizando as médias celulares das funções dadas em uma malha diádica inicial.

1.2.3 Parte Computacional

Nesta Seção serão apresentados alguns exemplos numéricos para mostrar os resultados dos algoritmos para transformada direta e inversa e o comportamento dos detalhes para as mesmas funções escolhidas dadas em (1.23):

Os gráficos a seguir mostram a transformada direta da função $v(x) = \sin(2\pi x)$ utilizando suas médias celulares e considerando interpolador associado a $\bar{r} = 5$. O nível mais refinado $k = 0$ contém 2^{n_0} pontos com $n_0 = 7$. São considerados $L = 4$ níveis de faturação, portanto o nível mais grosseiro possui 2^3 pontos.

Os gráficos dos detalhes a cada nível de faturação da função $v(x)$ são apresentados nas figuras (1.15) e (1.16). Observa-se que para função $v(x)$, o valor máximo do módulo dos detalhes em cada nível diminui quando o nível de resolução aumenta, como indica a estimativa (1.17) para funções suaves no contexto de médias celulares.

nível k	$\max_j d_j^k $
1	8.8822302957148D-08
2	2.8192714293385D-06
3	8.7318535709688D-05
4	2.4483304058511D-03

Tabela 1.4: Máximo dos detalhes de $v(x)$ por nível Figura 1.12: Médias Celulares da função $v(x)$ no nível mais refinado, $n_0 = 7$ Figura 1.13: Níveis de resolução de $v(x)$, $n_1 = 6$ e $n_2 = 5$

Figura 1.14: Níveis de resolução de $v(x)$, $n_3 = 4$ e $n_4 = 3$

Figura 1.15: Detalhes de $v(x)$ nos níveis 1 e 2

Figura 1.16: Detalhes de $v(x)$ nos níveis 3 e 4

O próximo exemplo é a função pulso, $u(x)$, cujo nível mais refinado, $k = 0$, também contém $n_0 = 2^7$ pontos. Os gráficos abaixo mostram sua faturação em $L = 4$ níveis e os respectivos detalhes. Neste exemplo, observa-se que o máximo dos detalhes em módulo permanece na mesma ordem para todos os níveis, uma vez que a função possui duas descontinuidades do tipo salto em $[0, 1]$. Por (1.47) tem-se $|d_j^k| \sim (h_k)^0 [u^{(0)}] \sim O(1)$.

Figura 1.17: Médias Celulares da função $u(x)$ no nível mais refinado, $n_0 = 7$

Na tabela (1.5) são dados os valores do máximo dos detalhes em módulo para cada nível de decomposição. Como $u(x)$ é descontínua, o decaimento dos detalhes não segue as taxas apresentadas nas sessões anteriores para funções suaves.

nível k	$\max_j d_j^k $
1	0.50000000
2	0.50000000
3	0.50000000
4	0.50000000

Tabela 1.5: Valor máximo dos detalhes de $u(x)$ por nível

Figura 1.18: Níveis de Resolução da função $u(x)$, $n_1 = 6$ e $n_2 = 5$

Figura 1.19: Níveis de Resolução da função $u(x)$, $n_3 = 4$ e $n_4 = 3$

Os gráficos dos detalhes para cada nível de faturação estão nas figuras (1.20) e (1.21). Observa-se que a localização dos detalhes com maior módulo é exatamente nas regiões onde a função é descontínua.

Figura 1.20: Detalhes da função $u(x)$ nos níveis 1 e 2 Figura 1.21: Detalhes da função $u(x)$ nos níveis 3 e 4

O terceiro exemplo é a função $w(x)$ que apresenta um salto na derivada primeira. Os valores máximos dos módulos dos detalhes por nível de resolução são apresentados na tabela (1.6). Pela estimativa dada em (1.47), $|d_j^k| \sim (h_k)^1[w^{(1)}]$, o que explica a diferença do comportamento dos coeficientes para funções suaves e funções não suaves.

nível k	$\max_j d_j^k $
1	7.4841471107973D-03
2	1.5004554389334D-02
3	3.0308329134750D-02
4	6.3263438229339D-02

Tabela 1.6: Valor máximo dos detalhes de $w(x)$ por nível

Figura 1.22: Médias Celulares da função $w(x)$ no nível mais refinado

Figura 1.23: Fatoração de $w(x)$ em quatro níveis

Os gráficos dos detalhes a cada nível de fatoração seguem abaixo. A localização dos detalhes de maior módulo indica onde a função deixa de ser suave.

Figura 1.24: Detalhes de $w(x)$ nos quatro níveis de fatoração

Com isso, pode-se observar que tanto no contexto de valores pontuais quanto no de médias celulares, o decaimento dos coeficientes wavelet está relacionado com a suavidade da função. No caso da função descontínua, o módulo dos coeficientes permaneceu constante em todos os níveis da transformação. Já a função suave apresentou detalhes cujo comportamento seguiu o previsto pelas estimativas (1.16) e (1.17). Outro aspecto interessante é a propriedade apresentada pela transformação para valores pontuais de manter inalterados os valores entre posições comuns a um nível e outro de transformação. O que não ocorre no contexto de médias celulares, as médias de um nível não se mantêm nos demais níveis de transformação, como indica a relação de escala (1.31).

Capítulo 2

Compressão de Dados

Uma das vantagens de utilização da representação multiresolução é possibilitar a inserção de algum procedimento de compressão ou processamento de dados entre a transformada direta e a sua inversa.

Uma vez que a representação multiresolução de uma seqüência discreta de valores, u^0 , pode ser vista como uma decomposição de u^0 em escalas, os coeficientes wavelets d_j^k podem ser interpretados como sendo informações “complementares” que ainda devem ser agregadas aos valores previstos pelos coeficientes u^k para a obtenção dos valores de u^{k-1} no nível mais refinado, ou seja, os coeficientes wavelet d_j^k guardam os detalhes que foram retirados de um nível mais refinado para a formação do nível mais grosseiro.

Como este processo de passagem de um nível para outro é feito através de interpolação, os coeficientes d_j^k possuem, além dos detalhes intrínsecos a uma nova escala, uma componente relativa ao erro causado pelo interpolador. Pode-se dizer, então, que os coeficientes wavelet carregam duas informações: uma que não pode ser eliminada, pois é fundamental para a reconstrução da função na escala original, e outra devido ao interpolador utilizado.

O processo de compressão de dados visa exatamente “eliminar” esta última componente da representação multiresolução. Descartam-se, então, coeficientes wavelet cuja norma esteja abaixo de uma certa tolerância ϵ produzindo uma representação multiresolução *truncada*. De fato, os coeficientes wavelet menores do que ϵ são substituídos por zero na representação. E após a aplicação da transformada wavelet inversa, obtém-se uma aproximação para a função original.

A tolerância, muitas vezes, está relacionada com a ordem do preditor utilizado. Na verdade, diferentes escolhas para a tolerância implicam em um número maior ou menor de coeficientes wavelet descartados, produzindo diferentes graus de compressão, e conseqüentemente, produzindo aproximações, em algum sentido, “melhores” ou “piores”.

É importante salientar que os coeficientes wavelets quando não descartados da representação multiresolução, indicam que, na posição correspondente, o valor da função não pode

ser substituído com uma boa precisão pelo valor fornecido pelo preditor, sendo, então, um indicador da presença ou do surgimento de uma “nova” escala.

Quando esta representação truncada e sua inversa ainda forem armazenadas de forma eficiente (esparsa), consegue-se uma redução significativa do espaço exigido para o armazenamento dos dados e conseqüentemente um aumento na velocidade de processamento destes.

Define-se, então, na seção 1 o operador de truncamento, passo fundamental para realização da compressão de dados. Na seção 2, aplicam-se as diferentes estratégias de compressão aos exemplos numéricos do capítulo anterior e verifica-se, para cada caso, a quantidade de coeficientes significativos obtida. E na seção 3 apresenta-se o conceito de representação pontual esparsa, definido por Holmström em [20].

2.1 Truncamento

No Capítulo 1 foram apresentadas estimativas para o módulo dos detalhes para cada nível k : a expressão (1.17) fornece uma estimativa para função suave e (1.19), para função com descontinuidade utilizando valores pontuais nas transformações. Já a expressão (1.47) fornece uma estimativa para função com descontinuidade utilizando médias celulares.

A idéia central da compressão de dados é descartar da representação multiresolução coeficientes wavelet que tenham módulo “muito pequeno”, menor do que a cota apresentada na estimativa (1.17), restando apenas aqueles denominados *coeficientes significativos*. Isto é possível, pois o detalhe d_j^k informa se o valor obtido pelo preditor está ou não próximo do valor real do nível mais fino e os detalhes descartados são exatamente aqueles cujos preditores fornecem boas aproximações para os valores da próxima escala mais refinada.

O procedimento de compressão mais simples e usado em análise numérica é definido da seguinte forma:

$$\hat{d}_j^k = \text{tr}_{\epsilon_k}(d_j^k) = \begin{cases} 0, & \text{se } |d_j^k| \leq \epsilon_k \\ d_j^k, & \text{c. c.,} \end{cases} \quad (2.1)$$

com tr_{ϵ_k} sendo denominado operador de truncamento e ϵ_k , uma certa tolerância escolhida para o erro em cada nível k .

Segundo Percival e Walden, [34], este operador é denominado operador de truncamento *duro* (*hard thresholding*), pois todo coeficiente que exceder ϵ_k fica inalterado e não existe nenhum coeficiente wavelet que possua valor entre $-\epsilon_k$ e ϵ_k . Na figura (2.1) este operador é representado pela linha contínua.

Outro operador apresentado em [34], denominado operador de truncamento *suave* (*soft thresholding*), tem a seguinte forma:

$$\hat{d}_j^k = \text{tr}_{\epsilon_k}(d_j^k) = \text{sign}(d_j^k)(|d_j^k| - \epsilon_k)_+, \quad (2.2)$$

sendo

$$\text{sign}(d_j^k) = \begin{cases} +1, & \text{se } d_j^k > 0; \\ 0, & \text{se } d_j^k = 0; \\ -1, & \text{se } d_j^k < 0; \end{cases} \quad \text{e} \quad (x)_+ = \begin{cases} x, & \text{se } x \geq 0; \\ 0, & \text{se } x \leq 0; \end{cases}$$

Os coeficientes que excedem ϵ_k em magnitude são diminuídos de ϵ_k e aqueles com magnitude menor ou igual a ϵ_k são anulados. Este operador é representado com linhas tracejadas na figura (2.1).

Ainda é possível a obtenção de um operador que seja uma mistura dos operadores duro e suave, o que em [34] é denominado *mid thresholding*, ou seja, um operador de truncamento médio, definido por:

$$\hat{d}_j^k = \text{tr}_{\epsilon_k}(d_j^k) = \text{sign}(d_j^k)(|d_j^k| - \epsilon_k)_{++}, \quad (2.3)$$

sendo

$$(|d_j^k| - \epsilon_k)_{++} = \begin{cases} 2(|d_j^k| - \epsilon_k)_+, & \text{se } |d_j^k| \leq 2\epsilon_k; \\ |d_j^k|, & \text{caso contrário;} \end{cases}$$

Aqui os coeficientes, cujos módulos são maiores do que $2\epsilon_k$, ficam inalterados. Aqueles com módulo entre ϵ_k e $2\epsilon_k$ são encolhidos e os demais são considerados nulos. Este operador é representado pela linha pontilhada na figura (2.1).

Figura 2.1: Operadores de truncamento

Estes operadores também são muito utilizados na análise de séries temporais, tomando-se $\epsilon_k = \delta$, para todo nível k , sendo δ uma constante fixada [34].

Após a aplicação do operador de truncamento em cada nível, obtém-se então uma representação multiresolução *truncada* denotada por:

$$\hat{u}_M = (\hat{d}^1, \hat{d}^2, \dots, \hat{d}^L, u^L) \quad (2.4)$$

Aplicando a $\hat{u}_M$ o algoritmo (1.15) para transformação inversa, obtém-se uma aproximação para o vetor inicial u^0 .

$$\tilde{u}^0 = M^{-1}\hat{u}_M \quad (2.5)$$

Harten [15] mostra, no caso da transformação envolver médias celulares da função, que a diferença entre o dado original e o obtido através da representação truncada está limitada por:

$$\|u^0 - \tilde{u}^0\| \leq c \sum_{k=1}^L \epsilon_k \quad (2.6)$$

e que considerando $\epsilon_k = 2^{-k}\epsilon/c$, obtém-se

$$\|u^0 - \tilde{u}^0\| \leq \epsilon, \quad (2.7)$$

sendo ϵ a tolerância para o erro total de compressão e ϵ_k a tolerância para cada nível.

Na realidade, na obtenção de (2.6) e (2.7) é essencial que os algoritmos para transformada direta e inversa sejam estáveis com respeito a perturbações. Em [17], Harten e Yad demonstram a estabilidade do algoritmo para transformada wavelet inversa com respeito a perturbações na representação multiresolução.

Para transformações lineares, Arandiga *et al.*, [1], mostra que: se o algoritmo de fatoração for estável, então dada uma perturbação para um dado inicial u_δ^0 , perturbações nos coeficientes wavelets d_δ^k e no nível mais grosso obtidos após a fatoração u_δ^L são limitadas pela perturbação imposta ao dado inicial.

Já para técnicas não lineares, cuja transformada depende dos pontos do estêncil considerado, as estimativas apresentadas em [1] não podem ser mais utilizadas e a questão da estabilidade é posta da seguinte forma por Arandiga *et al.* em [2]: *Dada uma seqüência f^0 e um nível de tolerância ϵ para a precisão, deseja-se obter uma fatoração truncada $\hat{f}_M = (f^L, \hat{d}^L, \dots, \hat{d}^1)^t$, tal que $\tilde{f}^0 = M^{-1}\hat{f}_M$ esteja próximo do dado inicial para uma norma apropriada, isto é, $\|f^0 - \tilde{f}^0\| \leq C\epsilon$.*

Sendo assim, em ambos os casos, estudar o efeito de usar $\hat{d}_j^k$ (detalhes truncados) ao invés de usar d_j^k para reconstruir o sinal é equivalente a estudar o efeito de uma perturbação nos coeficientes wavelet $d_{\delta j}^k = (d_j^k + \delta_j^k)$ no resultado da transformação inversa.

O erro introduzido pela perturbação nos coeficientes wavelet no resultado do algoritmo de reconstrução pode ser estimado por análise, mas não pode ser controlado diretamente. O que se pode controlar é a *taxa de compressão* de acordo com escolha feita para $\{\epsilon_k\}_{k=1}^L$.

Segundo Harten, [15], a taxa de compressão é dada por:

$$tc = \frac{N_0}{|D_\varepsilon| + N_L}, \quad (2.8)$$

onde N_0 é o número de pontos na escala mais fina, $N_L = N_0/2^L$ é o número de pontos na escala mais grossa e $|D_\varepsilon|$ é a quantidade de elementos no conjunto D_ε dos índices dos coeficientes wavelet significativos, $D_\varepsilon = D_\varepsilon(u^0) = \{(j, k) \mid |d_j^k| > \epsilon_k\}$.

Uma observação ainda é que se todos os coeficientes wavelet forem descartados da representação multiresolução, a taxa de compressão, tc , será máxima, $tc = N_0/N_L = 2^{n_0-L}$ e neste caso, o grau de compressão também será máximo, restando apenas as informações no nível mais grosseiro para representar a função. Qualquer outro valor na malha será calculado a partir de sucessivas interpolações a partir destes dados na malha grosseira. Caso nenhum coeficiente d_j^k seja descartado, tem-se $|D_\varepsilon| = N_0$ e nenhuma compressão será obtida, $tc = N_0/N_0 = 1$.

Em aplicações envolvendo transmissão de sinais pela internet, por exemplo, tem-se por objetivo conseguir a representação mais esparsa possível, ou a taxa de compressão maior possível, preservando a qualidade do sinal. Já no caso de aplicações que utilizem a representação esparsa na resolução de operadores diferenciais, devem ser incluídos algumas escalas por segurança, uma vez que a solução obtida ainda deverá evoluir no tempo. No próximo capítulo, serão discutidos tópicos referentes a este segundo tipo de aplicação e a evolução da solução em uma forma esparsa.

2.2 Estratégias de Compressão

Como foi observado na seção anterior, a taxa de compressão varia com a escolha feita para ε e $\{\epsilon_k\}_{k=1}^L$. A escolha para $\{\epsilon_k\}_{k=1}^L$ é feita de acordo com as estimativas de decaimento dos coeficientes wavelets, por exemplo as expressões (1.17) e (1.19) apresentadas no Capítulo 1, ou ainda, as estimativas apresentadas por Cohen em [7], obtidas dependendo do espaço no qual a função inicial é considerada. Então, fixada a tolerância ε , algumas possibilidades para as estratégias de compressão, também chamadas estratégias de corte, ϵ_k , são as seguintes:

1. *Estratégia 1* Segundo Harten [15], uma possibilidade de escolha é: $\epsilon_k = \frac{\varepsilon}{2^k}$, ou seja, os valores ϵ_k diminuem quando k cresce (quando a escala varia da mais fina para a mais grossa).
2. *Estratégia 2* Outra possibilidade é: $\epsilon_k = \frac{\varepsilon}{2^{n_0-k}}$. Os valores ϵ_k neste caso crescem com a escala k .
3. *Estratégia 3* Propomos uma forma “híbrida” que mistura as duas possibilidades anteriores é $\epsilon_k = \frac{\varepsilon}{2^{L-k}}$, com L = número de níveis na transformação. Os valores ϵ_k também crescem com o aumento da escala, só que ao contrário de Harten, tem-se fixada a tolerância ε no nível mais grosso L .

Ainda existem outras estratégias consideradas em [1], [2] e a mais simples de todas que é $\epsilon_k = \epsilon, \forall k$, mas que não serão estudadas nesta Seção.

Resultados Computacionais

As funções (1.23) utilizadas nos testes computacionais do capítulo (1) serão também consideradas nesta Seção para o estudo da taxa de compressão e sua relação com a estratégia de compressão empregada.

$$v(x) = \text{sen}(2\pi x), \quad x \in [0, 1]$$

$$w(x) = |\text{sen}(\frac{3}{2}\pi(x - \frac{1}{2}))|, \quad x \in [0, 1]$$

$$u(x) = \begin{cases} 1, & 0.25 \leq x < 0.75 \\ 0, & 0 \leq x < 0.25, \text{ e } 0.75 \leq x \leq 1 \end{cases}$$

Então, para cada estratégia analisam-se a taxa de compressão e a quantidade total de coeficientes significativos. Os gráficos mostram suas posições por escala para cada uma das funções-exemplo e para diferentes tipos de transformação. Nesta seqüência de testes

- o nível mais refinado considerado possui $N_0 = 2^{n_0}$ pontos com $n_0 = 9$,
- $L = 6$ níveis de transformação, portanto a escala mais grosseira possuirá $N_L = 2^{9-6} = 2^3$ pontos.

Quando for usado preditor linear para valores pontuais no algoritmo de fatoração, a transformação é denominada de *Transformada Wavelet Pontual Linear* (TWP1). Quando o preditor for cúbico, *Transformada Wavelet Pontual Cúbica* (TWP3) e quando o preditor for de grau 5, *Transformada Wavelet Pontual de grau 5* (TWP5)

Para as fatorações envolvendo médias celulares, usa-se a mesma regra de classificação. *Transformada Wavelet para Média Celular de grau 3* (TWMC3), quando $\bar{r} = 3$ e *Transformada Wavelet para Média Celular de grau 5* (TWMC5), quando $\bar{r} = 5$.

2.2.1 Estratégia 1 $\epsilon_k = \frac{\epsilon}{2^k}$

Teste - transformada fixada, valor de corte variando: Função $v(x)$

A primeira função a ser testada é a função $v(x) = \text{sen}(2\pi x)$ do exemplo (1.23). Os gráficos da figura (2.2) mostram as posições dos coeficientes significativos obtidas para TWP1 quando $\epsilon = 10^{-5}$ e $\epsilon = 10^{-4}$. Na figura (2.3) estão as posições significativas por nível obtidas quando $\epsilon = 10^{-3}$ e $\epsilon = 10^{-2}$, para a mesma transformação. Todas as posições são dadas em relação à malha mais refinada para o domínio da função, o intervalo $[0,1]$. As diferentes alturas

com que os pontos significativos são apresentados apenas indicam o nível de surgimento do coeficiente, conforme o nível da transformação. Na altura 1, por exemplo, estão os coeficientes obtidos no primeiro nível de transformação, e assim sucessivamente.

Figura 2.2: TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$

Figura 2.3: TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$

Têm-se ainda as posições significativas dos coeficientes de $v(x)$ quando é utilizado preditor de grau 3. Novamente são apresentados na figura (2.4) gráficos com os resultados para $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$, e na figura (2.5) os resultados para $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$

Figura 2.4: TWP3 $v(x)$:Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$

Figura 2.5: TWP3 $v(x)$:Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$

Seguem as posições significativas dos coeficientes de $v(x)$ quando é utilizado preditor de grau 5. São mostrados os resultados para $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$ na figura (2.6), e para $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$ na figura (2.7).

Figura 2.6: TWP5 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$

Figura 2.7: TWP5 $v(x)$: Posições significativas $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$

O total de coeficientes significativos para cada ε para cada uma das transformações utilizadas está na tabela (2.2.1) na parte esquerda e na parte direita, estão as taxas de compressão. Pode-se observar que para a faixa de valores escolhida para ε , a TWP1 teve as taxas de compressão mais baixas, no entanto todas as transformações, TWP1, TWP3 e TWP5, mantiveram a mesma tendência, aumentando a compressão conforme ε cresceu. Para taxas de compressão elevadas, é natural que a quantidade de pontos significativos na malha seja pequena e vice-versa. Observa-se também que conforme a ordem da transformação aumentou, obteve-se um aumento da taxa de compressão.

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	492	96	24	10^{-5}	1.02	4.92	16.00
10^{-4}	376	52	24	10^{-4}	1.33	8.53	16.00
10^{-3}	164	24	8	10^{-3}	2.97	16.00	32.00
10^{-2}	52	16	8	10^{-2}	8.53	21.33	32.00

Tabela 2.1: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixada, valor de corte variando: Função $u(x)$

A próxima função é a função pulso $u(x)$. Os gráficos a seguir repetem os testes anteriores para a *estratégia 1*. Não houve variação da taxa de compressão com o valor atribuído para ε , apenas com o grau do interpolador. Abaixo seguem as posições significativas dos coeficientes de $u(x)$ quando é utilizado preditor de grau 1, figura (2.8), preditor de grau 3, figura (2.9) e preditor de grau 5, figura (2.10).

Figura 2.8: TWP1 $u(x)$: Posições significativas, independente do valor de ε Figura 2.9: TWP3 $u(x)$: Posições significativas, independente do valor de ε

Figura 2.10: TWP5 $u(x)$:Posições significativas, independente do valor de ε

O total de coeficientes significativos para cada ε para cada uma das transformações utilizadas está na tabela (2.2) à esquerda e as correspondentes taxas de compressão estão na tabela (2.2) à direita. Os valores para a taxa de compressão ilustram um comportamento já esperado, uma vez que a função apresenta descontinuidade tipo salto e por isso seus coeficientes wavelet não decaem. Houve portanto apenas a atuação do grau do preditor sobre a taxa de compressão. Outra observação interessante é que o número de pontos mantido na representação da função na vizinhança da descontinuidade aumentou, conforme aumentou a ordem da transformação.

ε	TWP1	TWP3	TWP5	TWP1	TWP3	TWP5
10^{-5}	12	36	58	25.60000	11.636363	7.757575
10^{-4}	12	36	58	25.60000	11.636363	7.757575
10^{-3}	12	36	58	25.60000	11.636363	7.757575
10^{-2}	12	36	58	25.60000	11.636363	7.757575

Tabela 2.2: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixada, valor de corte variando: Função $w(x)$

A terceira função a ser testada é $w(x)$. Novamente mantém-se a primeira estratégia fixada e variam-se o grau do interpolador e o valor de corte atribuído no operador de truncamento. Na figura (2.11) o interpolador utilizado na transformação foi o linear. Nas figuras (2.12) e (2.13), o interpolador utilizado foi o cúbico e na figura (2.14), o de grau cinco.

Figura 2.11: TWP1 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

Figura 2.12: TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}

Figura 2.13: TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}

Figura 2.14: TWP5 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

A quantidade de coeficientes significativos para $w(x)$ e as correspondentes taxas de compressão estão na tabela (2.3), partes direita e esquerda respectivamente.

Uma observação quanto às posições significativas, no caso da TWP1 não houve contribuição de coeficientes wavelets das escalas mais refinadas para representar a região não suave, pois para as escalas consideradas, esta região foi aproximada linearmente. Já para as

transformações TWP3 e TWP5, a região não suave foi “bem capturada” pelos coeficientes wavelet.

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	492	68	56	10^{-5}	1.0240000	6.736842	8.0000000
10^{-4}	240	62	50	10^{-4}	2.0645161	7.314285	8.8275862
10^{-3}	114	40	44	10^{-3}	4.1967213	10.666666	9.8461538
10^{-2}	52	24	34	10^{-2}	8.5333333	16.000000	12.1904761

Tabela 2.3: Número total de coeficientes significativos e Taxas de compressão

2.2.2 Estratégia 2 $\epsilon_k = \frac{\varepsilon}{2^{n_0-k}}$

Teste - transformada fixada, valor de corte variando: Função $v(x)$

Para cada transformação via valores pontuais da função, os gráficos mostram as posições dos coeficientes significativos por escala de acordo com os quatro valores escolhidos para a tolerância. Nas figuras (2.15) e (2.16) são apresentadas as posições significativas obtidas para transformada linear quando o valor de corte é assumido $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$, $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$. Na figura (2.17) a transformação utilizada é a cúbica e consideram-se os mesmos valores para ε . E na figura (2.18) é utilizada transformada de grau 5 com $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$ e $\varepsilon = 10^{-3}$.

Figura 2.15: TWP1 $v(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}

Figura 2.16: TWP1 $v(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}

Figura 2.17: TWP3 $v(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

A quantidade de coeficientes significativos para $v(x)$ e as correspondentes taxas de compressão estão na tabela (2.4). Observa-se que praticamente não houve compressão quando utiliza-se TWP1 para estes valores de ε , e para as transformações com preditores de grau maior, as taxas de compressão foram realmente expressivas.

Figura 2.18: TWP5 $v(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} e 10^{-3}

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	504	392	24	10^{-5}	1.0000000000	1.280000000	16.000000000
10^{-4}	504	88	20	10^{-4}	1.0000000000	5.333333333	18.285714285
10^{-3}	492	32	8	10^{-3}	1.0240000000	12.800000000	32.000000000
10^{-2}	492	8	0	10^{-2}	1.0240000000	32.000000000	64.000000000

Tabela 2.4: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixada, valor de corte variando: Função $u(x)$

Não houve variação da taxa de compressão com ε para nenhuma das transformadas utilizadas, houve apenas variação com o grau do interpolador utilizado, como já havia ocorrido para a *estratégia 1*. São apresentados na figura (2.19) apenas os gráficos para $\varepsilon = 10^{-5}$ para cada um dos tipos de transformação.

Figura 2.19: Posições significativas para $u(x)$ independente do valor de ε : TWP1, TWP3 e TWP5

A quantidade de coeficientes significativos para $u(x)$ e as taxas de compressão para a função $u(x)$ são dadas pelas tabelas a seguir. A TWP1 forneceu a melhor taxa de compressão uma vez que a função inicial é linear por partes com descontinuidade tipo salto.

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	12	36	58	10^{-5}	25.6000	11.63636	7.75757
10^{-4}	12	36	58	10^{-4}	25.6000	11.63636	7.75757
10^{-3}	12	36	58	10^{-3}	25.6000	11.63636	7.75757
10^{-2}	12	36	58	10^{-2}	25.6000	11.63636	7.75757

Tabela 2.5: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixada, valor de corte variando: Função $w(x)$

No teste feito para transformação linear, a quantidade de significativos para $\varepsilon = 10^{-5}$ e para $\varepsilon = 10^{-4}$ é praticamente a mesma, 504 e 502 coeficientes respectivamente. Apresentam-se, então, os gráficos para $\varepsilon = 10^{-5}$, para $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$ na figura (2.20). Para a

transformada cúbica, os resultados para $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$, $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$ estão nas figuras (2.21) e (2.22). E para os mesmos valores de ε , os resultados para transformada de grau 5 estão na figura (2.23).

Figura 2.20: TWP1 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-3} e 10^{-2}

Figura 2.21: TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-5}$ e 10^{-4}

Figura 2.22: TWP3 $w(x)$: Posições significativas $\varepsilon=10^{-3}$ e 10^{-2}

Figura 2.23: TWP5 $w(x)$: Posições significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

A quantidade de coeficientes significativos para $w(x)$ e as taxas de compressão associadas estão na tabela (2.6). Pela definição obtém-se valores de corte ϵ_k menores para as escalas refinadas, o que provoca um corte grande de coeficientes nestas escalas. Para os níveis mais grosseiros ocorre o contrário, ϵ_k fica maior e portanto um número menor de coeficientes são desprezados.

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	504	124	56	10^{-5}	1.00000000	3.87878787	8.00000000
10^{-4}	502	66	48	10^{-4}	1.00392156	6.91891891	9.14285714
10^{-3}	492	40	48	10^{-3}	1.02400000	10.66666666	9.14285714
10^{-2}	300	28	48	10^{-2}	1.66233766	14.22222222	9.14285714

Tabela 2.6: Número total de coeficientes significativos e Taxas de compressão

2.2.3 Estratégia 3 $\epsilon_k = \frac{\varepsilon}{2^{L-k}}$

Nesta estratégia, os valores de corte ϵ_k crescem conforme a malha vai ficando mais grosseira. E a tolerância ε é exatamente o valor de corte do nível mais grosseiro n_L .

Teste - transformada fixada, valor de corte variando: Função $v(x)$

Nas figuras (2.24) e (2.25) estão os resultados para a transformada linear. Os casos $\varepsilon = 10^{-5}$ e $\varepsilon = 10^{-4}$ produzem o mesmo número de coeficientes significativos. Para a TWP3 $\varepsilon = 10^{-4}$ e $\varepsilon = 10^{-3}$ possuem o mesmo número de coeficientes nas mesmas escalas. Para o caso $\varepsilon = 10^{-2}$, todos os coeficientes wavelet foram descartados, sendo então mostrados apenas os casos relevantes na figura (2.26). E para TWP5 os resultados são mostrados na figura (2.27). $\varepsilon = 10^{-4}$ e $\varepsilon = 10^{-3}$ possuem o mesmo número de significativos e para $\varepsilon = 10^{-2}$ todos os coeficientes foram descartados.

Figura 2.24: TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-3}

Figura 2.25: TWP1 $v(x)$: Posições significativas $\varepsilon = 10^{-2}$ Figura 2.26: TWP3 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-4} Figura 2.27: TWP5 $v(x)$: Posições significativas $\varepsilon = 10^{-5}$ e 10^{-4}

A quantidade de coeficientes significativos para $v(x)$ e suas taxas de compressão são dadas na tabela (2.7) a seguir.

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	504	96	20	10^{-5}	1.00000000	4.92307692	18.2857142
10^{-4}	504	40	8	10^{-4}	1.00000000	10.66666666	32.0000000
10^{-3}	416	40	8	10^{-3}	1.20754716	10.66666666	32.0000000
10^{-2}	40	0	0	10^{-2}	10.66666666	64.00000000	64.0000000

Tabela 2.7: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixa, valor de corte variando: Função $u(x)$

Novamente não houve variação da quantidade de significativos com o valor atribuído para a tolerância, apenas com o grau do polinômio interpolador. Abaixo na figura (2.28) seguem os gráficos para cada um dos tipos de interpolador utilizado.

Figura 2.28: Posições significativas para $u(x)$ independente do valor de ε : TWP1, TWP3 e TWP5

A TWP1 novamente descartou o maior número de coeficientes do que todos os demais testes. Na tabela (2.8) estão os valores para o número de significativos por transformação e a taxa de compressão por transformação, para cada um dos valores atribuídos para ε .

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	12	36	58	10^{-5}	25.6000	11.63636	7.75757
10^{-4}	12	36	58	10^{-4}	25.6000	11.63636	7.75757
10^{-3}	12	36	58	10^{-3}	25.6000	11.63636	7.75757
10^{-2}	12	36	58	10^{-2}	25.6000	11.63636	7.75757

Tabela 2.8: Número total de coeficientes significativos e Taxas de compressão

Teste - transformada fixada, valor de corte variando: Função $w(x)$

Na figura (2.29) estão apresentados os resultados para o interpolador linear. Nas figuras (2.30), (2.31) e (2.32) estão os resultados para TWP3 e para TWP5, respectivamente.

Figura 2.29: TWP1 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

Figura 2.30: TWP3 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$, 10^{-4} , 10^{-3} e 10^{-2}

Figura 2.31: TWP5 $w(x)$: Posições Significativas $\varepsilon=10^{-5}$ e 10^{-4}

Figura 2.32: TWP5 $w(x)$: Posições Significativas $\varepsilon=10^{-3}$ e 10^{-2}

Pelos gráficos da figura (2.29) observa-se que para valores de ε menores do que 10^{-5} , a compressão será insignificante. Já valores entre 10^{-3} e 10^{-2} podem produzir representações esparsas com uma melhor distribuição de pontos significativos entre as escalas. E nas figuras (2.30), (2.31) e (2.32) também pode-se perceber a influência da tolerância sobre a representação esparsa.

O número de coeficientes significativos para $w(x)$ e a taxa de compressão seguem na tabela (2.9) abaixo:

ε	TWP1	TWP3	TWP5	ε	TWP1	TWP3	TWP5
10^{-5}	504	66	48	10^{-5}	1.000000000	6.918918918	9.142857142
10^{-4}	494	40	48	10^{-4}	1.019920318	10.666666666	9.142857142
10^{-3}	382	28	48	10^{-3}	1.312820512	14.222222222	9.142857142
10^{-2}	30	24	36	10^{-2}	13.473684210	16.000000000	11.636363636

Tabela 2.9: Número total de coeficientes significativos e Taxas de compressão

2.2.4 Análise Comparativa

Apresentam-se agora as tabelas com o número de significativos e com as taxas de compressão correspondentes para cada função, para cada tipo de transformação e para cada ε considerado.

Observa-se que todas as estratégias apresentaram os mesmos resultados para a função $u(x)$. Já para as outras duas funções, as *estratégias 1* e *3* tiveram resultados sempre comparáveis em termos das taxa de compressão obtidas. A *estratégia 2* considerou praticamente em todos os exemplos analisados um número maior de coeficientes significativos de acordo com o grau do interpolador utilizado, conseqüentemente fornecendo taxas de compressão

menores. Já a *estratégia 3* possui a vantagem de se identificar “a priori” o valor de corte no nível mais grosseiro.

Mesmo para a TWP1 que forneceu baixas taxas de compressão para funções suaves, a escolha da tolerância ε é muito importante. Dependendo desta escolha, uma estratégia pode fornecer taxas de compressão melhores ou piores, uma vez que ainda outros fatores são relevantes neste processo, como por exemplo, a suavidade da função.

Na maioria dos casos testados para cada uma das estratégias, TWP3 e TWP5 produziram resultados qualitativamente comparáveis em relação às posições significativas. Já as taxas de compressão para TWP5 foram maiores do que para TWP3, no entanto, em termos de custo computacional a TWP3 é mais vantajosa, pois requer um número menor de operações. Nas aplicações do capítulo 3 serão utilizadas TWP3, pela facilidade de implementação e TWP1, para dados iniciais tipo função pulso.

As tabelas (2.10) e (2.11) mostram os valores obtidos para o número de significativos e para a taxa de compressão pelas *estratégias 1, 2 e 3* para a função $v(x)$. A ordem da transformação foi considerada 1, 3 ou 5 e os valores de corte foram considerados entre 10^{-2} e 10^{-5} .

Da mesma forma são apresentados nas tabelas (2.12) e (2.13) os resultados para a função $w(x)$. E para a função $u(x)$ os valores para a quantidade de pontos na malha esparsa e as respectivas taxas de compressão estão nas tabelas (2.14) e (2.15).

		Estratégia 1	Estratégia 2	Estratégia 3
10^{-5}	TW1	492	504	504
	TW3	96	392	96
	TW5	24	24	20
10^{-4}	TW1	376	504	504
	TW3	52	88	40
	TW5	24	20	8
10^{-3}	TW1	164	492	416
	TW3	24	32	40
	TW5	8	8	8
10^{-2}	TW1	52	492	40
	TW3	16	8	0
	TW5	8	0	0

Tabela 2.10: Número de Significativos, função $v(x)$ -*estratégias 1, 2 e 3*

		Estratégia 1	Estratégia 2	Estratégia 3
10^{-5}	TW1	1,024	1,000	1,000
	TW3	4,923	1,280	4,923
	TW5	16,000	16,000	18,285
10^{-4}	TW1	1,333	1,000	1,000
	TW3	8,533	5,333	10,666
	TW5	16,000	18,285	32,000
10^{-3}	TW1	2,976	1,024	1,207
	TW3	16,000	12,800	10,666
	TW5	32,000	32,000	32,000
10^{-2}	TW1	8,533	1,024	10,666
	TW3	21,333	32,000	64,000
	TW5	32,000	64,000	64,000

Tabela 2.11: Taxas de compressão, função $v(x)$ -estratégias 1, 2 e 3

		Estratégia 1	Estratégia 2	Estratégia 3
10^{-5}	TW1	492	504	504
	TW3	68	124	66
	TW5	56	56	48
10^{-4}	TW1	240	502	494
	TW3	62	66	40
	TW5	50	48	48
10^{-3}	TW1	114	492	382
	TW3	40	40	28
	TW5	44	48	48
10^{-2}	TW1	52	300	30
	TW3	24	28	24
	TW5	34	48	36

Tabela 2.12: Número de Significativos, função $w(x)$ -estratégias 1, 2 e 3

		Estratégia 1	Estratégia 2	Estratégia 3
10^{-5}	TW1	1,024	1,000	1,000
	TW3	6,736	3,878	6,918
	TW5	8,000	8,000	9,142
10^{-4}	TW1	2,064	1,003	1,019
	TW3	7,314	6,918	10,666
	TW5	8,827	9,142	9,142
10^{-3}	TW1	4,196	1,024	1,312
	TW3	10,666	10,666	14,222
	TW5	9,846	9,142	9,142
10^{-2}	TW1	8,533	1,662	13,473
	TW3	16,000	14,222	16,000
	TW5	12,190	9,142	11,636

Tabela 2.13: Taxas de compressão, função $w(x)$ -estratégias 1, 2 e 3

		Estratégia 1	Estratégia 2	Estratégia 3
ε	TW1	12	12	12
	TW3	36	36	36
	TW5	58	58	58

Tabela 2.14: Número de Significativos, função $u(x)$ -estratégias 1, 2 e 3

		Estratégia 1	Estratégia 2	Estratégia 3
ε	TW1	25,600	25,600	25,600
	TW3	11,636	11,636	11,636
	TW5	7,757	7,757	7,757

Tabela 2.15: Taxa de Compressão, função $u(x)$ -estratégias 1, 2 e 3

Os testes com $\varepsilon = 10^{-5}$, $\varepsilon = 10^{-4}$, $\varepsilon = 10^{-3}$ e $\varepsilon = 10^{-2}$ produziram os mesmos resultados para a função $u(x)$, independentemente da ordem da transformação e da estratégia empregada.

O operador de truncamento e, conseqüentemente, as estratégias de compressão utilizadas nesta seção, podem ser aplicados a uma série wavelet, com uma base wavelet qualquer. No entanto a vantagem de se utilizar transformadas wavelets interpolatórias é que cada coeficiente wavelet está associado a apenas um valor pontual, e portanto, a uma única posição na malha uniforme inicial (nível mais refinado, no qual a função foi discretizada). E no caso

da transformada interpolatória ser para valores pontuais, a passagem de coeficientes wavelet para valor pontual da função se dá de forma simplificada. Na próxima seção explora-se esta característica apresentada pelas wavelets interpolatórias e apresenta-se o conceito de representação pontual esparsa.

2.3 Representação Pontual Esparsa

Uma aproximação para o dado inicial u^0 construída a partir da representação multiresolução truncada $\hat{u}_M = (\hat{d}^1, \dots, \hat{d}^L, u^L)$ não necessita utilizar o mesmo número de pontos que o dado inicial u^0 , N_0 , mas apenas a quantidade de significativos da representação truncada e dos valores no nível mais grosseiro.

Pelo algoritmo (1.15) para a transformada inversa, tem-se as seguintes operações para a obtenção de u^0 através de u_M :

$$\begin{aligned} &\text{Para } k = L, L-1, \dots, 1 \\ &\begin{cases} u_{2j}^{k-1} = u_j^k, & 1 \leq j \leq N_k. \\ u_{2j-1}^{k-1} = I(x_{2j-1}^{k-1}, u^k) + d_j^k, & 1 \leq j \leq N_k. \end{cases} \end{aligned}$$

Analisando o algoritmo (1.15) para o primeiro nível de reconstrução e lembrando que $\hat{u}^L = u^L$, observa-se que os valores em posições pares do nível $L-1$ não se alteram quando é utilizado $\hat{u}_M$ ao invés de u_M :

$$\begin{aligned} \tilde{u}_{2j}^{L-1} &= \hat{u}_j^L, \quad 1 \leq j \leq N_k. \\ \tilde{u}_{2j-1}^{L-1} &= I(x_{2j-1}^{L-1}, \hat{u}^L) + \hat{d}_j^L, \quad 1 \leq j \leq N_k. \end{aligned}$$

e os valores em posições ímpares do nível $L-1$ ou são aproximados por interpolação ou necessitam de correção fornecida pelos coeficientes significativos:

$$\begin{aligned} \tilde{u}_{2j}^{L-1} &= u_j^L, \quad 1 \leq j \leq N_k. \\ \tilde{u}_{2j-1}^{L-1} &= \begin{cases} I(x_{2j-1}^{L-1}, u^L) + d_j^L = u_{2j-1}^{L-1}, & \text{se } d_j^L \neq 0 \\ I(x_{2j-1}^{L-1}, u^L), & \text{se } d_j^L = 0. \end{cases} \end{aligned} \quad (2.9)$$

Portanto, para se representar os valores da aproximação no nível $L-1$ são suficientes os valores do nível L e os coeficientes wavelets significativos, uma vez que os demais valores são “bem aproximados” por interpolação a partir dos dados no nível L . Com isso o número de pontos utilizado na reconstrução é $N_L + N_{sL}$, com N_{sL} sendo o número de coeficientes significativos do nível L . Em geral $N_{sL} < N_L$. Utilizando o mesmo raciocínio para os níveis

que faltam ser reconstruídos, pode-se obter uma aproximação $\tilde{u}^0$ com apenas $((N_L + N_{sL}) + N_{sL-1}) + \dots + N_{s2}) + N_{s1} = N_L + |\mathbf{D}_\epsilon|$ pontos.

Esta transformação foi definida por Holmström em [20] e é denominada *transformada wavelet inversa esparsa* (TWIE), pois o algoritmo de reconstrução é de fato utilizado apenas para aqueles pontos que correspondem a coeficientes wavelets significativos. Os demais pontos são diretamente obtidos por interpolação dos valores em níveis mais grosseiros de forma recursiva.

Esta transformação inversa produz uma *Representação Pontual Esparsa* (*RPE*) da função (naturalmente associada a uma malha esparsa). Holmström [21] observa que a *RPE* pode ser obtida sem que a representação multiresolução seja explicitamente formada, como indicado no diagrama (2.33) abaixo. Isto é obtido quando os valores pontuais são diretamente armazenados ao invés dos coeficientes wavelet durante o cálculo das transformações direta (TWD) e inversa (TWIE). Na verdade os coeficientes wavelet são utilizados apenas para decidir se o valor pontual correspondente é ou não incluído na *RPE* (e portanto, se sua posição pertence ou não à malha esparsa). A representação esparsa de uma função não pode ser tratada como uma expansão da função em uma determinada base, ela é apenas a representação da função em um conjunto de pontos especialmente escolhido [21].

Figura 2.33: Diagrama com os operadores envolvidos na RPE

2.3.1 Reconstrução

No diagrama (2.33) ainda existe mais um operador a ser analisado. É o operador de Reconstrução, $\mathcal{R}$, que tem por objetivo obter os valores da função nas posições que não estão na

malha esparsa, mas que pertencem à malha uniforme inicialmente dada.

Os valores da função são obtidos unicamente por interpolação, produzindo assim uma aproximação para u^0 com N_0 pontos. O interpolador para cada ponto é construído de forma diádica, ou seja, os valores utilizados para construir o interpolador estão em um nível mais grosseiro do que o ponto a ser reconstruído.

Em aplicações envolvendo computação gráfica e tratamento de imagem [44], este processo de interpolações sucessivas por nível é denominado de *Algoritmo de Subdivisão*, que corresponde à parte do algoritmo (2.9) para coeficientes d_j^k nulos.

Os algoritmos de subdivisão utilizam interpolações de um nível para outro de forma diádica a partir de uma malha uniforme grossa para calcularem os valores da função nos pontos médios de cada intervalo que são acrescentados na malha a cada novo nível obtido.

Na verdade, só faz sentido aplicar o operador de reconstrução, se o algoritmo (2.9) tiver sido utilizado apenas para os coeficientes significativos, produzindo uma aproximação esparsa. Caso contrário, pode-se pensar em $\mathcal{R}$ como uma espécie de identidade (identidade de malhas), já que a aproximação produzida já possuirá o mesmo número de elementos do que o dado na malha fina.

Uma observação interessante é que no caso no qual todos os coeficientes wavelets são nulos, Deslauriers e Dubuc [9] mostram que a aproximação $\tilde{u}^0$ obtida através de sucessivas interpolações a partir dos valores no nível grosso L converge uniformemente a u^0 , quanto maior for o número de níveis na faturação.

Outra possibilidade para reconstrução de valores na escala mais fina seria utilizar interpoladores construídos de forma 'sequencial', isto é, interpoladores de grau k construídos de $k + 1$ em $k + 1$ pontos da representação esparsa, com variação de um em um ponto da malha esparsa, desta forma diferindo da reconstrução por níveis obtida pelo algoritmo de subdivisão. Neste caso, vários pontos da malha fina seriam obtidos através de um mesmo polinômio interpolador.

Nos exemplos a seguir apresentam-se as representações esparsas para as funções da seção anterior, mas seus valores não serão reconstruídos na malha fina.

Exemplos

As representações esparsas para as funções da seção anterior são construídas apenas para a *estratégia 3*. Nesta sequência de exemplos a tolerância ε foi fixada igual a 10^{-5} . Os gráficos abaixo mostram a posição por nível dos coeficientes wavelets obtidos pela TWP1, TWP3 e TWP5 e à direita está a RPE da função.

Figura 2.34: TWP1 $v(x)$: Coeficientes Significativos e RPE

Figura 2.35: TWP3 $v(x)$: Coeficientes Significativos e RPE

Figura 2.36: TWP5 $v(x)$: Coeficientes Significativos e RPE

Figura 2.37: TWP1 $u(x)$: Coeficientes Significativos e RPE Figura 2.38: TWP3 $u(x)$: Coeficientes Significativos e RPE Figura 2.39: TWP5 $u(x)$: Coeficientes Significativos e RPE

Figura 2.40: TWP1 $w(x)$: Coeficientes Significativos e RPE Figura 2.41: TWP3 $w(x)$: Coeficientes Significativos e RPE Figura 2.42: TWP5 $w(x)$: Coeficientes Significativos e RPE

No caso para médias celulares, Kaibara ([24]) utiliza a idéia de representação pontual esparsa e obtém uma representação em células irregulares. O próximo passo será utilizar a representação pontual esparsa associada à resolução de operadores diferenciais hiperbólicos. Assim questões como evolução temporal da solução sob a estrutura esparsa e a correção da representação esparsa conforme a solução evolui no tempo serão estudadas no Capítulo a seguir.

com o tamanho da mensagem. Quando a mensagem é pequena, a compressão não é necessária. Quando a mensagem é grande, a compressão é necessária. A compressão é necessária quando a mensagem é grande o suficiente para que a compressão seja benéfica. A compressão é necessária quando a mensagem é grande o suficiente para que a compressão seja benéfica. A compressão é necessária quando a mensagem é grande o suficiente para que a compressão seja benéfica.

Capítulo 3

Método Adaptativo

O objetivo deste Capítulo é mostrar como se pode associar esquemas de diferenças finitas à representação esparsa de uma função para produzir métodos que sejam adaptativos com relação à representação espacial do operador diferencial.

Um casamento harmonioso é obtido quando se resolvem equações diferenciais hiperbólicas não lineares por métodos adaptativos, pois a solução destas equações podem desenvolver fortes gradientes ao longo de sua evolução temporal em regiões limitadas do domínio espacial. Os métodos adaptativos, por sua vez, se encarregam de controlar, segundo algum critério, qual região do domínio espacial deve ser refinada e como deve ser este refinamento, de tal modo que as variações da solução sejam bem capturadas pelo método numérico de resolução, fornecendo também uma economia de tempo de processamento e de espaço de armazenamento de dados, principalmente se a estrutura de dados utilizada puder explorar a esparsidade dos dados.

Como foi visto no Capítulo 2, a transformada wavelet interpolatória produz uma representação multiresolução de uma função qualquer e através das estratégias de compressão 2.2 para os coeficientes wavelet, esta representação se torna esparsa. Existe uma concentração de pontos nas regiões onde a função apresenta variações, como foi visto nos exemplos do Capítulo 2.

É exatamente esta propriedade que será utilizada como critério de adaptatividade da malha. Regiões que apresentam variações são representadas por um número maior de pontos, gerando uma maior concentração de pontos na malha associada. Da mesma forma que regiões de maior suavidade serão representadas por menos pontos, e portanto a malha associada também possui um número menor de pontos. O operador diferencial será resolvido a cada instante de tempo na malha obtida como resultado da aplicação de transformada wavelet à solução no instante anterior.

Os problemas de valor inicial (PVI) resolvidos neste Capítulo são da forma:

$$\begin{cases} u_t + f(u)_x = 0 \\ u(x, 0) = u_0(x) \end{cases}, \quad x \in [0, 1], \quad (3.1)$$

com f podendo ser uma função linear ou não de u . E os métodos utilizados serão dados na formulação conservativa, envolvendo apenas valores pontuais da solução inicial.

Inicialmente apresenta-se na Seção 1 uma forma geral para o algoritmo de resolução do PVI, que na verdade, é comum a vários autores: Harten [15], Holmström [21], Kaibara [24], Bihari[5] e Donat[11], por exemplo. Depois, passa-se a discutir cada um dos passos do algoritmo dado.

Neste trabalho utiliza-se uma estratégia de compressão própria, apresentada no Capítulo anterior. Sugere-se uma estrutura esparsa para armazenamento dos dados e propõe-se também uma modificação no algoritmo para obtenção da representação esparsa da função.

3.1 Algoritmo Adaptativo

Em linhas gerais, algoritmos adaptativos que utilizam Wavelets possuem os seguintes passos em comum:

1. Representação Multiresolução: $u_M = Mu^0 = Mu_0$
Transformada Wavelet da condição inicial u_0 .
2. Truncamento: $\hat{u}_M = \text{tr}_{\varepsilon_k}(u_M)$
O operador $\text{tr}_{\varepsilon_k}$ representa alguma estratégia de corte de coeficientes wavelet de u_M , produzindo uma representação truncada $\hat{u}_M$.
3. Extensão e Obtenção da RPE: $\tilde{u}^0 = M^{-1}\mathcal{E}(\hat{u}_M)$
O operador de extensão agrega alguns pontos à representação multiresolução truncada $\hat{u}_M$ e aplicando a transformada wavelet inversa, obtém-se a representação esparsa da função (pontual ou para médias celulares).
4. Evolução: $\tilde{u}^1 = \mathcal{D}(\tilde{u}^0)$
O operador diferencial é discretizado por algum esquema de diferenças finitas e é evoluído no tempo, produzindo uma aproximação para solução em malha esparsa.
5. Reconstrução: $u^1 = \mathcal{R}\tilde{u}^1$
A aproximação para a solução é reconstruída na malha fina para que se possa aplicar novamente a transformada wavelet e reiniciar o processo.
6. Retornar ao passo 1, trocando u^0 por u^1 .

Uma maneira de visualizar os passos do algoritmo é através do diagrama (3.1):

Figura 3.1: Diagrama com os operadores para Evolução Temporal na RPE

3.1.1 Representação Multiresolução

O primeiro passo do algoritmo é aplicar alguma transformada wavelet, vista no Capítulo (1) à função dada como condição inicial do PVI. Esta transformação pode ser para valores pontuais ou pode ser para médias celulares, dependendo da formulação escolhida para o esquema numérico utilizado, caso a formulação seja conservativa. Nas referências [15],[19], [5], [24] são utilizadas transformada para médias celulares e [21], para valores pontuais. Nos exemplos apresentados posteriormente neste Capítulo será utilizada apenas transformação para valores pontuais.

3.1.2 Truncamento

Como visto no Capítulo (2), existem várias estratégias para se descartar coeficientes wavelet que não sejam significativos, isto é, que possuam módulo abaixo de um certo valor.

Em [15], Harten apresenta estimativas de decaimento para os coeficientes wavelet quando a transformação for para médias celulares e com isso obtém uma estratégia de corte de coeficientes, como foi visto na Seção 2.1 do Capítulo 2. Esta mesma estratégia foi utilizada por Kaibara em [24]. Em [21], Holmström utiliza um único valor para corte de coeficientes independentemente do nível em que se encontra o detalhe, agora no contexto de valores pontuais. Neste trabalho, utiliza-se a estratégia híbrida proposta na Seção 2.1 do Capítulo 2.

3.1.3 Extensão e Representação Esparsa

A representação truncada da solução inicial, obtida nos passos 1 e 2 do algoritmo, não leva em consideração o comportamento futuro da solução. É necessário que pontos adicionais sejam agregados à representação multiresolução para que, após a aplicação da transformada

inversa, a representação esparsa resultante permita que o operador seja calculado de forma a transmitir informação para pontos que, em instantes futuros, serão relevantes para a representação da solução.

Na resolução de equações diferenciais hiperbólicas existem dois efeitos que devem ser considerados: a velocidade finita de propagação da solução e a geração de ondas de choque (surgimento de descontinuidades conforme a solução evolui). Estes efeitos desempenham um papel importante na maneira com que são escolhidos os pontos adicionais, denominados pontos de segurança, na representação da função.

Da mesma forma que as estratégias de corte independem do tipo de dado a ser transformado, (média celular, valor pontual, série temporal...), os critérios de inclusão de pontos de segurança também podem ser utilizados para diversos tipos de dados.

Harten em [15] sugere o seguinte algoritmo:

1. Seja a matriz $i(j, k) = 0$, $1 \leq k \leq L$; $1 \leq j \leq N_k = 2^{n_0-k}$ com todas as posições de malha, para cada um dos níveis, igualadas a zero.
2. Para valores de k variando de 1 a L
Para valores de j variando de 1 a N_k

Se $(|d_j^k(v^n)| \leq \varepsilon_k)$, Então $d_j^k(v^n) = 0$

Se não

$i(j-l, k) = 1, -K \leq l \leq K$

Se $(|d_j^k(v^n)| \geq 2^{p+1}\varepsilon_k) \wedge (k > 1)$, Então

$i(2j-1, k-1) = 1$

$i(2j, k-1) = 1$

3. O conjunto das posições significativas para a iteração $n+1$ é dado por : $D^{n+1} = \{(j, k) \mid i(j, k) = 1\}$.

A matriz $[i(j, k)]$ de “zeros e uns” serve como um marcador. Verificar se $(j, k) \in D^{n+1}$, significa verificar se $i(j, k) = 1$. O parâmetro K varia entre 1 e o tamanho do suporte da função de fluxo numérico que ainda é limitada pela condição de CFL. Em [15] K é considerado igual a 1. O parâmetro p corresponde a análise de regularidade e é obtido através das considerações sobre o refinamento da malha. A variável r indica o grau do polinômio interpolador. Segundo Harten, $1 \leq p \leq (r-1)$, sendo que a cota superior para $(r-1)$ é escolhida, levando em conta a perda efetiva de suavidade nos estágios finais de formação do choque.

Neste algoritmo, a matriz $[i(j, k)]$ indica a posição de cada coeficiente significativo d_j^k , e também inclui as posições $i(j-1, k)$ e $i(j+1, k)$ vizinhas na mesma escala. Se o valor do coeficiente wavelet for ainda acima de um certo valor especial, $2^{p+1}\varepsilon_k$, então, $i(2j-1, k-1)$ e $i(2j, k-1)$, vizinhos na escala imediatamente mais fina também serão agregados ao conjunto de significativos, mesmo que os coeficientes wavelet destas posições não o sejam.

Propõe-se, então, a seguinte modificação no algoritmo acima que será aplicada para extensão de representações utilizando valores pontuais. Em termos de alocação de memória é mais eficiente do que o anterior, pois dispensa o uso da matriz $[i(j, k)]$:

1. Seja o vetor $psig(j) = 0, \quad j = 0, \dots, N_o$, com todas as posições da malha fina.
2. Para valores de k variando de 1 a L .
Para valores de j variando de 1 a N_k .

Se $[(|d_j^k(v^n)| \geq \varepsilon_k)]$ Então
 $psig(2^{k-1}(2j - 3)) = 1$
 $psig(2^{k-1}(2j - 1)) = 1$
 $psig(2^{k-1}(2j + 1)) = 1$
 Se $[(k > 1) \wedge (|d_j^k(v^n)| \geq 2^{p+1}\varepsilon_k)]$ Então
 $psig(2^{k-2}(4j - 3)) = 1$
 $psig(2^{k-2}(4j - 1)) = 1$

3. O conjunto das posições significativas para a iteração $n + 1$ é dado por: $D^{n+1} = \{j \mid psig(j) = 1\}$.

Neste algoritmo, a estratégia de inclusão de pontos de segurança é a mesma sugerida por Harten, dois pontos de segurança na mesma escala e dois na escala imediatamente mais refinada, como mostra a figura (3.2) abaixo. No entanto os dados referentes às posições significativas ao longo dos níveis são armazenados em um único vetor, fornecendo portanto uma formulação que utiliza menos espaço para armazenamento dos dados.

Os valores da malha mais refinada são definidos de forma diádica através de subdivisões sucessivas a partir de um nível mais grosseiro de referência, deste modo os valores ímpares de uma determinada escala k são valores pares em toda escala mais refinada construída a partir de subdivisões deste nível k e portanto não correspondem mais a posições associadas a detalhes. Sendo assim, o conjunto das posições ímpares de um nível é disjunto do conjunto das posições ímpares de outro nível qualquer. Isto faz com que não seja necessária a utilização de uma matriz para indicar as posições significativas. Basta um único vetor com o tamanho igual ao da malha mais refinada inicialmente dada.

Os valores significativos agregados na mesma escala, $2^{k-1}(2j - 3)$ e $2^{k-1}(2j + 1)$, como mostra a figura (3.2), são necessários para a transmissão dos efeitos advectivos. Já os agregados na escala mais fina, $2^{k-2}(4j - 3)$ e $2^{k-2}(4j - 1)$, são necessários para a captação do surgimento de novas escalas, ou seja, a formação do choque.

Uma outra possibilidade de inclusão de pontos na representação multiresolução com o objetivo de preservar informações e propagá-las de forma correta, proposta por Plantévin [36] e também utilizada por Kaibara [24], é a inclusão dos pontos pares na escala um nível mais grossa necessários para a construção do preditor do detalhe significativo em questão.

Figura 3.2: Pontos de Segurança

Estes pontos fazem parte do que Plantevin denomina de *condição de Cone*. Ou seja, no caso de um preditor cúbico, pontos da forma $2^k(j-2)$, $2^k(j-1)$, $2^k(j)$ e $2^k(j+1)$ devem ser agregados a representação esparsa da função. Segundo [36] a inclusão destes pontos faz com que a malha esparsa possua uma transição mais suave entre regiões com maior concentração de pontos e regiões com menos pontos.

Uma observação final em relação aos algoritmos que executam as tarefas dos operadores de Extensão e Inversão é que no caso de valores pontuais, os dois operadores podem ser aplicados quase que diretamente aos valores pontuais da função. Os coeficientes wavelets são utilizados apenas para decidir se o ponto é agregado ou não na representação esparsa [21], com isso não há a necessidade da aplicação da transformada inversa, e a extensão se dá diretamente sobre a representação pontual esparsa,[24]. É o que indica a seta pontilhada no diagrama (3.1) apresentado anteriormente.

Estrutura de Dados Esparsa

Em [21], Holmström introduziu o conceito de representação pontual esparsa, mas não utilizou nenhuma estrutura de dados esparsa para armazenar as informações. Na verdade, Holmström utilizou a função intrínseca do MATLAB, *NAN* (*not a number*), para marcar as posições originais dos vetores associadas a coeficientes não significativos, mantendo assim todos os vetores com o mesmo tamanho do vetor inicial dado para a função na malha fina. As operações foram realizadas apenas para as posições que não continham *NAN*. Para isso, todas as posições do vetor foram percorridas e para cada uma foi verificado o conteúdo do vetor através de outra função intrínseca do MATLAB, *isnan*, que devolve verdadeiro, se o seu argumento for *NAN*, e devolve falso, caso contrário.

Em outras linguagens de programação o valor *NAN* pode ser substituído por outro valor que garantidamente não ocorra durante os cálculos e a função *isnan* pode ser substituída por um teste lógico do tipo *se-então*.

Como em vários casos e aplicações pode-se obter taxas de compressão bastante grandes, a utilização de uma estrutura de dados que leve esta esparsidade em consideração proporciona uma economia ainda maior em termos de utilização de memória e tempo de CPU gasto.

Neste sentido é que se propõe a seguinte estrutura de dados, com a qual se fará a evolução do operador diferencial e a partir da qual serão reconstruídos na malha fina os valores da aproximação da solução a cada passo no tempo.

A partir dos valores do vetor *psig*, criam-se três vetores: *ptps*, *dist* e *gsps*. O vetor *ptps* armazena os valores pontuais da função relativos às posições significativas indicadas por *psig*. O vetor *gsps* guarda as posições dos valores pontuais associados aos detalhes significativos em relação à malha mais fina inicial e o vetor *dist* guarda a distância entre dois pontos consecutivos na representação esparsa.

Os próximos passos do algoritmo geral que serão discutidos a seguir já levam em consideração esta trinca de novos vetores como sendo os vetores de entrada para as rotinas que resolvem o operador diferencial. Ou seja, o operador diferencial será resolvido utilizando a estrutura esparsa de armazenamento de dados.

3.1.4 Discretização do operador

Neste Capítulo, a equação diferencial associada ao PVI (3.1) será resolvida numericamente, para efeito de ilustração, por esquemas *upwind*, cuja discretização espacial é da seguinte forma:

$$u_t = \frac{1}{h_i}(f_i(u) - f_{i-1}(u)) \quad (3.2)$$

Existem duas possibilidades para se deduzir os esquemas de resolução da parte espacial do operador diferencial: ou aplica-se um esquema para malha não uniforme (obtido via Taylor, por exemplo) ou, como será feito neste Capítulo, aplica-se um esquema de diferenças finitas para malha uniforme. Neste caso, a escolha do estêncil é feita para que localmente sejam considerados pontos equidistantes do ponto de referência, cuja distância seja a menor possível.

Uma vez que a malha considerada não é uniforme, as distâncias à esquerda e à direita do ponto a ser evoluído não são necessariamente iguais. Na figura (3.3) mostram-se as possibilidades para as distâncias entre três pontos quaisquer da malha. Surge, então, a necessidade de se obter pontos que não pertençam a RPE, denominados pontos fantasmas, para que *localmente* o operador seja discretizado em pontos equidistantes.

Figura 3.3: Posições Possíveis entre Três Pontos .

Obtenção dos Pontos Fantasmas

Os pontos fantasmas são obtidos através de interpolação Lagrangeana de ordem maior ou igual a ordem da Transformação Wavelet utilizada, ou seja, grau do interpolador utilizado como preditor.

Na figura (3.4), o ponto x_{j+1} está mais próximo do ponto x_j do que o ponto x_{j-1} , portanto $d_j > d_{j+1}$. Para se calcular uma aproximação para a derivada primeira em x_j utiliza-se um ponto fantasma, p_g , situado entre x_j e x_{j-1} , de tal modo que a distância entre p_g e x_j seja igual a d_{j+1} . Dessa forma, a derivada upwind, por exemplo, calculada em x_j utilizando p_g como ponto anterior terá a mesma ordem que a derivada calculada em x_{j+1} , cujo ponto anterior é x_j .

Figura 3.4: Uma Interpolação à Esquerda.

Na figura (3.5), apresenta-se um exemplo de malha esparsa e todas as suas decomposições até o nível mais grosseiro possível. Em cada uma das malhas que indicam os níveis intermediários de decomposição são marcadas com círculos as posições de detalhes significativos. A última malha representa a malha esparsa resultante.

Figura 3.5: Malhas Homogêneas e Malha Esparsa

Devido à forma diádica com que as malhas são construídas, se todas as distâncias entre pontos da malha esparsa em qualquer instante de tempo fossem da forma 2^α , com α sendo algum valor inteiro, figura (3.5), o cálculo de pontos fantasmas através de interpolações lineares ficaria extremamente simples, uma vez que utilizando a equação (3.3) seria possível descobrir a quantidade de interpolações necessárias para a construção de um ponto fantasma.

$$nit = \log_2 d_{j+1} - \log_2 d_j, \quad (3.3)$$

E através de k sucessivas interpolações lineares, com $k = |nit|$, obtém-se a seguinte expressão para determinar um ponto fantasma:

$$P_{gk} = \left(\frac{2^k - 1}{2^k} \right) f_j + \left(\frac{1}{2^k} \right) f_{j-1}. \quad (3.4)$$

Figura 3.6: Vários Níveis à Esquerda.

Na verdade, nit pode ser tanto positivo quanto negativo e assim, indicar o “sentido” de interpolação. Quando negativo, indica que a interpolação será à esquerda e quando positivo, a interpolação será feita à direita do ponto de referência, x_j . Naturalmente se nit for igual à zero, não será feita nenhuma interpolação e os pontos x_{j-1} , x_j e x_{j+1} são equidistantes.

E após vários testes numéricos, verificou-se que dependendo do valor utilizado para se descartar os coeficientes wavelet e da ordem da transformada, em algumas iterações surgiam pontos na malha esparsa que não satisfaziam a hipótese da distância a seus vizinhos ser sempre potência de 2. Com isso, abandonou-se a idéia de se procurar expressões simplificadas e passou-se a calcular todo e qualquer fantasma necessário através de interpolação Lagrangiana.

Ordem da Derivada

Considera-se, sem perda de generalidade, o polinômio interpolador de Lagrange de grau 4, $P_L(x)$, que passa pelos pontos x_{i-2} , x_{i-1} , x_i , x_{i+1} e x_{i+2} da malha esparsa.

$$P_L(x) = u_{i-2}\mathcal{L}_{i-2}(x) + u_{i-1}\mathcal{L}_{i-1}(x) + u_i\mathcal{L}_i(x) + u_{i+1}\mathcal{L}_{i+1}(x) + u_{i+2}\mathcal{L}_{i+2}(x),$$

sendo u_{i-2} , u_{i-1} , u_i , u_{i+1} e u_{i+2} os valores da função nos pontos fixados, e os polinômios $\mathcal{L}$ da forma:

$$\mathcal{L}_k(x) = \frac{\prod_{j=i-2, j \neq k}^{i+2} (x - x_j)}{\prod_{j=i-2, j \neq k}^{i+2} (x_k - x_j)}, k = i - 2, \dots, i + 2$$

Assim, para algum $\xi(x) \in [0, 1]$

$$u(x) = P_L(x) + r(x), \quad (3.5)$$

com $r(x)$ sendo o erro de truncamento do polinômio interpolador:

$$r(x) = \frac{u^{(v)}(\xi(x))}{5!} \prod_{j=i-2}^{i+2} (x - x_j).$$

Derivando (3.5), tem-se:

$$u'(x) = P'_L(x) + r'(x), \quad (3.6)$$

sendo

$$r'(x) = \frac{d}{dx} \left(\frac{u^{(v)}(\xi(x))}{5!} \right) \prod_{j=i-2}^{i+2} (x - x_j) + \frac{u^{(v)}(\xi(x))}{5!} \frac{d}{dx} \left(\prod_{j=i-2}^{i+2} (x - x_j) \right).$$

Avaliando (3.6) em algum dos pontos do estêncil, x_i por exemplo, obtém-se:

$$u'(x_i) = P'_L(x_i) + r'(x_i) \quad (3.7)$$

com

$$r'(x_i) = \frac{u^{(v)}(\xi(x_i))}{5!} (h_i + h_{i+1}) h_i h_{i+1} (h_{i+1} + h_{i+2}).$$

e assumindo que $h = h_i = |x_i - x_{i-1}|$ e que $h_{i+1} = \alpha h$, $h_{i+2} = \beta h$ e que $h_{i-1} = \gamma h$, obtém-se a seguinte expressão para o resto

$$r'(x_i) = \frac{u^{(v)}(\xi)}{5!} (h^4 (1 + \gamma) \alpha (\alpha + \beta)).$$

Expressões análogas são obtidas para os demais pontos do estêncil.

Quando os pontos do estêncil forem igualmente espaçados, tem-se aproximações para a derivada primeira $u'(x_j)$ de ordem quatro. O erro para derivada primeira é da forma $\frac{u^{(v)}(\xi(x_j))}{5!} h^4 c_1$, com c_1 sendo uma constante que envolve α , β e γ .

Quando os pontos do estêncil, obtidos pela transformada wavelet, possuírem distâncias maiores do que a distância para a malha mais refinada, estes pontos serão encontrados em regiões de maior regularidade da função. Neste caso, para estas posições da malha esparsa, o valor em módulo de $u^{(v)}(\xi(x_j))$ será bastante pequeno. Tem-se, então, uma espécie de compensação entre a ordem h e o fator que multiplica este h . Portanto, pode-se dizer que a aproximação para a derivada é de ordem variável ao longo do estêncil.

3.1.5 Evolução Temporal

A evolução temporal do operador diferencial associada ao PVI (3.1) será realizada, para efeitos ilustrativos, pelo método explícito de Euler de passo simples:

$$\frac{1}{\delta t_i}(u^{n+1} - u^n) = \frac{1}{h_i}(f_i(u) - f_{i-1}(u)), \quad i = 1, \dots, nsig. \quad n = 1, \dots, nmax; \quad (3.8)$$

sendo *nsig* o número de pontos da malha esparsa e *nmax* o número máximo de iterações no tempo.

Após a evolução em um passo no tempo na malha esparsa obtida pela transformação wavelet, é necessário atualizar a RPE e, conseqüentemente, a malha esparsa, para que esta esteja de acordo com os novos valores da solução obtida. Para isso é preciso aplicar a transformada wavelet novamente à solução. Como a transformada wavelet é aplicada em funções que estejam em um nível fino de resolução, é preciso antes de mais nada, reconstruir a solução na malha mais fina, para que se possa aplicar novamente a transformação wavelet.

3.1.6 Reconstrução

Neste passo do algoritmo são reconstruídos por interpolação todos os valores associados às posições da malha fina que haviam sido descartados pela estratégia de corte do passo anterior. No entanto, agora o interpolador é construído com os novos valores da função já evoluídos no tempo.

Neste trabalho o operador de reconstrução é empregado com o objetivo de possibilitar a atualização da representação pontual esparsa para o próximo instante de tempo, analogamente ao que é feito em [21]. A representação esparsa foi atualizada através das rotinas de subdivisão, apresentadas no Capítulo anterior, para se obter todos os valores da malha fina e com isso retornar o algoritmo ao primeiro passo.

Algoritmos de Subdivisão

Como mencionado no Capítulo 2, Seção 2.3.1, a solução obtida em malha esparsa ainda deve ser “preenchida” até que todas as posições da malha mais finada estejam associadas a valores para a solução evoluída.

Novamente é construído o interpolador para as posições pares seguindo o sentido contrário da transformada Wavelet interpolatória, ou seja, os pontos ímpares são visitados das escalas mais grossas para a mais fina, no entanto o valor obtido pelo preditor só será incluído no vetor associado a malha fina na posição ímpar correspondente, se esta não for associada a detalhe significativo. Caso contrário, o valor da função já pertence a malha esparsa e passará automaticamente para a posição correspondente na malha fina. O algoritmo fica do seguinte modo:

Se $|dj^k| < \epsilon$, Então

$$u(2^{k-1}j - 1) = \text{preditor}(2^k j, 2^k(j-1), 2^k(j-2), 2^k(j+1))$$

Se não

$$u(2^{k-1}j - 1) \in RPE$$

Subdivisão para dados em Malha não uniforme

Em [25] são desenvolvidos algoritmos de subdivisão para dados em malhas não uniformes, cujos pontos são estritamente monótonos, $x_j < x_{j+1}, \forall j$. A principal característica nestes algoritmos é a existência de esquemas distintos de subdivisão para os pontos da malha e para os valores da função. Na verdade, a subdivisão para os valores da função depende da escolha feita para a subdivisão das abcissas.

O esquema de subdivisão que utiliza quatro pontos para o cálculo das abcissas apresentado em [25] é da seguinte forma:

$$\begin{cases} x_{2i}^{(k+1)} = x_i^{(k)} \\ x_{2i+1}^{(k+1)} = \frac{1}{2} \left(x_i^{(k)} + x_{i+1}^{(k)} \right) + \frac{1}{2} s_i^{(k)} \mathcal{G} \left(r_i^{(k)}, R_{i+1}^{(k)} \right), \end{cases}$$

sendo

$$\begin{aligned} s_j^k &= x_{j+1}^k - x_j^k \\ r_j^k &= \frac{s_{j-1}^k}{s_j^k} \\ R_j^k &= \frac{s_j^k}{s_{j-1}^k} = \frac{1}{r_j^k}, \end{aligned}$$

cujas função $\mathcal{G}$ satisfaz as seguintes propriedades:

- se os dados iniciais forem uniformes, estes deverão permanecer uniformes em todos os níveis. $\mathcal{G}(1, 1) = 0$
- $\mathcal{G}$ deve satisfazer uma condição de monotonicidade estrita; ou seja, existe uma constante μ tal que $|\mathcal{G}(r, R)| \leq \mu < 1$. Esta propriedade garante que a malha resultante se torne densa, enquanto as razões $r_i^{(k)}$ e $R_i^{(k)}$ permanecem limitadas [26].
- localmente, a malha x_i^k se torna uniforme [26].

Em [25] são apresentados exemplos para as funções $\mathcal{G}$ baseados em funções racionais interpolatórias cúbicas, que, segundo [8], têm os polinômios de Hermite como caso particular.

No esquema de subdivisão para os valores da função a ser interpolada, a escolha feita para a função $\mathcal{F}$ depende da escolha feita para $\mathcal{G}$.

$$\begin{cases} f_{2j}^{k-1} = f_j^k \\ f_{2j+1}^{k-1} = \frac{f_j^k + f_{j+1}^k}{2} - s_j^k \mathcal{F}(d_j^k, d_{j+1}^k, r_j^k, R_{j+1}^k), \end{cases}$$

sendo que

$$g_j^k = \frac{f_{j+1}^k - f_j^k}{s_j^k}$$

$$d_j^k = g_j^k - g_{j-1}^k$$

Um caso particular de subdivisão não uniforme apresentado em [25] que preserva convexidade, cuja função limite é continuamente diferenciável e que tem ordem de aproximação igual à três, é o seguinte: O fator de correção $\mathcal{F}$ é definido por:

$$\mathcal{F}(x, y, r, R) = \frac{1}{2} \frac{1}{\frac{1+r}{(1+\mathcal{G}(r,R))x} + \frac{1+R}{(1-\mathcal{G}(r,R))y}}$$

e a função $\mathcal{G}$ é considerada $\mathcal{G} = 0$.

No exemplo a seguir, são dados os pontos $\{-2, -1, 0, 1, 1.5, 7, 8, 10, 11\}$ como pontos da malha irregular inicial e os valores da função $(x-5)^2(1 + \frac{x^2}{1000})$ nestes pontos. São realizados quatro níveis de subdivisão não uniforme, segundo o caso particular apresentado acima. Cada subintervalo formado por pontos consecutivos da malha inicial possui o mesmo número de pontos gerados pela subdivisão. A função obtida no final do processo, cujo gráfico é mostrado na figura (3.7), continua sendo dada em malha não uniforme.

Figura 3.7: Reconstrução Não Uniforme

Neste trabalho não foi utilizado este tipo de subdivisão, mas acredita-se que o algoritmo possa ser modificado para que no final do processo de subdivisão, a malha gerada seja uniforme. Desta forma, este algoritmo poderia ser uma alternativa de reconstrução da função na malha fina.

3.2 Exemplos Numéricos

3.2.1 Resolução da equação de advecção

O problema de valor inicial (3.9) será utilizado para todos os testes feitos nesta Seção para a equação de advecção.

$$\begin{cases} u_t + u_x = 0 \\ u(x, 0) = 2 + \text{sen}(2\pi x) \end{cases}, \quad x \in [0, 1] \quad (3.9)$$

Serão fixados os seguintes parâmetros a seguir:

- a escala mais refinada terá $N_0 = 2^{12}$ pontos, portanto $\Delta x = 1/N_0$;
- o número de níveis de transformação é $L = 8$;
- a constante de CFL=0.20 em relação a escala mais fina;
- o valor para corte $\varepsilon = 3,111 \cdot 10^{-4}$ e a estratégia de corte $\epsilon_k = \varepsilon/2^{L-k}$;
- são utilizadas transformada wavelet linear e subdivisão linear e
- integração temporal pelo método de Euler.

Exemplo 1

O objetivo deste primeiro exemplo é ilustrar a necessidade de acrescentar os coeficientes de segurança para que a solução possa evoluir coerentemente.

Os gráficos da figura (3.8) mostram as soluções para o PVI (3.9) no intervalo $[0,1]$ de acordo com os pontos de segurança incluídos na RPE. Cada gráfico contém as soluções de 1000 em 1000 passos no tempo até completar 10000 iterações.

Caso 1 No primeiro gráfico da esquerda para direita de cima para baixo, a evolução temporal foi realizada sem o acréscimo de nenhum tipo de coeficiente de segurança.

Caso 2 O segundo gráfico mostra a evolução da solução quando são acrescentados apenas os coeficientes de segurança da mesma escala que o detalhe significativo.

Caso 3 No terceiro foram acrescentados os coeficientes da mesma escala e os coeficientes na escala imediatamente mais fina do que o coeficiente significativo, como visto na figura (3.2).

Caso 4 Este caso é referente ao acréscimo apenas dos coeficientes da escala mais fina, excluindo os coeficientes da condição de cone e os da mesma escala.

Caso 5 Tem-se, agora, a solução quando são acrescentados os coeficientes da condição de cone e os demais coeficientes de segurança.

Caso 6 E por último são acrescentados apenas os coeficientes referentes à condição de cone.

Figura 3.8: Solução em $[0,1]$ nos casos 1, 2, 3, 4, 5 e 6

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6
0	5.197	5.094	5.094	5.197	5.094	5.197
1000	5.919	4.923	4.923	5.919	4.923	5.902
2000	7.013	4.911	4.911	7.013	4.911	6.942
3000	8.569	4.923	4.923	8.569	4.923	8.497
4000	11.130	4.923	4.923	11.130	4.923	10.893
5000	15.633	4.911	4.911	15.633	4.911	15.170
6000	25.924	4.934	4.934	25.924	4.934	25.283
7000	62.060	4.946	4.946	62.060	4.946	60.235
8000	170.666	4.970	4.970	170.666	4.970	157.538
9000	256.000	4.982	4.982	256.000	4.982	227.555
10000	256.000	4.995	4.995	256.000	4.995	227.555

Tabela 3.1: Taxas de compressão para advecção

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6
0	772	788	788	772	788	772
1000	676	816	816	676	816	678
2000	568	818	818	568	818	574
3000	462	816	816	462	816	466
4000	352	816	816	352	816	360
5000	246	818	818	246	818	254
6000	142	814	814	142	814	146
7000	50	812	812	50	812	52
8000	8	808	808	8	808	10
9000	0	806	806	0	806	2
10000	0	804	804	0	804	2

Tabela 3.2: Número de Coeficientes Significativos para advecção

Apenas por uma análise qualitativa da solução, percebe-se que os casos 2, 3 e 5 fornecem resultados muito melhores do que os casos 1, 4 e 6. Os coeficientes de segurança de mesma escala que o detalhe significativo são indispensáveis para a evolução da solução. Já os demais coeficientes de segurança, no caso da equação de advecção, não trouxeram nenhuma contribuição adicional para a formação da malha esparsa.

As taxas de compressão ao longo das iterações são apresentadas na tabela (3.1), o número coeficientes significativos para cada caso é apresentado na tabela (3.2) e os respectivos erros são dados na tabela (3.3). Nos casos 1 e 4 a taxa aumentou muito em relação à taxa apresentada para a solução inicial. Isto indica um comportamento anômalo para a solução, uma vez que a solução deve ser apenas transportada ao longo do tempo, portanto a taxa

de compressão deveria permanecer da mesma ordem. Já as taxas obtidas nos casos 2, 3 e 5 apresentam o comportamento esperado. Na verdade, não há distinção entre estes casos. Isto significa que os pontos de segurança na mesma escala são efetivamente os que permanecem na malha esparsa final.

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6	m. unif.
1000	0.000937	0.000689	0.000689	0.000937	0.000689	0.000990	0.000133
2000	0.004376	0.001365	0.001365	0.004376	0.001365	0.003663	0.000266
3000	0.012335	0.002033	0.002033	0.012335	0.002033	0.008965	0.000399
4000	0.025357	0.002697	0.002697	0.025357	0.002697	0.018612	0.000532
5000	0.043502	0.003359	0.003359	0.043502	0.003359	0.033281	0.000665
6000	0.066340	0.004019	0.004019	0.066340	0.004019	0.052959	0.000798
7000	0.093099	0.004679	0.004679	0.093099	0.004679	0.077064	0.000931
8000	0.122792	0.005337	0.005337	0.122792	0.005337	0.104734	0.001064
9000	0.154133	0.005994	0.005994	0.154133	0.005994	0.134850	0.001196
10000	0.185193	0.006649	0.006649	0.185193	0.006649	0.165757	0.001329

Tabela 3.3: Erro na norma 2 para equação de advecção

Na verdade, o caso 2 acaba sendo o mais econômico em relação ao número de operações realizadas, pois nos demais casos os pontos adicionais incluídos na extensão da representação esparsa são retirados depois pelo processo de truncamento.

Com respeito a taxas de compressão muito elevadas, observa-se que estas podem não ser um bom indicador de qualidade da solução, como mostram os casos 1 e 4.

Na tabela (3.3) são apresentados os erros calculados na norma 2 para a solução de 1000 em 1000 iterações para os casos 1 a 6 e para a malha uniforme com $2^{12} = 4096$ pontos.

Observa-se ainda que o comportamento para a solução esparsa apresentado neste teste se mantém quando a ordem da transformação é aumentada ou quando se considera uma outra estratégia de corte. A diferença está nas taxas de compressão e consequentemente no erro em relação à malha uniforme.

Exemplo 2

O próximo teste tem como objetivo ilustrar a necessidade de atualização da malha esparsa ao longo do tempo. A tabela (3.4) mostra a variação do erro ao longo de 10.000 iterações quando a malha esparsa é corrigida a cada passo no tempo ($\text{adpt}=1$) e quando a malha esparsa permanece igual a malha esparsa da condição inicial ($\text{adpt}=0$) durante todas as iterações. Diferentes valores de corte foram utilizados na resolução do PVI (3.9).

Observa-se que, independentemente do valor de corte considerado, a solução cuja malha esparsa não é corrigida apresenta maior erro em relação à solução cuja malha é atualizada a cada iteração.

iteração	$2, 111.10^{-4}$	$3, 111.10^{-4}$	$5, 111.10^{-4}$	$8, 111.10^{-4}$	$1, 11.10^{-4}$
0	0.000477	0.000782	0.001078	0.001734	0.002315
1000	0.001217	0.001863	0.002518	0.004072	0.005173
2000	0.002203	0.003258	0.004386	0.007101	0.008755
4000	0.003328	0.004856	0.006537	0.010600	0.012874
5000	0.004484	0.006518	0.008781	0.014259	0.017204
6000	0.005572	0.008110	0.010930	0.017773	0.021406
7000	0.006511	0.009521	0.012835	0.020894	0.025198
8000	0.007259	0.010683	0.014406	0.023471	0.028394
9000	0.007814	0.011596	0.015630	0.025485	0.030963
10000	0.008225	0.012322	0.016596	0.027072	0.033066
iteração	$2, 111.10^{-4}$	$3, 111.10^{-4}$	$5, 111.10^{-4}$	$8, 111.10^{-4}$	$1, 11.10^{-4}$
1	0.000389	0.000689	0.000942	0.001543	0.002064
1000	0.000762	0.001365	0.001832	0.003022	0.004003
2000	0.001129	0.002033	0.002703	0.004473	0.005872
3000	0.001494	0.002697	0.003561	0.005911	0.007705
4000	0.001854	0.003359	0.004405	0.007338	0.009465
5000	0.002212	0.004019	0.005235	0.008760	0.011237
6000	0.002568	0.004679	0.006056	0.010176	0.012940
7000	0.002922	0.005337	0.006866	0.011611	0.014679
8000	0.003273	0.005994	0.007670	0.012995	0.016348
9000	0.003621	0.006649	0.008469	0.014426	0.018061

Tabela 3.4: Erro na norma 2 para equação de advecção: solução com $\text{adpt}=0$ acima e com $\text{adpt}=1$ abaixo

Na figura (3.9) o primeiro gráfico da esquerda para direita mostra as soluções na iteração 10000 para cada um dos valores de corte no caso ($\text{adpt}=0$). No gráfico que segue, são apresentadas para $\varepsilon = 2, 111.10^{-4}$ as comparações entre a solução exata, a solução evoluída sobre a malha esparsa sem correção ($\text{adpt}=0$) e a solução cuja malha foi corrigida a cada passo no tempo ($\text{adpt}=1$). Na figura (3.10) são feitas as comparações para $3, 111.10^{-4}$, $\varepsilon = 5, 111.10^{-4}$, $8, 111.10^{-4}$ e $1, 11.10^{-4}$. Para que se perceba o quanto a solução para $\text{adpt}=0$ ficou mais dissipada do que a solução para $\text{adpt}=1$ e a exata, os gráficos estão com variações menores para x e para y .

A soluções cujas malhas esparsas iniciais foram dadas com maior número de coeficientes significativos apresentaram uma dissipação menor ao longo das 10.000 iterações. Na tabela (3.5) são apresentados a quantidade de coeficientes significativos obtido para cada valor de

ε considerado.

iteração	$2,111 \cdot 10^{-4}$	$3,111 \cdot 10^{-4}$	$5,111 \cdot 10^{-4}$	$8,111 \cdot 10^{-4}$	$1,111 \cdot 10^{-3}$
adpt=0	1404	804	616	368	284
adpt=1					
1000	1464	816	656	384	310
2000	1478	818	662	386	314
3000	1484	816	670	386	316
4000	1488	816	668	388	316
5000	1488	818	672	388	322
6000	1490	814	670	384	318
7000	1492	812	672	382	320
8000	1488	808	670	378	318
9000	1490	806	672	382	322
10000	1490	804	674	378	318

Tabela 3.5: Número de Coeficientes Significativos para adpt=0 e adpt=1

Figura 3.9: Comparação entre soluções em malhas esparsas distintas

Figura 3.10: Comparação entre soluções em malhas esparsas distintas

3.2.2 Resolução da equação de Burgers

Os testes feitos para a equação de advecção serão repetidos para a equação de Burgers dada abaixo, com a mesma condição inicial e com os mesmos parâmetros fixados inicialmente: $\Delta x = 1/N_0 = 1/2^{12}$, $cfl = 0.20$, $\Delta t = (cfl\Delta x)$

$$\begin{cases} u_t + \left(\frac{u^2}{2}\right)_x = 0 \\ u(x, 0) = 2 + \sin(2\pi x), \end{cases} \quad x \in [0, 1] \quad (3.10)$$

Exemplo 1

Novamente comparam-se qualitativamente as soluções da equação de Burgers ao longo de 10000 iterações, variando-se os coeficientes de segurança considerados na rotina que estende a representação esparsa em cada iteração. Os gráficos da figura (3.11) mostram as

soluções do PVI (3.10) para as diferentes possibilidades de inclusão de pontos de segurança. As soluções são apresentadas nos gráficos de 2000 em 2000 iterações.

Caso 1 No primeiro gráfico não é incluído nenhum coeficiente de segurança, apenas o detalhe significativo é considerado na RPE.

Caso 2 No segundo gráfico são mostradas as soluções evoluídas quando são incluídos coeficientes de segurança da mesma escala que os significativos.

Caso 3 No terceiro gráfico são incluídos os coeficientes na mesma escala e na escala mais fina.

Caso 4 Neste caso a solução evolui com uma RPE que apenas inclui pontos de segurança da escala mais fina, além dos coeficientes significativos.

Caso 5 Neste caso são agregados todos os coeficientes, inclusive os coeficientes dados pela condição de cone.

Caso 6 São agregados apenas os coeficientes referentes à condição de cone.

As taxas de compressão obtidas no teste são dadas pela tabela (3.6). No caso 1, a ausência de pontos de segurança causa uma grande variação na taxa de compressão. A compressão exagerada faz com que a dissipação numérica seja ainda maior do que normalmente ocorre num método de primeira ordem. No caso 6 ocorre a mesma dissipação que provoca a deterioração da solução, uma vez que os coeficientes da condição de cone para o caso da transformada linear são apenas os dois vizinhos do detalhe na escala mais grossa. No caso 2 a solução fica bem representada até a formação completa do choque, uma vez que os pontos de segurança da mesma escala são os responsáveis pela evolução da solução. Após a formação do choque, a solução fica um pouco dissipada devido a falta dos coeficientes de segurança na escala mais fina do que o detalhe, que são exatamente os responsáveis pela captura de variações da solução em diferentes escalas.

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6
0	5.197	5.094	5.094	5.197	5.094	5.197
2000	14.371	6.400	6.400	14.075	6.360	13.653
4000	54.613	11.040	6.470	8.714	6.380	39.384
6000	132.129	29.681	28.054	33.851	26.947	65.015
8000	141.241	56.888	55.351	240.941	53.894	227.555
10000	256.000	68.266	69.423	256.000	64.000	256.000

Tabela 3.6: Taxas de compressão para cada caso para Burgers

O caso 3 representou bem a solução tanto antes da formação do choque quanto depois, considerando o menor número possível de coeficientes de segurança. O caso 5 apresentou

o melhor resultado entre taxa de compressão e erro, conseguindo representar bem a região após a formação do choque, no entanto é computacionalmente mais caro do que o caso 3, pois precisa verificar a inclusão de uma quantidade maior de pontos na RPE. No caso 4, o acréscimo apenas dos coeficientes na escala mais fina na RPE não foi suficiente para possibilitar a evolução da solução, causando também uma dissipação exagerada.

Figura 3.11: Evolução temporal da solução em $[0,1]$: casos 1 a 6

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6
0	772	788	788	772	788	772
2000	269	624	624	275	628	284
4000	59	355	617	454	626	88
6000	15	122	130	105	136	47
8000	13	56	58	1	60	2
10000	0	44	43	0	48	0

Tabela 3.7: Número de Coeficientes Significativos para cada caso

iteração	caso 1	caso2	caso3	caso 4	caso 5	caso 6	m.unif.
2000	0.065698	0.004704	0.004704	0.065296	0.004639	0.055652	0.000545
4000	0.382694	0.030470	0.021150	0.226032	0.021185	0.290271	0.007940
6000	0.475988	0.026045	0.012301	0.230698	0.012144	0.299420	0.007425
8000	0.391030	0.028291	0.020869	0.262058	0.019838	0.270380	0.007770
10000	0.349949	0.030164	0.023543	0.256390	0.021581	0.259117	0.007981

Tabela 3.8: Erro para cada um dos 6 casos para equação de Burgers

Os números de coeficientes significativos que foram agregados à malha grossa estão apresentados na tabela (3.7) e os erros para cada uma das soluções apresentadas estão na tabela (3.8). O erro, na norma 2, foi calculado em relação à solução calculada na malha uniforme com 2^{16} pontos.

Exemplo 2

No próximo exemplo são comparadas as soluções obtidas quando a representação pontual esparsa é reconstruída a cada passo no tempo e quando a evolução é feita sobre a mesma malha esparsa obtida para a condição inicial. Serão mantidos os mesmos parâmetros fixados inicialmente: $\Delta x = 1/N_0 = 1/2^{12}$, $cfl = 0.20$, $\Delta t = (cfl \Delta x)$. Nos gráficos apresentados na figura (3.12) pode-se observar que a evolução sem reconstrução da RPE ficou mais dissipada em relação à solução para a qual a reconstrução foi feita a cada passo do tempo, ao longo das 10000 iterações realizadas. Porém, tanto para a equação de advecção quanto para a equação de Burgers, a dissipação que ocorre não chega a destruir a solução. Isto serve como um indicativo de que a malha esparsa possa ser mantida inalterada para mais de uma iteração temporal, sem causar prejuízos para a evolução correta da solução.

A seguir apresentam-se a tabela (3.9) com o número de coeficientes significativos e a tabela (3.10) com erros a cada 2000 iterações, ambas para valores de corte iguais a 2, 111.10^{-4} , 3, 111.10^{-4} , 5, 111.10^{-4} e 8, 111.10^{-4} . Os gráficos da figura (3.12) indicam quanto a solução

obtida sem correção de malha ficou dissipada quando comparada com a solução obtida em malha uniforme com 2^{16} pontos.

iteração	$2, 111.10^{-4}$	$3, 111.10^{-4}$	$5, 111.10^{-4}$	$8, 111.10^{-4}$
adpt=0	1388	788	600	352
adpt=1				
2000	1188	624	518	275
4000	825	617	432	321
6000	213	130	75	54
8000	86	58	45	42
10000	58	43	41	41

Tabela 3.9: Número de Coeficientes Significativos para adpt=0 e adpt=1

Figura 3.12: Comparação entre a solução Exata e a solução cuja RPE não é atualizada

iteração	$2,111.10^{-4}$	$3,111.10^{-4}$	$5,111.10^{-4}$	$8,111.10^{-4}$
2000	0.005255	0.007706	0.010517	0.017076
4000	0.025071	0.028964	0.043206	0.046147
6000	0.040453	0.049259	0.054842	0.078966
8000	0.063042	0.055240	0.066617	0.084181
10000	0.038257	0.039285	0.046426	0.057641
iteração	$2,111.10^{-4}$	$3,111.10^{-4}$	$5,111.10^{-4}$	$8,111.10^{-4}$
2000	0.002543	0.004704	0.00626	0.010514
4000	0.013779	0.021150	0.02680	0.042136
6000	0.009701	0.012301	0.01646	0.025074
8000	0.011716	0.020869	0.02661	0.042095
10000	0.013668	0.023543	0.02817	0.039484

Tabela 3.10: Erro na norma 2 para equação de advecção: solução com $\text{adpt}=0$ acima e com $\text{adpt}=1$ abaixo

Exemplo 3 : Estratégias de Compressão x Ordem de Transformação

Através dos próximos testes deseja-se mostrar como o tipo de estratégia de corte, o valor de corte considerado e o grau do interpolador utilizado na transformação alteram as taxas de compressão relativas à solução da equação de Burgers. Dependendo dos parâmetros, os efeitos dissipativos podem ser bastante intensificados. Serão considerados os mesmos parâmetros fixados inicialmente: $\Delta x = 1/N_0 = 1/2^{12}$, $cfl = 0.20$, $\Delta t = (cfl \Delta x)$.

As estratégias consideradas são:

1. $\epsilon_k = \frac{\epsilon}{2^k}$
2. $\epsilon_k = \frac{\epsilon}{2^{n_0-k}}$
3. $\epsilon_k = \frac{\epsilon}{2^{L-k}}$
4. $\epsilon_k = \epsilon * 2^k$
5. $\epsilon_k = \epsilon$

A tabela (3.11) mostra o valor máximo do módulo do detalhe por nível de transformação para a condição inicial do PVI (3.10), quando são utilizadas as transformadas linear e cúbica. O nível mais fino é 2^{11} e portanto os níveis para detalhe vão de 2^{10} a 2^3 . Os valores apresentados ilustram as estimativas para as taxas de decaimento dos coeficientes wavelet vistas no Capítulo (1) para um dado inicial suave, seno neste caso.

São apresentados na tabela (3.12) os valores de corte por nível para cada uma das estratégias enumeradas acima. Estes valores independem do grau do interpolador utilizado.

Conforme a solução evolui, há a formação de uma onda de choque que, para cada tipo de transformação, é ‘capturada’ de forma diferente. Assim as estratégias produzem taxas de compressão diferentes e como consequência, soluções com qualidade de representação diferente. Pode-se observar que as soluções para as estratégias 4 e 5 ficaram mais dissipadas do que as soluções nos demais casos.

Nível	TWPT1	TWPT3
10	0.	0.
9	1.4438389372984D-08	1.0258460747536D-13
8	1.1550629386292D-07	3.2607250233241D-12
7	9.2402426421501D-07	1.0436362885002D-10
6	7.3913592033570D-06	3.3388733999828D-09
5	5.9104162637658D-05	1.0674745065664D-07
4	4.7197932149601D-04	3.4035922982234D-06
3	3.7486058335834D-03	1.0735061210632D-04

Tabela 3.11: Módulo máximo do detalhe por tipo de Transformação

Nível	E-1	E-2	E-3	E-4	E-5
10	5.000000D-06	9.7656250D-09	7.81250D-08	2.000D-05	1.0D-05
9	2.500000D-06	1.9531250D-08	1.56250D-07	4.000D-05	1.0D-05
8	1.250000D-06	3.9062500D-08	3.12500D-07	8.000D-05	1.0D-05
7	6.250000D-07	7.8125000D-08	6.25000D-07	1.600D-04	1.0D-05
6	3.125000D-07	1.5625000D-07	1.25000D-06	3.200D-04	1.0D-05
5	1.562500D-07	3.1250000D-07	2.50000D-06	6.400D-04	1.0D-05
4	7.812500D-08	6.2500000D-07	5.00000D-06	1.280D-03	1.0D-05
3	3.906250D-08	1.2500000D-06	1.00000D-05	2.560D-03	1.0D-05

Tabela 3.12: Valores de Corte por nível para cada Estratégia

Preditor Linear x Preditor Cúbico

Na tabela (3.13) são apresentadas as taxas de compressão e o respectivo número de coeficientes significativos para cada estratégia de corte nas iterações 0, 2000, 4000, 6000, 8000 e 10000, quando o preditor na transformação wavelet for considerado linear. No caso do preditor ser cúbico, os resultados para a compressão são apresentados na tabela (3.14) para as mesmas iterações.

iteração	TC-1	TC-2	TC-3	TC-4	TC-5
0	1.236714975	1.000000000	1.003921568	9.846153846	2.473429951
2000	1.980657640	1.000000000	1.046499744	9.941747572	3.820895522
4000	4.231404958	1.004906771	1.279200499	42.666666666	11.19125683
6000	5.752808988	1.031738035	2.017733990	41.79591836	17.35593220
8000	6.942372881	1.219047619	2.719787516	37.23636363	19.32075471
10000	8.497925311	1.509211495	3.141104294	37.92592592	23.27272727
iteração	$ D_\epsilon -1$	$ D_\epsilon -2$	$ D_\epsilon -3$	$ D_\epsilon -4$	$ D_\epsilon -5$
0	1648	2040	2032	200	820
2000	1026	2040	1949	198	528
4000	476	2030	1593	40	175
6000	348	1977	1007	41	110
8000	287	1672	745	47	98
10000	233	1349	644	46	80

Tabela 3.13: Taxas de Compressão e Quantidade de Significativos para cada uma das estratégias - Preditor Linear

iteração	TC-1	TC-2	TC-3	TC-4	TC-5
0	9.84615384	8.827586206	17.066666666	128.0000000	32.00000000
2000	5.38947368	1.039066463	2.22608695	35.92982456	12.19047619
4000	8.03137254	1.476568132	13.65333333	31.50769230	22.50549450
6000	8.71489361	3.878787878	19.14018691	30.56716417	24.67469879
8000	11.90697674	7.728301886	22.75555555	29.68115942	24.97560975
10000	12.04705882	9.266968325	21.33333333	28.05479452	24.09411764
iteração	$ D_\epsilon -1$	$ D_\epsilon -2$	$ D_\epsilon -3$	$ D_\epsilon -4$	$ D_\epsilon -5$
0	200	224	112	8	56
2000	372	1963	912	49	160
4000	247	1379	142	57	83
6000	227	520	99	59	75
8000	164	257	82	61	74
10000	162	213	88	65	77

Tabela 3.14: Taxas de Compressão e Quantidade de Significativos para cada uma das estratégias - Preditor Cúbico

Os gráficos a seguir mostram as soluções para o PVI (3.10) nas iterações para as quais foram dadas as taxas de compressão. Cada gráfico apresenta a solução obtida para cada estratégia fixada, considerando-se o preditor linear ou cúbico. O gráfico ainda mostra a solução calculada na malha fina uniforme com 2^{11} pontos pelo mesmo esquema 'upwind'.

Figura 3.13: Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 1

Figura 3.14: Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 2

Figura 3.15: Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 3

Figura 3.16: Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 4 Figura 3.17: Soluções em $[0,1]$ para Preditores Linear e Cúbico - Estratégia 5

Como é possível observar pelos gráficos e pelas tabelas acima, a estratégia 1 apresenta uma boa taxa de compressão, mas apresenta discrepâncias para o máximo do sinal nos instantes anteriores à formação do choque. A estratégia 2 é mais precisa às custas de uma baixíssima taxa de compressão. A estratégia 3 apresenta uma taxa de compressão moderada para o preditor linear, mas representa bem o sinal em todos os instantes a exceção dos trechos lineares após a formação do choque. As estratégias 4 e 5 são imprecisas.

Já para o preditor cúbico, a estratégia 3 mostrou-se mais eficiente do que as estratégias 1 e 2, pois aumentou significativamente a taxa de compressão, mantendo a qualidade da solução. Em linhas gerais, o preditor cúbico comparado com o linear é mais eficiente, pois fornece taxas de compressão cerca de 6 vezes maiores, a custos computacionais não tão elevados.

3.3 Atualização da Malha Esparsa

Em métodos adaptativos baseados em diferentes técnicas, [4], [13], é muito freqüente que uma configuração de malha seja aproveitada para mais do que um passo no tempo antes de ser atualizada. Isto faz com que o custo computacional caia consideravelmente, assim como o tempo de simulação.

Em alguns métodos a quantidade de passos no tempo com uma mesma malha é controlada através da velocidade de propagação da onda, [4]; em outros, o critério de atualização da configuração da malha é dado pelo resíduo entre soluções em passos do tempo diferentes, [13]; ou ainda, através da distância da solução à fronteira da região mais refinada da malha, [35].

No caso de malhas adaptativas geradas através da aplicação de transformadas wavelet, segundo [20], [21], [24], [15], [5] e [32], a transformada wavelet é aplicada à solução em cada passo do tempo. No entanto, o resultado do exemplo 2 da Seção anterior, apresentado na figura (3.12), sugere que a solução poderia ser evoluída sobre uma mesma malha mais de um passo no tempo. Uma vez que a solução evoluída sobre a malha esparsa sem atualização, apesar de mais dissipativa, ainda manteve as características qualitativas da solução exata após as 10.000 iterações no tempo.

Acredita-se ainda que no contexto da transformada wavelet também seja possível criar um dispositivo capaz de indicar quando a representação wavelet deva ser corrigida, diminuindo desta forma o custo computacional na resolução da equação. Para isso, define-se um parâmetro denominado *adpt* que controla o número de iterações no tempo com uma mesma configuração de malha esparsa. Cabe lembrar que as malhas esparsas geradas pela RPE de uma solução ainda são estendidas de tal modo que pontos de segurança são acrescentados à malha, como visto na Seção 3.1.3.

3.3.1 Exemplos Numéricos

Resolve-se o PVI para a equação de advecção dado por:

$$\begin{aligned} u_t + au_x &= 0, & u &= u(x, t), & x &\in [0, 1], & a &= 1.0, \\ u(x, 0) &= 2 + \text{sen}(2\pi x). \end{aligned} \quad (3.11)$$

São fixados os seguintes parâmetros:

- malha mais refinada com $No = 2^{13}$ pontos;
- transformada wavelet linear com $L=9$ níveis de transformação;
- subdivisão linear para reconstrução dos dados esparsos na malha fina;
- estratégia de corte dada por $\varepsilon_k = \frac{\varepsilon_{\text{corte}}}{2^{L-k}}$, $k = 1, \dots, L$;

- valor de corte $\epsilon_{corte} = 2 \times 10^{-4}$;
- esquema upwind de resolução seguindo algoritmo adaptativo apresentado na Seção 1;
- evolução temporal através do método de Euler passo simples e
- $cfl = 0.2$ em relação a malha mais refinada.

O PVI (3.11) é resolvido até o tempo $T=0,2441$, ou seja, são realizadas 10.000 iterações. Primeiramente a malha é atualizada a cada passo no tempo, o que corresponde a $adpt=1$. A cada $\Delta_1 = 1.000$ passos no tempo, até atingir a iteração 10.000, são medidos:

- o número de coeficientes significativos,
- a taxa de compressão e
- o erro na norma dois em relação a solução exata.

Após este teste, o parâmetro $adpt$ passa a valer 2, significando que a malha wavelet é atualizada de dois em dois passos no tempo. A cada $\Delta_2 = 1.000$ iterações são medidos os mesmos dados. Para $adpt=3$, as medições foram feitas a cada $\Delta_3 = 999$ iterações, assim como para $adpt=9$. Para $adpt=4$ e $adpt=6$, $\Delta_4 = \Delta_6 = 996$. Para $adpt=7$ as medições são feitas a cada $\Delta_7 = 994$ passos e para $adpt=5$, $adpt=8$ e $adpt=10$, o número de passos no tempo para se realizar as medições é novamente de $\Delta_5 = 1.000$. Na tabela (3.15) são mostrados o número de coeficientes significativos para cada um dos instantes medidos e para cada valor do parâmetro $adpt$ fixado. Ao final de cada coluna, é informado o tempo, fornecido pelo comando *time* - *user* gasto para cada simulação, que pode ser usado como uma medida aproximada de custo computacional.

it	1	2	3	4	5	6	7	8	9	10
$0\Delta_i$	1428	1428	1428	1428	1428	1428	1428	1428	1428	1428
$1\Delta_i$	1478	1476	1476	1476	1476	1474	1476	1476	1472	1474
$2\Delta_i$	1488	1488	1488	1486	1486	1486	1484	1486	1484	1482
$3\Delta_i$	1496	1494	1494	1492	1494	1490	1492	1490	1490	1490
$4\Delta_i$	1502	1500	1500	1500	1500	1498	1494	1498	1496	1498
$5\Delta_i$	1504	1502	1500	1502	1500	1500	1500	1496	1496	1498
$6\Delta_i$	1502	1500	1502	1502	1498	1502	1502	1500	1498	1496
$7\Delta_i$	1504	1504	1508	1504	1506	1502	1502	1504	1502	1502
$8\Delta_i$	1504	1504	1502	1502	1502	1502	1504	1500	1500	1498
$9\Delta_i$	1510	1508	1508	1508	1508	1506	1504	1506	1500	1502
$10\Delta_i$	1506	1506	1502	1504	1502	1502	1504	1500	1502	1498
tempo	6:34.66	3:28.97	2:27.7	1:56.5	1:39.35	1:25.6	1:21.85	1:10.2	1:05.3	1:00.3

Tabela 3.15: Número de coeficientes significativos por iteração para $adpt$ de 1 a 10

A diferença entre os tempos de simulação para o caso $adpt=1$, no qual a malha é atualizada a cada instante, e o caso $adpt=10$, no qual a malha é atualizada a cada 10 passos no tempo, é realmente bastante significativa.

Figura 3.18: Número de coeficientes significativos de acordo com o parâmetro $adpt$ nas iterações fixadas para medição

Figura 3.19: Taxas de compressão de acordo com o parâmetro $adpt$ nas iterações fixadas para o teste

Figura 3.20: Erros na norma 2 para $adpt$ de 1 a 10 nos instantes $(i+1)\Delta_i$, $i=0,1,\dots,9$

Nas figuras (3.18), (3.19) e (3.20) são apresentados os gráficos com os coeficientes significativos para cada valor do parâmetro $adpt$, as respectivas taxas de compressão e os erros em relação à solução exata em cada instante de medição. Através do gráfico dos erros, fica claro que a 'qualidade' da solução obtida em $T=0,24411$ é a mesma para cada um dos testes com atualização da malha em instantes diferentes, e ainda o ganho em tempo de simulação é evidente. Calcula-se também o erro obtido em malhas uniformes com $No = 2^{10}$, $No = 2^{11}$, $No = 2^{12}$ e $No = 2^{13}$ pontos. Pela figura (3.21), pode-se verificar que os erros para $adpt=1$ ficam todos limitados pelos erros obtidos para $No = 2^{10}$ e $No = 2^{11}$. Isto indica que o método adaptativo mantém uma taxa de decaimento para o erro coerente com a quantidade de pontos obtida pela estratégia de compressão e o valor de corte escolhidos, ou seja, dependente da distância entre os pontos da malha. Uma comparação direta de tempo de processamento entre resolução em malha uniforme e em malha adaptativa seria injusta, pois o operador é extremamente simples; entretanto, quando o operador é sofisticado, o uso da malha adaptativa é extremamente eficiente, como mostram os exemplos do Capítulo 4.

Na figura (3.22), comparam-se os valores dos erros para os parâmetros $adpt$ de 1 a 10 com as quotas inferior e superior fornecidas pelos erros de $No = 2^{10}$ e $No = 2^{11}$.

A seguir resolve-se aumentar ainda mais a quantidade de iterações com uma mesma malha esparsa e repete-se o mesmo teste computacional, agora para $adpt$ valendo 15, 20, 25, 30, 50 e 100. As medições das taxas de compressão, número de significativos e erro em relação à solução exata na norma dois foram realizadas de $\Delta_{10} = 1.000$ em $\Delta_{10} = 1.000$ iterações para $adpt$ igual a 20, 25, 50 e 100. Para $adpt=15$, $\Delta_{15} = 1.005$ e para $adpt=30$, $\Delta_{30} = 1.020$.

Os resultados para o número de coeficientes significativos estão na tabela (3.16). Ao final de cada coluna estão os tempos dados pelo comando *timex-user* para cada simulação com 10.000 iterações.

Figura 3.21: Erros para malhas uniformes e para malha adaptativa, considerando-se $adpt=1$, nos instantes de medição

Figura 3.22: Quotas inferior e superior para o erro obtido para diferentes valores fixados para o parâmetro $adpt$ nos instantes de medição

it	15	20	25	30	50	100
$0\Delta_i$	1428	1428	1428	1428	1428	1428
$1\Delta_i$	1474	1466	1470	1466	1448	1420
$2\Delta_i$	1484	1472	1480	1476	1444	1390
$3\Delta_i$	1488	1476	1486	1482	1440	1358
$4\Delta_i$	1498	1478	1488	1484	1432	1326
$5\Delta_i$	1494	1480	1492	1486	1426	1292
$6\Delta_i$	1498	1478	1490	1490	1416	1258
$7\Delta_i$	1500	1482	1496	1488	1406	1224
$8\Delta_i$	1496	1478	1492	1484	1392	1188
$9\Delta_i$	1498	1476	1488	1486	1382	1152
$10\Delta_i$	1498	1474	1492	1490	1372	1116
tempo	47.20	40.65	36.72	35.02	28.29	22.82

Tabela 3.16: Número de coeficientes significativos para $adpt=15,20,25,30,50$ e 100

O ganho de tempo é ainda maior, no entanto é necessário garantir que a qualidade da solução esteja preservada, apesar do aumento da taxa de compressão.

Figura 3.23: Número de coeficientes significativos por iteração para $adpt$ de 15 a 100

Nos gráficos apresentados nas figuras (3.23), (3.24) e (3.25) verifica-se que para $adpt = 15, 20, 25$ e 30 o número de significativos e, conseqüentemente, as taxas de compressão estão na mesma faixa dos valores obtidos para $adpt = 1, \dots, 10$. Da mesma forma, os erros obtidos

para estes casos também são da mesma ordem dos erros apresentados na figura (3.20). Já os valores medidos para $adpt$ igual a 50 e 100 não seguem as tendências anteriores.

Figura 3.24: Taxas de compressão para $adpt$ de 15 a 100 nas respectivas iterações de teste

Figura 3.25: Erros para $adpt$ de 15 a 100, nos instantes $(i + 1)\Delta_i$, $i=0,1,\dots,9$

O gráfico da figura (3.26) mostra que no instante de tempo $T=0,2441$, as soluções obtidas

para os diferentes valores de *adpt* fornecem soluções muito próximas, no entanto os valores obtidos para os erros indicam que para *adpt* igual a 50 e 100 existe uma perda de precisão.

Figura 3.26: Soluções em $T=0,2441$ para malha fina e malhas com *adpt* diferentes

Para se poder avaliar esta perda de precisão, aumenta-se a quantidade de iterações de 10.000 para 100.000, ou seja o tempo final da simulação é agora $T=2,44$. Na tabela (3.17) são apresentados o número de coeficientes significativos para cada valor de *adpt* a cada 5000 iterações. Para *adpt* 50 e 100 a quantidade de significativos diminui consideravelmente. Uma vez que a equação resolvida tem a propriedade de apenas transportar a solução ao longo do tempo, esta grande diminuição do número de pontos da malha esparsa pode indicar um aumento na dissipação numérica. O que é realmente confirmado, quando se analisa o gráfico para os erros, figura (3.27), e se comparam os valores obtidos neste caso estudado com as quotas fornecidas para as malhas uniformes com $No = 2^{11}$ e $No = 2^{10}$.

Nas figuras (3.28) e (3.29) são analisados os gráficos para as soluções para cada *adpt* ao final do processo, em $T=2,441$. Percebe-se, então, que as soluções que evoluíram em malhas, cuja atualização foi muito “lenta”, apresentaram uma dissipação maior em relação às demais.

it	10	50	100
0	1428	1428	1428
5000	1498	1426	1292
10000	1498	1372	1116
15000	1500	1318	940
20000	1500	1266	764
25000	1494	1212	740
30000	1498	1164	712
35000	1494	1108	682
40000	1494	1058	656
45000	1488	998	628
50000	1484	946	606
55000	1480	892	582
60000	1474	838	550
65000	1468	828	522
70000	1462	832	498
75000	1460	828	472
80000	1456	822	452
85000	1452	822	450
90000	1452	820	452
95000	1446	822	454
100000	1444	820	452
tempo	10:33.38	3:45.28	2:08.76

Tabela 3.17: Número de significativos para $adpt=10, 50$ e 100 em 100000 iterações

Figura 3.27: Erros para $adpt$ 10, 30 e 100 e erros para malhas uniformes com $N_0 = 2^{10}$ e $N_0 = 2^{11}$ pontos. As 20 medidas foram feitas de 5000 em 5000 passos no tempo ao longo das 100000 iterações

Figura 3.28: Soluções em $T=2,441$ para malha fina e malhas com adpt diferentes

Figura 3.29: Soluções em $T=2,441$ exata, para malha fina $No=2^{10}, 2^{11}, 2^{13}$ com $adpt=10$

Os testes feitos nesta Seção indicam o grande potencial de redução de custo computacional baseado no controle do parâmetro $adpt$ para a atualização da malha esparsa, no entanto o que se faz ainda é um controle empírico deste parâmetro.

Busca-se, portanto, um critério de controle para o parâmetro $adpt$ que esteja relacionado com os coeficientes wavelets, possibilitando a atualização da representação esparsa também de forma adaptativa.

Capítulo 4

Método de Alta Resolução Adaptativo

Pela teoria de aproximações, sabe-se que quanto maior for o estêncil utilizado, maior será a ordem de precisão do interpolador obtido. Isto, se a função a ser interpolada for suave dentro do estêncil.

Os métodos de diferenças finitas tradicionais são baseados em interpolações com estêncil fixo. No entanto, interpolações neste tipo de estêncil com ordem maior ou igual a dois são necessariamente oscilatórias próximo de descontinuidades da função a ser interpolada (ver Capítulo 1 para exemplo). Além disso, tais oscilações ainda não decaem em magnitude, mesmo se a malha for refinada.

Segundo [39], até 1987 existiam duas maneiras para eliminar ou diminuir oscilações causadas pelo método numérico perto de descontinuidades: adicionar um termo de viscosidade artificial, como feito em [41], ou aplicar limitadores de fluxo, como sugerido por [29].

A idéia ENO, proposta em [19] por Harten *et all*, foi apontada em [39] como sendo uma das primeiras tentativas com sucesso de se obter uma interpolação que não necessite de parâmetros dependentes do tamanho da malha, que seja de alta ordem de resolução e ainda que seja essencialmente não oscilatória (a magnitude das oscilações decai com ordem $O((\Delta x)^k)$, sendo k a ordem de precisão do interpolador) para funções suaves por partes; classe, na qual se encontram as soluções genéricas para leis de conservação

$$u_t(x, t) + f_x(u(x, t)) = 0. \quad (4.1)$$

Neste Capítulo apresenta-se na Seção 1 a formulação ENO, baseada em valores pontuais, conforme desenvolvida por Osher em [33], e proposta por Shu em [39] para a discretização espacial de leis de conservação hiperbólicas. Na Seção 2, apresentam-se esquemas de diferenças finitas baseados em aproximações conservativas para a função de fluxo exato. Mostra-se ainda, como são associadas as transformadas wavelets interpolatórias aos esquemas anteriores, como forma de se produzir malhas esparsas que se adaptam de acordo com a evolução temporal da solução. Através da modificação aqui sugerida, o método de alta resolução apresentado

por Shu torna-se, então, um método ‘wavelet-adaptativo’. São descritas ainda formas de evolução temporal baseadas em métodos TVD Runge-Kutta, propostos por Gottlieb e Shu [14]. Na Seção 3 são resolvidos vários exemplos para o problema do tubo de choque. E por fim na Seção 4, estende-se o método de alta resolução wavelet adaptativo para problemas bidimensionais escalares e vetoriais. Resolvem-se problemas escalares bidimensionais e também dois problemas de Riemann para a equação de Euler bidimensional.

4.1 Método de Alta Resolução

Na resolução de leis de conservação hiperbólicas, (4.1), através de esquemas de diferenças finitas, deseja-se obter uma aproximação conservativa para a derivada espacial usando valores pontuais do fluxo exato $f(u(x, t))$, [33].

4.1.1 Aproximação Conservativa Unidimensional

Seja dada uma malha definida pela seguinte partição do intervalo $[a, b]$,

$$a = x_{\frac{1}{2}} < x_{\frac{3}{2}} < \dots < x_{N-\frac{1}{2}} < x_{N+\frac{1}{2}} = b, \quad (4.2)$$

$I_i = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$ é denominada célula, $\Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$ é o tamanho da célula, $x_i = \frac{1}{2}(x_{i+\frac{1}{2}} + x_{i-\frac{1}{2}})$ é o seu centro e $\Delta x = \max |\Delta x_i|$, $1 \leq i \leq N$ é o tamanho máximo de célula.

QUESTÃO: Para este tipo de malha, são dados valores pontuais de uma função $f(x)$, $f_i = f(x_i)$, $i = 1, 2, \dots, N$ nos centros das células e deseja-se encontrar uma *função de fluxo numérico*

$$\hat{f}_{i+\frac{1}{2}} = \hat{f}(f_{i-r}, \dots, f_{i+s}), \quad i = 1, 2, \dots, N, \quad r, s \geq 0, \quad r + s + 1 = k,$$

de tal forma que a diferença de fluxo aproxima a derivada $f'(x)$ com ordem k de precisão em x_i ,

$$\frac{1}{\Delta x_i}(\hat{f}_{i+\frac{1}{2}} - \hat{f}_{i-\frac{1}{2}}) = f'(x_i) + O((\Delta x)^k), \quad i = 1, 2, \dots, N. \quad (4.3)$$

A solução deste problema é essencial na obtenção de esquemas conservativos de ordem alta baseados em valores pontuais (diferenças finitas) ou médias celulares (volumes finitos).

Assume-se a existência de uma função $h(x)$ que dependa apenas do tamanho da malha, Δx , tal que

$$f(x) = \frac{1}{\Delta x} \int_{x-\frac{\Delta x}{2}}^{x+\frac{\Delta x}{2}} h(\xi) d\xi; \quad (4.4)$$

pelo teorema fundamental do cálculo,

$$f'(x) = \frac{1}{\Delta x} [h(x + \frac{\Delta x}{2}) - h(x - \frac{\Delta x}{2})].$$

Assim, a solução do problema de aproximação proposto é obtida considerando-se

$$\hat{f}_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) + O((\Delta x)^k).$$

A questão, agora, é determinar a função $h(x)$. Uma possibilidade seria diretamente via (4.4), o que se torna difícil, pois $h(x)$ está implicitamente definida. Observa-se, então, que a função conhecida, $f(x)$, é a média da função incógnita, $h(x)$, na célula $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$. Com isso, uma alternativa para a obtenção de $h(x)$ é através da construção de uma aproximação polinomial por partes de ordem k , k fixado, dentro de cada célula I_i da partição do domínio $[a, b]$,

$$p_i(x) = h(x) + O((\Delta x)^k), \quad x \in I_i, \quad i = 1, 2, \dots, N$$

Apenas para não sobrecarregar a notação, omite-se o subíndice i no polinômio $p_i(x)$ acima.

A construção de $p(x)$ é feita como em [18], utilizando os valores da primitiva de $h(x)$.

4.1.2 Reconstrução via Primitiva

Seja $H(x) = \int_0^x h(\xi) d\xi$ a primitiva da função $h(x)$ em $[0, 1]$. Assim, $H(x)$ é *exatamente* determinada em cada interface $x_{i-\frac{1}{2}}$ e $x_{i+\frac{1}{2}}$ das células I_i , $i = 1, \dots, N$,

$$H(x_{i+\frac{1}{2}}) = \sum_{j=0}^i \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} h(\xi) d\xi = \sum_{j=0}^i \Delta x_j f_j, \quad (4.5)$$

através dos valores pontuais da $f(x)$, conhecidos nos centros x_i de cada uma das células.

Fixadas a célula I_i e a ordem k de aproximação, é escolhido um estêncil com r células à esquerda, s à direita e a própria I_i , com $r + s + 1 = k$, então, nestas condições, existe um único polinômio de grau no máximo k , denotado por $P(x)$, que interpola $H(x)$ nos $k + 1$ pontos $x_{i-r-\frac{1}{2}}, \dots, x_{i+s+\frac{1}{2}}$, fronteiras das células escolhidas para o estêncil.

Seja o polinômio $p(x) = P'(x)$, então, uma vez que $P(x)$ interpola $H(x)$ nas células do estêncil, tem-se:

$$\frac{1}{\Delta x_j} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} p(\xi) d\xi = \frac{1}{\Delta x_j} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} P'(\xi) d\xi = \frac{1}{\Delta x_j} (P(x_{i+\frac{1}{2}}) - P(x_{i-\frac{1}{2}})) = \frac{1}{\Delta x_j} (H(x_{i+\frac{1}{2}}) - H(x_{i-\frac{1}{2}})),$$

Como $H(x)$ é a primitiva de $h(x)$ e usando (4.4), tem-se ainda:

$$\frac{1}{\Delta x_j} (H(x_{i+\frac{1}{2}}) - H(x_{i-\frac{1}{2}})) = \frac{1}{\Delta x_j} \left(\int_0^{x_{j+\frac{1}{2}}} h(\xi) d\xi - \int_0^{x_{j-\frac{1}{2}}} h(\xi) d\xi \right) = \frac{1}{\Delta x_j} \int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} h(\xi) d\xi = f(x_j),$$

para $j = i - r, \dots, i + s$.

Logo, $p(x)$ é a aproximação de ordem k para a função $h(x)$ que se desejava, fornecendo, ainda, para cada célula I_i , aproximações para os valores de $h(x)$ em $x_{i-\frac{1}{2}}$ e em $x_{i+\frac{1}{2}}$:

$$h_{i+\frac{1}{2}}^- = p(x_{i+\frac{1}{2}}), \quad h_{i-\frac{1}{2}}^+ = p(x_{i-\frac{1}{2}}), \quad i = 1, 2, \dots, N$$

e portanto

$$h_{i+\frac{1}{2}}^- = h(x_{i+\frac{1}{2}}) + O((\Delta x)^k), \quad h_{i-\frac{1}{2}}^+ = h(x_{i-\frac{1}{2}}) + O((\Delta x)^k), \quad i = 1, 2, \dots, N.$$

4.1.3 Aproximação ENO para Primitiva

Durante o processo de reconstrução via primitiva, a forma com que se escolhe o estêncil para interpolação de $H(x)$ exerce um papel fundamental para que o método seja efetivamente de alta resolução.

Caso a escolha seja independente do ponto x_i , o estêncil considerado será fixo, r e s não variam com i . Para funções globalmente suaves, a melhor aproximação é obtida por interpoladores centrados, $r = s - 1$ para k ímpar, ou $r = s$ ou $s - 2$ para k par.

Já para o caso de funções suaves por partes, as aproximações com um estêncil fixo não serão adequadas próximo dos pontos de descontinuidade da solução. Isto motiva a idéia de estêncil 'adaptativo', ou seja, a quantidade de pontos à esquerda r e à direita s de I_i varia com a localização de x_i , $r = r(i)$ e $s = s(i)$, na tentativa de, sempre que possível, excluir do estêncil, células que contenham as descontinuidades.

No Capítulo 1, Seção 1.1.2 foram apresentados dois procedimentos para determinação do primeiro ponto do estêncil, um de forma hierárquica e o outro de forma não hierárquica.

Com isso, considera-se a formulação de Newton para o polinômio interpolador $P(x)$, cujos coeficientes envolvem as diferenças divididas da primitiva $H(x)$, escolhidas de acordo com a busca feita para a determinação do estêncil:

$$P(x) = \sum_{j=0}^k H[x_{i-r(j)-\frac{1}{2}}, \dots, x_{i+s(j)+\frac{1}{2}}] \prod_{m=0}^{j-1} (x - x_{i-r(j)+m-\frac{1}{2}}).$$

Derivando-se a expressão acima, obtém-se o polinômio $p(x)$ procurado:

$$p(x) = \sum_{j=1}^k H[x_{i-r(j)-\frac{1}{2}}, \dots, x_{i+s(j)+\frac{1}{2}}] \sum_{m=0}^{j-1} \prod_{l=0, l \neq m}^{j-1} (x - x_{i-r(j)+l-\frac{1}{2}}). \quad (4.6)$$

Na prática, para cada célula, apenas o valor do polinômio $p(x)$ é construído nos dois pontos de interesse, $x_{i-\frac{1}{2}}$ e $x_{i+\frac{1}{2}}$. A obtenção de $p(x)$ também pode ser feita de forma hierárquica. Conforme cada ponto é agregado ao estêncil, seu coeficiente correspondente é agregado à formulação de Newton (4.6).

A seguir, são colocados em forma de algoritmo os passos necessários para a obtenção do fluxo numérico via interpolação ENO:

Algoritmo 1 - Reconstrução ENO

Dados os valores pontuais $f_i = f(x_i)$ da função de fluxo exato,

1. Calcular H , a primitiva da função f nas fronteiras das células, utilizando a relação (4.5).
2. Calcular as diferenças divididas de H de ordem 0 a k para todos os pontos da malha.
3. Para cada célula I_i , começar com o estêncil $\{x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\}$ e utilizar o algoritmo de *Busca Hierárquica* para obter o estêncil do interpolador em I_i .
4. Construir o polinômio $p(x)$ através de (4.6) e obter as aproximações nas fronteiras de I_i .

Em [39], o autor fornece expressões simplificadas para o cálculo de $p(x)$, caso a malha seja uniforme. Em [33] foi proposta uma forma alternativa de se calcular os coeficientes de $p(x)$ utilizando apenas as diferenças divididas da função $f(x)$ ao invés das diferenças de sua primitiva, uma vez que

$$H[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] = \frac{H(x_{i+\frac{1}{2}}) - H(x_{i-\frac{1}{2}})}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} = f_i.$$

Para os exemplos numéricos deste Capítulo foi utilizado o algoritmo de busca hierárquica, o polinômio $p(x)$ foi obtido após o primeiro ponto do estêncil ter sido determinado e seus coeficientes foram calculados através das diferenças divididas da primitiva de f .

4.2 Formulação por Diferenças Finitas

Nesta Seção descrevem-se esquemas de diferenças finitas em malhas não uniformes para resolução de leis de conservação unidimensionais, (4.1), utilizando diretamente uma aproximação conservativa para a derivada espacial, descrita na Seção (4.1.1):

$$\frac{du_i(t)}{dt} = -\frac{1}{\Delta x_i}(\hat{f}_{i+\frac{1}{2}} - \hat{f}_{i-\frac{1}{2}}), \quad (4.7)$$

sendo $u_i(t)$ a aproximação numérica para o valor pontual $u(x_i, t)$ e $\hat{f}_{i+\frac{1}{2}} = \hat{f}(u_{i-r}, \dots, u_{i+s})$ o fluxo numérico que satisfaz as seguintes condições:

- $\hat{f}$ é uma função Lipschitz contínua em todos os seus argumentos;
- $\hat{f}$ é consistente com o fluxo exato f , ou seja, $\hat{f}(u, \dots, u) = f(u)$.

Inicialmente, discute-se a discretização espacial do operador.

4.2.1 Esquema ENO-Roe

O fluxo numérico $\hat{f}_{i+\frac{1}{2}}$ é obtido pela aproximação ENO descrita na Seção 4.1.3 acima. Para estabilidade do esquema, é importante que a ‘direção do vento’ seja considerada na construção do fluxo.

A maneira mais simples e mais barata de se conseguir seguir o vento é calcular a velocidade de Roe:

$$\bar{a}_{i+\frac{1}{2}} = \frac{f(u_{i+1}) - f(u_i)}{u_{i+1} - u_i};$$

- se $\bar{a}_{i+\frac{1}{2}} \geq 0$, então o vento sopra da esquerda para direita e, neste caso, o fluxo numérico $\hat{f}_{i+\frac{1}{2}} = h_{i+\frac{1}{2}}^-$, como mostra a figura abaixo:

Figura 4.1: Vento da Esquerda para Direita

- se $\bar{a}_{i+\frac{1}{2}} < 0$, então o vento sopra da direita para esquerda e, então, o fluxo numérico $\hat{f}_{i+\frac{1}{2}} = h_{i+\frac{1}{2}}^+$, como mostra a figura a seguir:

Figura 4.2: Vento da Direita para Esquerda

Este esquema é denominado ENO-Roe. Segundo Shu, [39], uma desvantagem deste esquema é que ele pode convergir para soluções que violam a condição de entropia da equação dada, como ocorria com o esquema Roe original sem interpolação ENO.

4.2.2 Esquema ENO-Lax-Friedrich

Uma maneira robusta de solucionar o problema apresentado pelo esquema anterior é utilizar uma fatoração global para o fluxo exato, do tipo $f(u) = f^+(u) + f^-(u)$, com $\frac{df^+}{du} \geq 0$ e $\frac{df^-}{du} \leq 0$. É necessário que os fluxos f^+ e f^- tenham tantas derivadas quanto a ordem do esquema.

A fatoração mais simples e suave é a de Lax-Friedrich:

$$f^+ = \frac{1}{2}(f(u) + \alpha u), \quad f^- = \frac{1}{2}(f(u) - \alpha u); \quad (4.8)$$

sendo $\alpha = \max_u |f'(u)|$, u variando sobre todo o domínio.

Agora, aplica-se a reconstrução ENO a cada uma das componentes da fatoração, f^+ e f^- , separadamente, e obtêm-se os fluxos numéricos para estas componentes:

$$\hat{f}_{i+\frac{1}{2}}^+ \quad \text{e} \quad \hat{f}_{i+\frac{1}{2}}^- \quad (4.9)$$

Na verdade,

$$\hat{f}_{i+\frac{1}{2}}^+ = h_{i+\frac{1}{2}}^- \quad \text{e} \quad \hat{f}_{i+\frac{1}{2}}^- = h_{i+\frac{1}{2}}^+ \quad i = 1, 2, \dots, N \quad (4.10)$$

Para obter o fluxo numérico relativo ao fluxo exato, soma-se as duas aproximações para cada fronteira de célula da malha.

$$\hat{f}_{i+\frac{1}{2}} = \hat{f}_{i+\frac{1}{2}}^+ + \hat{f}_{i+\frac{1}{2}}^-, \quad i = 1, 2, \dots, N. \quad (4.11)$$

Em forma de algoritmo, têm-se os seguintes passos para a obtenção do fluxo ENO:

Algoritmo 2 - Fluxo ENO

1. Obter a fatoração Lax-Friedrich para o fluxo exato
 - Cálculo de α
 - Decomposição f^+ , f^- na malha
2. Aplicar reconstrução ENO à componente f^+ do fluxo exato e considerar apenas as aproximações pela esquerda $h_{i+\frac{1}{2}}^-$ para construir a aproximação:

$$\hat{f}_{i+\frac{1}{2}}^+ = h_{i+\frac{1}{2}}^-, \quad i = 1, 2, \dots, N$$

3. Aplicar reconstrução ENO à componente f^- do fluxo exato e considerar apenas as aproximações pela direita $h_{i+\frac{1}{2}}^+$ para construir a aproximação:

$$\hat{f}_{i+\frac{1}{2}}^- = h_{i+\frac{1}{2}}^+, \quad i = 1, 2, \dots, N$$

4. Formar o fluxo numérico final para todos os pontos da 'malha metade':

$$\hat{f}_{i+\frac{1}{2}} = \hat{f}_{i+\frac{1}{2}}^+ + \hat{f}_{i+\frac{1}{2}}^-, \quad i = 1, \dots, N$$

4.2.3 Condições de Contorno

Mostra-se agora como deve ser estendido o domínio computacional para que o cálculo dos fluxos numéricos nas fronteiras possa ser feito também utilizando interpolação ENO.

Figura 4.3: Posições Fantasmas para Cálculo de Fluxo nas Fronteiras

Para que os esquemas possam evoluir corretamente nas fronteiras, são necessários os fluxos numéricos $\hat{f}_{-\frac{1}{2}}$ e $\hat{f}_{\frac{1}{2}}$ para o cálculo do operador em x_0 e os fluxos numéricos $\hat{f}_{N-\frac{1}{2}}$ e $\hat{f}_{N+\frac{1}{2}}$ para x_N .

Para serem obtidas as aproximações que compõe estes fluxos numéricos, são necessários valores pontuais fora do domínio físico do problema, em posições 'fantasmas' do domínio computacional, como mostra a figura (4.2.3).

Para o cálculo de $h_{-\frac{1}{2}}^+$, obrigatoriamente os pontos $x_{-\frac{1}{2}}$ e $x_{\frac{1}{2}}$ devem pertencer ao estêncil do interpolador a ser construído. No caso de interpolação ENO de ordem 3, por exemplo, os pontos $x_{-\frac{3}{2}}$ e $x_{-\frac{5}{2}}$ ainda poderiam ser escolhidos pelo procedimento de busca.

Para o cálculo de $h_{-\frac{1}{2}}^-$, o estêncil deve conter obrigatoriamente os pontos $x_{-\frac{3}{2}}$ e $x_{-\frac{1}{2}}$. Considerando ainda que a ordem do interpolador seja 3, $x_{-\frac{7}{2}}$ e $x_{-\frac{5}{2}}$ são pontos passíveis de serem escolhidos pelo procedimento ENO, como pode ser visto na figura (4.2.3).

Isto implica que a função de fluxo exato, de alguma forma, deve ser estendida para os pontos x_{-4} , x_{-3} , x_{-2} e x_{-1} , no caso da evolução do operador na fronteira esquerda, x_0 , e deve ser estendida nos pontos $x_{N+\frac{1}{2}}$, $x_{N+\frac{3}{2}}$, $x_{N+\frac{5}{2}}$ e $x_{N+\frac{7}{2}}$, no caso da fronteira direita, x_N .

Ou seja, para um interpolador de ordem k , são necessários $k + 1$ pontos à esquerda de x_0 e $k + 1$ pontos à direita de x_N .

A maneira com que a extensão do fluxo exato é feita, depende das condições de contorno impostas ao problema estudado.

No caso particular de condições de contorno periódicas, são necessários apenas k pontos fantasmas em um dos lados, uma vez que fluxos em posições fantasmas estão identificados com fluxos em posições internas ao domínio, pela periodicidade da solução : $\hat{f}_{-\frac{1}{2}} \equiv \hat{f}_{N-\frac{1}{2}}$ e $\hat{f}_{N+\frac{1}{2}} \equiv \hat{f}_{\frac{1}{2}}$.

E no caso de condições de contorno fixas, o fluxo nas posições fantasmas, $x_{-\frac{1}{2}}$ e $x_{N+\frac{1}{2}}$, é nulo.

4.2.4 Malhas Wavelets

Através da RPE (representação pontual esparsa) da condição inicial da lei de conservação (4.1) e da malha esparsa estendida associada a esta representação (vista nos Capítulos 1 e 2), obtém-se a partição (4.2), na qual o operador diferencial será discretizado e evoluído um passo no tempo.

Após esta primeira evolução, o operador diferencial é reconstruído na malha fina fixada *a priori* para que uma nova RPE e uma nova malha esparsa estendida sejam obtidas, agora associadas à solução no novo instante de tempo calculado. E assim a próxima evolução temporal pode ser realizada sob uma malha esparsa adaptada à solução obtida no passo de tempo anterior.

No caso de equações vetoriais multidimensionais, ao invés de se ter uma função incógnita, tem-se um vetor com $n > 1$ componentes a serem determinadas.

Quando se aplica a transformada wavelet à condição inicial destes problemas obtém-se, na verdade, uma RPE para cada componente da condição inicial e consequentemente, tem-se uma malha esparsa associada a cada RPE.

Uma possibilidade seria manter uma malha para cada componente e realizar interpolações toda vez que um valor fosse necessário, mas que não estivesse em alguma das malhas. Esta estratégia exigiria uma estrutura de dados que utilizasse listas ligadas ou árvores para que as interpolações em diferentes níveis de resolução pudessem ser feitas de forma recursiva e local.

A alternativa empregada para o caso foi a construção de uma RPE comum a todas as componentes do vetor. Esta nova representação esparsa, dita 'unificada', é obtida através da união das RPE de cada componente, de tal forma que cada uma das representações esteja definida em todas as posições significativas. Desta forma, todas as componentes do vetor estão definidas nas mesmas posições da malha esparsa onde o operador é resolvido.

A unificação se dá, de fato, através dos coeficientes wavelets. Basta que um coeficiente wavelet, de qualquer uma das componentes, seja significativo, para que sua posição seja incluída na malha esparsa 'unificada', bem como os coeficientes de segurança gerados por este coeficiente, como mostra a figura (4.2.4).

Figura 4.4: Representação Esparsa Unificada

4.2.5 Evolução Temporal

Segundo [39], existem basicamente duas maneiras de se discretizar o tempo: Uma é do tipo Lax-Wendroff usando:

$$u_t = -f_x, \quad u_{tt} = (f'f_x)_x, \quad \dots$$

$$u_j^{n+1} = u_j^n + \Delta t(u_t)_j^n + \frac{\Delta t^2}{2}(u_{tt})_j^n + \dots$$

e então discretizando as derivadas espaciais.

Entre as desvantagens para esta formulação, citadas em [39], estão:

- na dificuldade de programação especialmente no caso multidimensional e/ou problemas com termos forçantes e
- na dificuldade de verificação que o esquema resultante possua variação total decrescente ou limitada (TVD ou TVB), sendo a *variação total*, TV , de uma solução numérica definida por: $TV(u) = \sum_j |u_{j+1} - u_j|$.

A outra maneira de discretização do tempo é usar um método do tipo Runge-Kutta ou multinível.

Euler no tempo

O método de Euler de passo simples no tempo foi a primeira alternativa aplicada para se discretizar a derivada temporal de (4.7). Sendo assim a formulação completa do esquema de diferenças finitas é dada por:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{1}{\Delta x_i}(\hat{f}_{i+\frac{1}{2}}^n - \hat{f}_{i-\frac{1}{2}}^n), \quad (4.12)$$

sendo $\Delta t = \frac{cfl \Delta x_*}{\alpha}$: com $\alpha = \max_{u^n} |f'(u^n)|$, u^n variando em todo o domínio; cfl , a constante de CFL de estabilidade e Δx_* , o espaçamento da malha mais fina.

Em [14], Gottlieb e Shu apresentam um exemplo de PVI para a equação de Burger unidimensional no qual a discretização temporal é via um método de Runge Kutta de segunda ordem, a discretização espacial é de segunda ordem e mesmo assim a solução numérica oscila.

Uma forma de evitar este tipo de instabilidade é através da utilização de uma outra classe de métodos do tipo Runge-Kutta, denominados TVD-Runge Kutta. Os autores afirmam ainda que para resolver problemas hiperbólicos esta discretização temporal é muito mais segura.

TVD-Runge-Kutta

Uma aproximação para o fluxo espacial da equação (4.1) é de *variação total decrescente - TVD*, se a variação total da solução não cresce, $TV(u^{n+1}) \leq TV(u^n)$ (quando utilizado um esquema de passo simples no tempo, $u^{n+1} = u^n + \Delta t L(u^n)$ e com uma restrição para o passo no tempo, $\Delta t \leq \Delta t_1$, Δt_1 dado para a primeira iteração).

O objetivo dos métodos numéricos TVD-Runge-Kutta de alta resolução para discretização temporal é manter a propriedade TVD, $TV(u^{n+1}) \leq TV(u^n)$, enquanto consegue ainda alta ordem de aproximação temporal:

$$u^{n+1} = S(u^n), \quad u(x_j, t^{n+1}) - S(u^n) = O(\Delta x^{r+1}), \quad r = \text{ordem de aproximação},$$

no entanto com uma restrição um pouco diferente para o passo no tempo:

$$\Delta t = c \Delta t_1, \quad (4.13)$$

com c sendo denominado coeficiente de CFL para a discretização de alta ordem no tempo.

Segundo definido em [14], um método Runge-Kutta, cuja forma geral é dada por:

$$\begin{aligned} u^{(i)} &= \sum_{k=0}^{i-1} (\alpha_{ik} u^{(k)} + \Delta t \beta_{ik} \mathcal{L}(u^{(k)})), \quad i = 1, \dots, m \\ u^{(0)} &= u^n, \quad u^{(m)} = u^{n+1}, \end{aligned} \quad (4.14)$$

sendo $\mathcal{L}(u)$ uma aproximação para $-f(u)_x$, é dito TVD Runge-Kutta, com coeficiente CFL (4.13) definido por:

$$c = \min_{i,k} \frac{\alpha_{ik}}{\beta_{ik}} \quad (4.15)$$

sempre que os coeficientes da combinação (4.14) forem $\alpha_{ik} \geq 0$ e $\beta_{ik} \geq 0$.

O método TVD-Runge-Kutta ótimo de segunda ordem, no sentido de possuir coeficiente CFL o maior possível, é dado por:

$$\begin{aligned} u^{(1)} &= u^n + \Delta t \mathcal{L}(u^n) \\ u^{n+1} &= \frac{1}{2} u^n + \frac{1}{2} u^{(1)} + \frac{1}{2} \Delta t \mathcal{L}(u^{(1)}), \end{aligned} \quad (4.16)$$

cujo coeficiente CFL vale um.

Em [14] é demonstrado ainda que o método TVD-Runge-Kutta ótimo de terceira ordem é dado por:

$$\begin{aligned} u^{(1)} &= u^n + \Delta t \mathcal{L}(u^n), \\ u^{(2)} &= \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t \mathcal{L}(u^{(1)}), \\ u^{n+1} &= \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t \mathcal{L}(u^{(2)}), \end{aligned} \quad (4.17)$$

com coeficiente CFL também igual a um.

Segundo [40] e [19] a discretização temporal de alta resolução TVD é útil não só para discretizações espaciais TVD, mas inclusive para discretizações espaciais TVB (de variação limitada) e ENO (essencialmente não oscilatórias). Os métodos TVD-Runge-Kutta mantêm ainda a estabilidade apresentada pelo método de Euler no tempo, em qualquer norma, para discretizações temporais de alta ordem, sob a condição de CFL (4.13). Para leis de conservação escalares multidimensionais, segundo [14], não é possível preservar a propriedade de TVD, no entanto esta formulação permite que estabilidade na norma do máximo para discretizações espaciais de ordem alta sejam obtidas.

Algoritmo 3 - Alta Resolução Adaptativo

Fixados o nível de resolução N_0 mais fino, a quantidade de níveis da transformada wavelet interpolatória L , o grau do polinômio interpolador r e o valor de corte para o processo de truncamento da série wavelet ϵ_c .

Dadas as condições inicial e de contorno para a lei de conservação (4.1):

1.
 - Aplicar transformada wavelet à condição inicial.
 - Obter a RPE ou a RPE unificada, no caso de grandezas vetoriais.
2.
 - Calcular o fluxo exato nas posições da malha esparsa produzida no passo anterior.
 - Calcular o valor de α
 - Obter a fatoração Lax-Friedrich do fluxo exato: f^+ e f^-
 - fixar o valor de Δt para o cálculo da solução no próximo passo de tempo: $\Delta t = \frac{cfl\Delta x_*}{\alpha}$, com Δx_* sendo dado pela escala mais fina fixada *a priori*
3. Calcular o fluxo numérico ENO
 - Para f^+ obter as aproximações ENO h^-
 - Para f^- obter as aproximações ENO h^+
 - $\hat{F}_{i+\frac{1}{2}} = f_{i+\frac{1}{2}}^+ + f_{i+\frac{1}{2}}^-$
4. Evoluir um passo no tempo. No caso do esquema TVD-Runge-Kutta de segunda ordem, este é o passo intermediário.

5. Repetir os passos (2) e (3) agora para a solução intermediária obtida em (4)
6. Evoluir mais um passo do tempo. No caso do esquema TVD Runge-Kutta de segunda ordem, este será o passo que completa uma iteração temporal.
7. Reconstruir a solução na malha fina
8. Retornar ao passo (1), aplicando a transformação à solução obtida no passo anterior e seguir o processo tantos quantos forem os passos no tempo desejados.

A seguir descreve-se o problema do tubo de choque associado à equação de Euler unidimensional e apresentam-se seis configurações iniciais diferentes para as variáveis de estado. Cada um dos PVI's é resolvido de forma adaptativa conforme o algoritmo 3 acima.

4.3 Exemplos Unidimensionais

O problema do Tubo de Choque

O problema físico é um tubo cheio de gás inicialmente dividido em duas partes por uma membrana. O gás tem pressão e densidade mais altas em uma das partes e velocidade nula em todo o tubo.

No instante de tempo $t = 0$ a membrana é removida. Espera-se um movimento na direção de menor pressão. Assumindo que o escoamento seja uniforme ao longo do tubo, existe uma variação apenas em uma direção e a equação de Euler unidimensional se aplica. À medida que o tempo passa, a estrutura deste escoamento envolve três ondas distintas que separam regiões nas quais as variáveis de estado são constantes. Ao longo de duas destas ondas existem descontinuidades em alguma das variáveis de estado.

Uma das ondas, dita 'onda de choque', se propaga dentro da região de pressão mais baixa, na qual a densidade e a pressão saltam para valores muito mais altos e todas as variáveis de estado são descontínuas. Isto é seguido por uma descontinuidade dita 'de contato', ao longo da qual a densidade é novamente descontínua, mas a velocidade e a pressão são constantes.

A terceira onda se move na direção oposta às duas anteriores e tem uma estrutura bem diferente: todas as variáveis de estado são contínuas e existe uma transição suave. Esta onda é chamada onda de rarefação, uma vez que a densidade do gás diminui enquanto a onda passa.

Na figura (4.5) pode-se identificar cada uma das ondas onde ocorrem as descontinuidades nas variáveis de estado. O último gráfico da figura (4.5) contém todas as variáveis de estado em um mesmo sistema de eixos, possibilitando então a identificação da descontinuidade de contato, presente apenas na densidade.

Figura 4.5: Variáveis ρ , u , p

A seguir são apresentadas seis configurações iniciais para o problema do tubo de choque que serão resolvidas pelo algoritmo adaptativo proposto neste capítulo.

Equação de Euler - Caso Vetorial

Considera-se a equação de Euler unidimensional para um gás politrópico, com $\gamma = 1.4$ para o ar.

$$U_t + F(U)_x = 0, \quad (4.18)$$

sendo U o vetor envolvendo as variáveis de estado ρ, u, E , $F(U)$ a função de fluxo exato e P a pressão dada em função da energia total E ,

$$U = \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}, \quad F(U) = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ u(E + P) \end{pmatrix} \quad \text{e} \quad P = (\gamma - 1) \left(E - \frac{1}{2} \rho u^2 \right)$$

Todos os problemas de valor inicial a seguir possuem condições de contorno fixas. Os resultados obtidos nesta Seção através do método wavelet-adaptativo conferem com os

resultados apresentados nas referências [31], [39], como exemplos de validação de diferentes métodos numéricos.

EXEMPLO 1 Problema SOD₁.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (3.0, 0.0, 3.0)^T, & x < 0.5 \\ (1.0, 0.0, 1.0)^T, & x \geq 0.5 \end{cases}$$

EXEMPLO 2 Problema SOD₂.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (1.0, 0.0, 1.0)^T, & x < 0.5 \\ (0.125, 0.0, 0.1)^T, & x \geq 0.5 \end{cases}$$

EXEMPLO 3 Problema de Lax.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (0.445, 0.698, 3.528)^T, & x < 0.5 \\ (0.5, 0.0, 0.571)^T, & x \geq 0.5 \end{cases}$$

EXEMPLO 4 Problema SWHS, dado por Shu e Osher, seno provocando choque.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (3.857143, 2.629369, 10.33333)^T, & x < 0.25 \\ (1 + 0.2\sin(50x - 25), 0.0, 1.0)^T, & x \geq 0.25 \end{cases}$$

EXEMPLO 5 Problema DCI, descontinuidade de contato isolada.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (1.0, 1.0, 0.2)^T, & x < 0.5 \\ (2.0, 1.0, 0.2)^T, & x \geq 0.5 \end{cases}$$

EXEMPLO 6 Problema DEIB, densidade e energia interna baixas.

$$\begin{pmatrix} \rho \\ u \\ P \end{pmatrix} = \begin{cases} (1.0, -2.0, 0.4)^T, & x < 0.5 \\ (1.0, 2.0, 0.4)^T, & x \geq 0.5 \end{cases}$$

Para os testes computacionais são fixados os seguintes parâmetros para cada um dos seis exemplos apresentados:

- a escala fina tem $N_o = 2^{14}$ pontos;

- o número de níveis de transformação, L , é igual a 11;
- o valor para corte $\varepsilon = 1.10^{-4}$ e a estratégia de corte $\epsilon_k = \varepsilon * 2^k$;
- são utilizadas transformada wavelet cúbica e subdivisão cúbica e
- integração temporal pelo método de TVD-Runge-Kutta de 2ª ordem ou de 3ª ordem.

EXEMPLO 1 PVI SOD₁

Na figura (4.6) são apresentadas as soluções no instante $T = 0,048$ para cada uma das componentes do vetor de estado, ρ , u e E e ainda para a pressão p . A constante de $cfl = 0,1$ para este exemplo. Todos os gráficos são para valores da solução por posição da malha.

Figura 4.6: Variáveis ρ , u , E , p : PVI SOD₁

Na figura (4.7) as soluções para as variáveis de estado são apresentadas em dois instantes diferentes, $T = 0,0097$ e $T = 0,048$. Para cada instante são apresentadas as posições significativas em cada um dos níveis de transformação. A solução evolui em uma malha formada pela união das posições significativas de todos os níveis e ainda as posições referentes ao nível mais grosso.

A taxa de compressão e o número de significativos obtidos em cada uma das iterações são apresentados na figura (4.8).

Figura 4.7: RPE em instantes diferentes: PVI SOD₁

Figura 4.8: Número de coeficientes significativos e Taxa de Compressão em cada uma das 8000 iterações realizadas: PVI SOD₁

Conforme a solução evolui no tempo, a quantidade de pontos na malha aumentou em relação à quantidade de pontos necessários para a representação da condição inicial. Mesmo

assim, o número máximo de pontos utilizado para evoluir a solução ainda é 23 vezes menor do que o número de pontos da malha fina.

Na figura (4.9) são comparadas soluções no mesmo instante de tempo $T = 0,048$, mas em níveis de resolução diferentes, ou seja, uma solução tem como malha fina de referência a malha com 2^{14} pontos, a outra tem 2^{10} . Ambas considerando o nível grosso com 2^3 pontos.

Figura 4.9: Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI SOD₁

EXEMPLO 2 PVI SOD₂

Para este exemplo foram considerados: $cfl = 0,6$ e $T = 0,1644$. A figura (4.10) apresenta as variáveis de estado e a pressão em $T = 0,1644$ para cada ponto da malha esparsa de $[0,1]$.

Figura 4.10: Variáveis ρ , u , E , p : PVI SOD₂

Na figura (4.11) são apresentadas as posições significativas para a malha unificada em dois instantes de tempo diferentes: $T = 0,01644$ e $T = 0,1644$. As variáveis ρ , u e p são mostradas em um mesmo gráfico. O que se observa é exatamente a capacidade de adaptatividade da malha esparsa de acordo com a evolução da solução.

A taxa de compressão e a quantidade de pontos significativos na malha esparsa unificada para cada uma das iterações até atingir $T = 0,1644$ são apresentados na figura (4.12). O número de pontos da malha no final do processo é quase 10 vezes menor do que a quantidade de pontos na malha fina.

Figura 4.11: RPE em instantes diferentes: PVI SOD₂

Figura 4.12: Número de coeficientes significativos e Taxa de compressão para cada uma das iterações realizadas: PVI SOD₂

Na figura (4.13) comparam-se as soluções e suas respectivas malhas esparsas unificadas em dois níveis de resolução diferentes. Os gráficos da esquerda são para o caso em que a

malha fina de referência possui 2^{14} pontos. Os gráficos da direita são para uma malha fina de referência com 2^9 pontos. A malha unificada gerada para o nível fino com 2^{14} pontos e 11 níveis de decomposição para a transformada wavelet concentra mais pontos nas vizinhanças dos choques do que para o caso com malha fina 2^9 com 6 níveis de transformação. Consequentemente, as ondas de choque são representadas com menos dissipação, como mostram os gráficos na coluna esquerda da figura (4.13).

Figura 4.13: Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI SOD₁

EXEMPLO 3 PVI LAX

Este exemplo também é resolvido em [40]. Para a resolução deste PVI são fixados $cfl = 0,6$ e $T = 0,16$. A figura (4.14) mostra as soluções das variáveis de estado e da pressão por posição da malha esparsa.

Figura 4.14: Variáveis ρ , u , E , p : PVI LAX

Na figura (4.15) as variáveis de estado ρ , u e a pressão p são mostradas em um mesmo sistema de eixos e logo abaixo segue o gráfico que indica as posições significativas obtidas através da transformada wavelet. À esquerda tem-se a solução em $T = 0,016$ e à direita, em $T = 0,16$. Observa-se claramente a modificação da malha unificada do instante $T = 0,016$ para $T = 0,16$.

O número de pontos na malha nos instantes em torno de $T = 0,16$ fica na faixa de 2300 pontos, e consequentemente a taxa de compressão fica em torno de 7, como mostram os gráficos para a quantidade de significativos e para a taxa de compressão por iteração da figura (4.16).

Os gráficos da figura (4.17) apresentam as variáveis de estado ρ e u e a pressão p no instante $T = 0,16$ para malhas finas de referência com 2^{14} e 2^9 pontos. As transformadas wavelets aplicadas utilizam 11 e 6 níveis de fatoração respectivamente. A representação com mais níveis de resolução consegue capturar mais nitidamente as regiões com grandes variações da solução.

Figura 4.15: RPE em instantes diferentes: PVI LAX

Figura 4.16: Número de coeficientes significativos e Taxa de compressão para cada uma das iterações realizadas: PVI LAX

Figura 4.17: Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI LAX

EXEMPLO 4 PVI SWHS

Para este exemplo foram considerados $cfl = 0.6$ e $T = 0.15$. Os gráficos da figura (4.18) apresentam cada solução do vetor de estado por posição da malha para o instante de tempo fixado. Na figura (4.19) são mostradas as soluções para ρ , u e p em um mesmo sistema de eixos para instantes diferentes e abaixo de cada gráfico para as soluções está o gráfico da respectiva malha esparsa unificada, onde a solução foi calculada. O número de pontos significativos em cada iteração e a respectiva taxa de compressão por iteração são dados na

figura (4.19)

Figura 4.18: Variáveis ρ , u , E , p : PVI SWSH

Na figura (4.21) estão as soluções para densidade, velocidade e pressão e suas respectivas malhas esparsas unificadas no instante $T = 0.15$ para dois níveis de resolução diferentes. Um com 2^{14} pontos no nível fino e o outro com 2^9 pontos. Em ambos os casos a transformada wavelet considera a malha com 2^3 pontos como nível grosso.

Figura 4.19: RPE em instantes diferentes: PVI SWSH

Figura 4.20: Número de coeficientes significativos e Taxa de Compressão para cada uma das iterações realizadas: PVI SWSH

Figura 4.21: Variáveis ρ , u , p -malha fina com 2^9 e 2^{14} pontos: PVI SWSH

EXEMPLO 5 PVI DCI

O exemplo 5 apresenta uma descontinuidade de contato isolada no instante inicial. A solução para cada uma das variáveis do problema no instante $T = 0,146$ é apresentada na figura (4.22). A constante de cfl utilizada é 0,5.

Figura 4.22: Variáveis ρ , u , E , p : PVI DCI.

As malhas adaptativas para este exemplo apresentam-se extremamente bem concentradas na região onde se encontra a descontinuidade, como mostra a figura (4.23). Na verdade, a solução está sendo representada e evoluída em cada instante de tempo sobre malhas muito esparsas e com concentrações de pontos apenas nas vizinhanças dos choques, como indicam as taxas de compressão e o número de significativos por iteração apresentados na figura (4.24). Na figura (4.25) repete-se a comparação entre soluções em níveis de resolução diferentes. Novamente a representação com 2^{14} pontos para o nível fino e com 11 níveis para decomposição wavelet consegue capturar melhor o choque.

Pelo que se percebe na figura (4.25) a malha esparsa para a solução em relação à malha fina com 2^{10} pontos possui até um número maior de pontos nas vizinhanças do choque, no entanto em níveis de resolução mais grossos. E o resultado final para a solução é a existência de mais dissipação do que no caso anterior com mais níveis.

Figura 4.23: RPE em instantes diferentes: PVI DCI.

Figura 4.24: Número de coeficientes significativos e Taxa de Compressão para cada uma das iterações realizadas: PVI DCI.

Figura 4.25: Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI DCI.

EXEMPLO 6 PVI DEIB

Finalmente para o EXEMPLO 6 assumem-se $cfl = 0,6$ e $T = 0,146$. A solução para este PVI é apresentada na figura (4.26) através dos gráficos para cada uma das variáveis de estado e a pressão por posição na malha do intervalo $[0,1]$.

Figura 4.26: Variáveis ρ , u , E , p : PVI DEIB.

Na figura (4.27) são mostradas as soluções para densidade, velocidade e pressão e a respectiva malha esparsa unificada para dois instantes de tempo diferentes: $T = 0,0146$ e $T = 0,146$. Na figura (4.28) são apresentados os números de pontos significativos da malha e as taxas de compressão em cada iteração até ser atingido o instante $T = 0,146$. E por fim são apresentadas as soluções obtidas para o mesmo instante de tempo, $T = 0,146$, mas para duas malhas com número de níveis de resolução diferentes. Os gráficos da esquerda da figura (4.29) são para uma malha fina com 2^{14} pontos e 11 níveis de decomposição. Os gráficos da direita são para uma malha fina com 2^{10} pontos e 7 níveis de resolução. Como nos demais exemplos, as soluções e suas discontinuidades foram melhor representadas nas malhas com maior número de níveis de resolução. Em alguns casos, o número de pontos significativos nas malhas com mais níveis foi ainda menor do que na malha com menos níveis.

Figura 4.27: RPE em instantes diferentes: PVI DEIB.

Figura 4.28: Número de coeficientes significativos e Taxas de Compressão para cada iteração realizada: PVI DEIB.

Figura 4.29: Variáveis ρ , u , p - malha fina com 2^{10} e 2^{14} pontos: PVI DEIB.

Para este exemplo comparam-se os tempos de execução quando o PVI é resolvido em malha fina uniforme pelo método de alta resolução e quando o PVI é resolvido pelo algoritmo adaptativo de alta resolução. A constante de cfl fixada é igual à 0,2. O tempo foi medido através do comando UNIX *time* - *user* para 24, 240 e 2400 iterações no tempo, considerando-se a malha fina de referência com 2^{14} pontos.

iterações	malha esparsa	malha fina
24	4.64	53.93
240	49.71	7:24.03
2400	13:04.18	1:13:46.66

Tabela 4.1: Tempo de Processamento

A seguir passa-se a descrever a extensão do método de alta resolução empregado aqui para problemas bidimensionais. Apresentam-se então alguns exemplos para PVI escalares e problemas de Riemann associados a equação de Euler bidimensional.

4.4 Extensão à Duas Dimensões

As técnicas escolhidas ao longo deste trabalho permitem a extensão natural do algoritmo adaptativo a problemas de leis de conservação hiperbólicas multidimensionais. O problema de Riemann unidimensional foi resolvido adaptativamente com sucesso utilizando a combinação da transformada wavelet interpolatória, para compressão, ao uso de interpoladores ENO de alta resolução na reconstrução dos fluxos.

A extensão dessas escolhas a problemas multidimensionais requer o desenvolvimento de um algoritmo no qual as direções permaneçam desacopladas. O esquema, então, obtido segue a linha dos métodos de fatoração no tempo (*time splitting methods* [43]) que reduzem problemas multidimensionais a uma sequência de problemas unidimensionais por direção. Uma das vantagens deste desacoplamento das direções faz com que o esquema obtido seja paralelizável em cada direção.

Considere o sistema de leis de conservação bidimensionais (4.4), escalares ou vetoriais a seguir:

$$\begin{aligned}
 U_t + F(U)_x + G(U)_y &= 0, \\
 U &= (u_1, u_2, \dots, u_s)^T, \quad s \geq 1 \\
 u_i(x, y, t), \quad i &\leq s, \quad (x, y) \in [a, b] \times [c, d]
 \end{aligned} \tag{4.19}$$

Uma das motivações para a extensão do esquema utilizado foi o método de MacCormack [43], dado pela seguinte formulação em malha uniforme, para o problema (4.4) com fluxos lineares da forma :

$$\begin{aligned}
 \tilde{v}_{l,m}^{n+\frac{1}{2}} &= v_{l,m}^n - A\lambda(v_{l+1,m}^n - v_{l,m}^n) \\
 v_{l,m}^{n+\frac{1}{2}} &= \frac{1}{2}[v_{l,m}^n + \tilde{v}_{l,m}^{n+\frac{1}{2}} - A\lambda(\tilde{v}_{l,m}^n - \tilde{v}_{l-1,m}^n)] \\
 \tilde{v}_{l,m}^{n+1} &= v_{l,m}^{n+\frac{1}{2}} - B\lambda(v_{l,m+1}^{n+\frac{1}{2}} - v_{l,m}^{n+\frac{1}{2}}) \\
 v_{l,m}^{n+1} &= \frac{1}{2}[v_{l,m}^{n+\frac{1}{2}} + \tilde{v}_{l,m}^{n+1} - B\lambda(\tilde{v}_{l,m}^{n+1} - \tilde{v}_{l,m-1}^{n+1})],
 \end{aligned} \tag{4.20}$$

sendo $\lambda = \frac{\Delta t}{\Delta x}$, A e B operadores lineares, $F(v) = Av$ e $G(v) = Bv$.

A mesma idéia de (4.20) de evoluir o operador no tempo através de um passo ‘preditor’ e outro ‘corretor’ para cada uma das direções separadamente também foi utilizada nesta extensão. No entanto, a diferença entre o método (4.20) e o método apresentado nesta seção está na forma com que o operador é discretizado. Agora os fluxos $F(U)$ e $G(U)$ do sistema (4.4) são expressos na formulação conservativa, como feito na seção anterior.

Através da interpolação ENO, utilizada para reconstruir o fluxo numérico, obtém-se novamente um esquema de alta resolução em cada uma das direções. Isto evita o surgimento de um termo dissipativo, como ocorre no esquema (4.20) MacCormack. Com isso a formulação utilizada para os problemas bidimensionais é dada por:

$$\begin{aligned}
 \tilde{v}_{l,m}^{n+\frac{1}{2}} &= v_{l,m}^n - \lambda(\hat{F}(v_{l+\frac{1}{2},m}^n) - \hat{F}(v_{l-\frac{1}{2},m}^n)) \\
 v_{l,m}^{n+\frac{1}{2}} &= \frac{1}{2}[v_{l,m}^n + \tilde{v}_{l,m}^{n+\frac{1}{2}} - \lambda(\hat{F}(\tilde{v}_{l+\frac{1}{2},m}^n) - \hat{F}(\tilde{v}_{l-\frac{1}{2},m}^n))] \\
 \tilde{v}_{l,m}^{n+1} &= v_{l,m}^{n+\frac{1}{2}} - \lambda(\hat{G}(v_{l,m+\frac{1}{2}}^{n+\frac{1}{2}}) - \hat{G}(v_{l,m-\frac{1}{2}}^{n+\frac{1}{2}})) \\
 v_{l,m}^{n+1} &= \frac{1}{2}[v_{l,m}^{n+\frac{1}{2}} + \tilde{v}_{l,m}^{n+1} - \lambda(\hat{G}(\tilde{v}_{l,m+\frac{1}{2}}^{n+1}) - \hat{G}(\tilde{v}_{l,m-\frac{1}{2}}^{n+1}))],
 \end{aligned} \tag{4.21}$$

sendo $\hat{F}$ e $\hat{G}$ os fluxos numéricos obtidos através de interpolação ENO.

Pode-se observar que tanto na direção x quanto na direção y utiliza-se o método de Runge-Kutta de segunda ordem como forma de evoluir no tempo. A figura (4.30) ilustra a fatoração por direção e os passos no tempo realizados no esquema (4.21).

Figura 4.30: Fatoração por Direção, Evolução no Tempo com Passo Fracionário

Inicialmente, para ilustrar o uso da técnica, aplicou-se o princípio para problemas escalares utilizando o método de Euler de passo simples no tempo para cada uma das direções. A formulação resultante é da forma por (4.22):

$$\begin{aligned}
 v_{l,m}^{n+\frac{1}{2}} &= v_{l,m}^n - \lambda(\hat{F}(v_{l+\frac{1}{2},m}^n) - \hat{F}(v_{l-\frac{1}{2},m}^n)) \\
 v_{l,m}^{n+1} &= v_{l,m}^{n+\frac{1}{2}} - \lambda(\hat{G}(v_{l,m+\frac{1}{2}}^{n+\frac{1}{2}}) - \hat{G}(v_{l,m-\frac{1}{2}}^{n+\frac{1}{2}}))
 \end{aligned}
 \tag{4.22}$$

A formulação (4.21) será novamente resolvida de forma adaptativa, uma vez que a evolução temporal será realizada para valores pontuais do operador espacial dados em uma malha esparsa que surge da RPE unificada para as componentes do vetor incógnita, sendo a transformada wavelet aplicada separadamente por direção e por iteração no tempo.

Uma das vantagens da formulação (4.21) é a utilização das transformadas wavelets unidimensionais direta e inversa, exatamente como nos casos unidimensionais. Outra vantagem é a facilidade de extensão a problemas com dimensão maior ainda, desde que se possa realizar a fatoração por direção. O fato da formulação do método envolver apenas valores pontuais também é fundamental, pois a obtenção dos fluxos numéricos via interpolação ENO dispensa decomposições das matrizes jacobianas dos fluxos nas direções de seus autovetores, tarefa complicada e computacionalmente cara.

Em forma de algoritmo o esquema (4.21) é dado por:

Algoritmo

Para cada passo n no tempo, $n = 1, \dots, nmax$:

1. Resolve-se o operador diferencial na direção x para cada uma das posições y_j da malha fina na direção y .
 - Aplica-se transformada wavelet em $u_{i,j}^n$ na direção x e obtém-se a malha esparsa associada a RPE de $u_{i,j}^n$, na qual o operador será evoluído.
 - Resolve-se a primeira parte da fatoração, usando interpolação ENO para o cálculo do fluxo numérico referente a $F(U)$. A evolução temporal pode ser pelo método de passo simples de Euler ou pelo método de Runge Kutta de segunda ordem.
 - Reconstrói-se a solução na malha fina para a linha resolvida, obtendo $u_{i,j}^{n+\frac{1}{2}}$.
2. Resolve-se o operador diferencial na direção y para cada uma das posições x_i da malha fina na direção x , utilizando como condição inicial a solução na malha fina obtida no item anterior.
 - Aplica-se transformada wavelet em $u_{i,j}^{n+\frac{1}{2}}$ na direção y e obtém-se a malha esparsa associada a RPE de $u_{i,j}^{n+\frac{1}{2}}$, na qual o operador será evoluído.
 - Resolve-se a última parte da fatoração, usando interpolação ENO para o cálculo do fluxo numérico referente a $G(U)$. A evolução temporal pode ser pelo método de passo simples de Euler ou pelo método de Runge Kutta de segunda ordem.
 - Reconstrói-se a solução na malha fina para a coluna evoluída no tempo, obtendo a solução no novo instante de tempo $u_{i,j}^{n+1}$.
 - Retorna-se ao primeiro passo, agora consideram-se a solução $u_{i,j}^{n+1}$ como condição inicial. E assim sucessivamente, até chegar no número de iterações máximo

A seguir são apresentadas as soluções para problemas bidimensionais escalares:

4.4.1 Exemplos Escalares

Equação de Advecção

Considera-se a equação de advecção bidimensional.

$$u_t + u_x + u_y = 0,$$

cujas condições iniciais são dadas por:

$$u(x, y, 0) = \begin{cases} 1, & x^2 + y^2 \leq \frac{1}{4} \\ 0, & x^2 + y^2 > \frac{1}{4} \end{cases}, \quad (4.23)$$

com $(x, y) \in [0, 1] \times [0, 1]$.

Equação de Burgers

Considera-se a equação de Burgers bidimensional.

$$u_t + uu_x + uu_y = 0,$$

com condição inicial dada por:

$$u(x, y, 0) = \begin{cases} 1, & x^2 + y^2 \leq \frac{1}{4} \\ 0, & x^2 + y^2 > \frac{1}{4} \end{cases}, \quad (4.24)$$

com $(x, y) \in [0, 1] \times [0, 1]$.

E ainda com condição inicial dada por:

$$u(x, y, 0) = \begin{cases} \frac{1}{2}, & x < 0.5 \text{ e } y < 0.5 \\ -\frac{1}{2}, & x > 0.5 \text{ e } y > 0.5 \\ \frac{1}{4}, & \text{c.c.} \end{cases}, \quad (4.25)$$

com $(x, y) \in [0, 1] \times [0, 1]$.

Equação de Advecção com coeficientes variáveis

Para o problema do cone, proposto em [4], que gira em torno da origem em sentido anti-horário tem-se a seguinte equação :

$$u_t - yu_x + xu_y = 0,$$

cujas condições iniciais são dadas por:

$$u(x, y, 0) = \begin{cases} 1 - 2(4(x - \frac{1}{2})^2 + 6(y - 0)^2) & , \quad 4(x - \frac{1}{2})^2 + 6(y - 0)^2 < \frac{1}{2} \\ 0 & , \quad \text{c.c.} \end{cases}, \quad (4.26)$$

com $(x, y) \in [-1, 1] \times [-1, 1]$.

A seguir seguem as soluções numéricas em vários instantes de tempo para cada um dos exemplos de problemas de valor inicial apresentados acima. Apresentam-se as respectivas malhas esparsas associadas a cada quadro das seqüências. Para os PVI's (4.23) e (4.24) a evolução temporal em cada componente da fatoração por direção foi realizada pelo método de passo simples de Euler. Já para os PVI's (4.25) e (4.26) a evolução temporal foi pelo método de Runge-Kutta de segunda ordem.

Os parâmetros fixados para as simulações de cada um dos exemplos escalares foram:

1. nível fino de resolução na direção x e na direção y: $N_0 = 2^7$
2. quantidade de níveis de fatoração nas duas direções: $L = 4$

3. constante de CFL: $CFL = 0,4$
4. condição de contorno periódica
5. número máximo de iterações: $itmax = 540$
6. número de passos no tempo entre uma solução e outra da animação: 30
7. transformada wavelet cúbica,
8. reconstrução pelo algoritmo de subdivisão de grau 3
9. $\epsilon_{corte} = 0,0001$
10. $\epsilon_k = \frac{\epsilon_{corte}}{2^{L-k}}$

Apenas para o exemplo do cone girando ao redor da origem, tem-se:

1. constante de CFL: $CFL = 0,2$
2. número máximo de iterações: $itmax = 1000$
3. número de passos no tempo entre uma solução e outra da animação: 100

A soluções apresentadas na seqüência (4.31) para a equação de advecção mostram a resolução do operador na direção x da fatoração, apresentada no esquema (4.22). Sendo assim, para cada valor da malha na direção y , resolve-se o operador sobre a RPE dos valores da malha na direção de x . A seqüência (4.32) mostra a configuração da malha esparsa em x resultante para cada quadro apresentado na seqüência (4.31). No segundo passo da fatoração do esquema (4.22), as direções de resolução e fixada são invertidas. Aplica-se, então, a transformada wavelet interpolatória aos valores da solução para cada uma das colunas da malha original, sendo fixados os valores da solução na direção de x . A seqüência com a resolução na direção de y é apresentada na figura (4.33), e as malhas para cada um dos quadros desta seqüência são apresentadas em (4.34). Na resolução via esquema (4.22) sempre a solução será dada em malha fina em uma das direções, enquanto que para a outra direção será obtida sua RPE.

De forma análoga apresentam-se as soluções por direção de fatoração para a equação de Burgers. As figuras (4.35) e (4.36) mostram as soluções e as respectivas malhas para a fatoração na direção x , onde se aplica a transformada wavelet por linha. Já as figuras (4.37) e (4.38) mostram as soluções e as malhas associadas para a fatoração na direção y .

A figura (4.39) mostra uma sobreposição das malhas esparsas obtidas no final de cada iteração do esquema (4.22), quando o operador foi atualizado em cada uma das direções. Na verdade, a evolução do operador não se dá efetivamente sobre estas malhas, mas ao final de cada passo no tempo, os pontos de (4.39) são os significativos para a solução.

Sequência para a Equação de Advecção

Figura 4.31: Equação de Advecção - RPE na direção x

Figura 4.32: Malha Esparsa para a equação de Advecção - direção x

Figura 4.33: Equação de Advecção - RPE na direção y

Figura 4.34: Malha Esparsa para a equação de Advecção - direção y

Sequência para a Equação de Burgers PVI (4.24)

Figura 4.35: Equação de Burger - RPE na direção x

Figura 4.36: Malha Esparsa para a equação de Burgers - direção x

Figura 4.37: Equação de Burgers - RPE na direção y

Figura 4.38: Malha Esparsa para a equação de Burgers - direção y

Figura 4.39: Malha Esparsa para a Equação de Burgers - direções x e y

As seqüências de soluções a seguir foram obtidas pelo esquema (4.21), onde cada direção é resolvida através de dois passos intermediários de tempo. Os gráficos apresentados nas seqüências a seguir foram obtidos sempre na segunda iteração para direção indicada. Na figura (4.40) estão as soluções para o PVI (4.24) para a fatoração na direção x , na qual se aplica a transformada wavelet interpolatória para se obter a RPE da solução, para cada um dos valores de y fixado na malha fina. As malhas esparsas para as soluções apresentadas em (4.40) estão na figura (4.41). Omitem-se as soluções na outra direção da fatoração (4.21). Na figura (4.42), as soluções para o PVI (4.26) são para a fatoração na direção y , na qual se aplica a transformada wavelet. As malhas associadas estão na figura (4.43).

Sequência para a Equação de Burgers - PVI (4.25)

Figura 4.40: Equação de Burgers - RPE na direção x

Figura 4.41: Malha Esparsa para a equação de Burgers - direção x

Rotação de um Cone em torno da Origem

Figura 4.42: Rotação do Cone - RPE na direção y

Figura 4.43: Malha Esparsa para a rotação do Cone - direção y

4.4.2 Exemplo Vetorial - Sistema Hiperbólico

Equação de Euler Bidimensional

A interação de ondas elementares (onda de choque, onda de rarefação ou descontinuidade de contato) para sistemas hiperbólicos de leis de conservação, segundo [38], é o mecanismo chave na determinação de propriedades qualitativas de soluções para problemas de valores iniciais muito mais complexos.

A característica mais importante de ondas elementares e de problemas de Riemann é que suas soluções podem ser consideradas como sendo auto-similares [38]. Isto faz com que se possa reduzir o número de dimensões de dois para um e, portanto, se possa aproveitar resultados analíticos obtidos para o caso unidimensional no estudo de casos bidimensionais, uma vez que a quantidade de resultados sobre a estrutura analítica da solução é muito maior para problemas unidimensionais.

A coleção de informações auto-similares em duas dimensões é muito diversa. Na verdade, qualquer coleção de raios centrados em algum ponto (origem, por exemplo) e separados por estados constantes irá gerar soluções auto-similares. Em [28], [38] e [27] os problemas de Riemann (de valor inicial) para a equação de Euler bidimensional são restritos ao caso particular de quatro estados constantes, um em cada quadrante. Ainda com a imposição de que a onda que separa estados adjuntos seja uma onda elementar.

Considera-se a equação de Euler bidimensional para um gás politrópico:

$$U_t + F(U)_x + G(U)_y = 0, \quad (4.27)$$

sendo

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \quad F(U) = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u(E + P) \end{pmatrix}, \quad G(U) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v(E + P) \end{pmatrix}$$

$$\text{e } P = (\gamma - 1) \left(E - \frac{1}{2} \rho(u^2 + v^2) \right)$$

O problema de Riemann (4.28) é dado pelas soluções iniciais constantes fixadas por quadrante, sendo que o subíndice indica o quadrante considerado:

$$(p, \rho, u, v)(x, y, 0) = \begin{cases} (p_1, \rho_1, u_1, v_1), & x > 0, 5 \ y > 0, 5 \\ (p_2, \rho_2, u_2, v_2), & x < 0, 5 \ y > 0, 5 \\ (p_3, \rho_3, u_3, v_3), & x < 0, 5 \ y < 0, 5 \\ (p_4, \rho_4, u_4, v_4), & x > 0, 5 \ y < 0, 5 \end{cases}, \quad (4.28)$$

Este problema tem sido estudado por vários autores nos últimos tempos. Zhang e Zheng [46] previram 16 classes genuinamente diferentes de problemas de Riemann (4.28) para gases

politrópicos. Schulz e Rinne [37] provaram que um dos 16 casos era impossível. E Zhang, Chen e Yang [45] observam que quando inicialmente estão presentes duas descontinuidades de contato paralelas, isto faz com que as vorticidades geradas tenham sinais opostos ou não, Produzindo, assim, quatro novas classes de problemas de Riemann, totalizando 19 configurações diferentes para o problema.

Segundo Lax e Liu [28], cada uma das 19 configurações possíveis são expressas através das seguintes relações: ω representa a componente da velocidade perpendicular à linha de descontinuidade e ω' representa a componente paralela a ela. Na interface entre cada quadrante os índices $(l, r) \in \{(2, 1), (3, 2), (3, 4), (4, 1)\}$, a onda de rarefação para trás $\overleftarrow{R}_{lr}$ ou para frente $\overrightarrow{R}_{lr}$ é descrita pela seguinte fórmula:

$$\frac{p_l}{p_r} = \left(\frac{\rho_l}{\rho_r} \right)^\gamma.$$

A onda de choque para trás $\overleftarrow{S}_{lr}$ ou para frente $\overrightarrow{S}_{lr}$ é descrita pela seguinte fórmula:

$$\frac{\rho_l}{\rho_r} = \left(\frac{p_l}{p_r} + \frac{\gamma - 1}{\gamma + 1} \right) / \left(1 + \frac{(\gamma - 1)p_l}{(\gamma + 1)p_r} \right).$$

E as descontinuidades de contato J_{lr}^- negativa e J_{lr}^+ positiva são obtidas através de:

$$\begin{aligned} \omega_l &= \omega_r, & p_l &= p_r, & \omega'_l &\geq \omega'_r; \\ \omega_l &= \omega_r, & p_l &= p_r, & \omega'_l &\leq \omega'_r, \end{aligned} \quad \text{respectivamente.}$$

A seguir são apresentados dois exemplos de configurações distintas para problemas de Riemann (4.28), segundo a sequência apresentada por Kurganov e Tadmor em [27].

Configuração 8

As condições iniciais para esta configuração são as seguintes:

$$\begin{array}{cc|cc} \overleftarrow{R}_{21} & & p_2 = 1,0 & \rho_2 = 1,0 \\ J_{32}^- & \overleftarrow{R}_{41} & u_2 = -0,6259 & v_2 = 0,1 \\ J_{34}^- & & p_3 = 1,0 & \rho_3 = 0,8 \\ & & u_3 = 0,1 & v_3 = 0,1 \end{array} \quad \begin{array}{cc} p_1 = 0,4 & \rho_1 = 0,5197 \\ u_1 = 0,1 & v_1 = 0,1 \\ p_4 = 1,0 & \rho_4 = 1,0 \\ u_4 = 0,1 & v_4 = -0,6259 \end{array}$$

Configuração 10

As condições iniciais para esta configuração são as seguintes:

$$\begin{array}{cc|cc} J_{21}^- & & p_2 = 1,0 & \rho_2 = 0,5 \\ \overrightarrow{R}_{32} & \overrightarrow{R}_{41} & u_2 = 0 & v_2 = 0,6076 \\ J_{34}^+ & & p_3 = 0,3333 & \rho_3 = 0,2281 \\ & & u_3 = 0 & v_3 = -0,6076 \end{array} \quad \begin{array}{cc} p_1 = 1,0 & \rho_1 = 1,0 \\ u_1 = 0 & v_1 = 0,4297 \\ p_4 = 0,3333 & \rho_4 = 0,4562 \\ u_4 = 0 & v_4 = -0,4297 \end{array}$$

Os exemplos apresentados a seguir são resolvidos de forma adaptativa, segundo o algoritmo apresentado anteriormente. São apresentadas as soluções apenas para a densidade, conforme é feito em [27].

CONFIGURAÇÃO 10

O nível de resolução mais fino em cada direção possui 2^8 pontos e a quantidade de níveis de faturação na transformada wavelet é igual a 5. O número máximo de iterações no tempo é 250, sendo o tempo final correspondente $T=0,1458$, e a constante de CFL é igual a 0,25. O resultado apresentado na figura (4.44) confere com o resultado apresentado em [27]. Este gráfico foi obtido através do aplicativo *tecplot*.

Figura 4.44: Densidade-Configuração 10

Na figura (4.45) a solução apresentada para a densidade do gás está de acordo com o resultado mostrado em [28]. O aplicativo utilizado agora para criar o gráfico das curvas de mesma densidade foi o *plotmtv*, com 40 contornos possíveis. Percebe-se que neste gráfico são mostrados níveis que não constam no gráfico da figura (4.44).

Wed Jul 17 17:58:49 2002

O segundo gráfico da figura (4.46) representa a malha esparsa obtida através da aplicação

da transformada wavelet na direção y , por coluna, no instante final da evolução temporal. Pode-se pensar que o segundo gráfico é a representação bidimensional do que ocorre na última linha do primeiro gráfico, quando se aplica a transformada por coluna.

Figura 4.46: Número de significativos e Malha Esparsa em y - Conf. 10

CONFIGURAÇÃO 8

O nível de resolução mais fino em cada direção possui 2^8 pontos e a quantidade de níveis de faturação na transformada wavelet é igual a 5. O número máximo de iterações no tempo é 400, sendo o tempo correspondente $T=0,228$ e a constante de CFL é igual a 0,25. O resultado apresentado na figura (4.47) confere como o apresentado em [27]. Este gráfico foi obtido através do aplicativo *tecplot*.

Figura 4.47: Densidade-Configuração 8

O resultado apresentado na figura (4.48) foi obtido através do aplicativo *plotmtv* com 30 contornos para a densidade, conferindo com o resultado apresentado em [28]. Pode-se perceber claramente a influência do recurso computacional utilizado para representar a solução graficamente.

Wed Jul 17 18:21:45 2002

Figura 4.48: Densidade-Configuração 8

O primeiro gráfico apresentado na figura (4.49) indica o número de coeficientes significativos obtidos pela transformada wavelet aplicada às linhas para cada coluna fixada da malha, medidos ao longo das iterações no tempo. O segundo gráfico da figura (4.49) representa a malha esparsa obtida através da aplicação da transformada wavelet na direção y , por colunas, no instante final da evolução temporal.

Figura 4.49: Número de significativos e Malha Esparsa em y - Conf. 8

Os tempos de execução dos programas para cada uma das configurações para malha esparsa e para malha fina são apresentados na tabela (4.2) abaixo:

tempo em segundos	malha esparsa	malha 256x256
configuração 10	475,83	806,77
configuração 08	796,85	1244,06

Tabela 4.2: Tempo de execução do programa

Fica evidente o ganho em tempo de processamento, quando se utiliza o algoritmo adaptativo apresentado neste Capítulo.

Conclusões

Neste trabalho foi apresentado um método adaptativo baseado em esquemas *upwind* de diferenças finitas e em esquemas ENO de alta resolução para resolver equações e sistemas de equações diferenciais hiperbólicas multidimensionais.

A adaptatividade foi obtida através da aplicação de uma *transformada wavelet interpolatória* à solução da equação a cada instante de tempo. Desta forma, obteve-se uma representação multiresolução da solução que, através das operações de truncamento e extensão, transformou-se em uma representação esparsa, associada a uma malha esparsa, na qual o operador diferencial foi evoluído.

A representação esparsa adapta-se automaticamente à solução da equação, conforme esta varia com o tempo, sendo sua variação dada em função da advecção da informação ou em função do desenvolvimento de descontinuidades ao longo da evolução (ondas de choque).

Uma estratégia de corte para os coeficientes wavelet foi proposta neste trabalho e comparada a outras citadas na literatura em relação à taxa de compressão ou ao número de coeficientes de significativos presente na representação esparsa final. Foi proposta também uma estrutura de dados esparsa para armazenar a representação esparsa da solução, obtendo-se assim uma economia em relação ao tempo de processamento e à quantidade de espaço de armazenamento de dados.

Na resolução de problemas unidimensionais, percebeu-se a possibilidade de se atualizar a representação esparsa da solução não mais a cada passo no tempo, mas a cada cinco, dez, n passos no tempo, preservando a qualidade da solução e diminuindo ainda mais o tempo de CPU gasto na simulação.

Através da interpolação ENO e da combinação de esquemas baseados em valores pontuais para representar de forma conservativa a função de fluxo do operador diferencial, construiu-se um esquema adaptativo de alta resolução, com o qual foi resolvido uma ampla coleção de exemplos para problemas de valor inicial associados a equação de Euler unidimensional, denominados problemas do tubo de choque.

Uma extensão natural para problemas bidimensionais foi obtida através da fatoração do operador no tempo (*time splitting method*). Desta forma foi possível aplicar por direção a transformada wavelet unidimensional à solução. Com esta fatoração, resolve-se cada direção em frações diferentes de tempo e por causa da maneira com que se obtém o fluxo numérico,

não é necessária decomposição espectral. Para validação da extensão do método, resolveram-se dois problemas de Riemann para a equação de Euler bidimensional.

A combinação das wavelets interpolatórias como dispositivo de escolha da malha esparsa com esquemas ENO e com desacoplamento das equações por direção resultou numa técnica eficiente para resolução de leis de conservação multidimensionais. Comprovando, portanto, o grande potencial da associação de esquemas de alta resolução para valores pontuais a transformadas wavelet interpolatórias para resolver de forma adaptativa equações cujas soluções possuem grandes variações em pequenas regiões do domínio.

Para trabalhos futuros deseja-se estudar e analisar as seguintes modificações:

- utilizar transformadas direta e inversa, nas quais o interpolador seja não linear, isto é, que o estêncil de interpolação do preditor seja escolhido de acordo com o comportamento local da solução, seguindo a filosofia ENO;
- adaptar o algoritmo de subdivisão para malhas não uniformes para que este recupere a malha uniforme fina, preservando no final do processo de subdivisão propriedades de convexidade e diferenciabilidade para a função;
- obter uma estratégia de inserção de pontos de segurança na RPE de tal forma que as distâncias entre os pontos sejam sempre diádicas. Desta forma, pelo que vimos no capítulo 3 sobre discretização do operador, será possível obter uma simplificação relevante na forma final da discretização para esquemas *upwind*.
- encontrar um dispositivo para realizar a correção da representação esparsa. Acredita-se que seja possível através do comportamento dos coeficientes wavelets controlar também o número de passos no tempo evoluídos sob a mesma representação esparsa. Desta forma, o controle da atualização da malha esparsa associada, também seria feito de forma adaptativa;
- utilizar, por direção, uma integração temporal de ordem ainda mais alta através dos métodos TVD-Runge-Kutta, uma vez que para os exemplos bidimensionais apresentados no capítulo 4 cada direção foi resolvida por um passo no tempo de ordem 1 ou 2;
- modificar a maneira com que se aplica a transformada wavelet por direção nos problemas bidimensionais, uma vez que nos exemplos do capítulo 4, para cada passo intermediário no tempo foram aplicadas as transformações varrendo-se as posições significativas da malha sempre no mesmo sentido;
- o fato de ter-se desacoplado o problema por direção possibilita a implementação de um algoritmo paralelo. O cálculo do operador ao longo das linhas ou colunas pode ser distribuído entre os processadores disponíveis de forma independente. A distribuição do cálculo das tarefas pode ser feita de acordo com o número de processadores disponível em tempo real de execução, uma vez que cada processador poderá receber uma nova

tarefa à medida que termina o cálculo da anterior. Como a distribuição das tarefas neste problema não precisa ser necessariamente sequencial, um bom balanceamento das cargas pode ser obtido, pois sempre que um processador estiver disponível, ele poderá receber imediatamente a próxima tarefa a ser executada de forma a otimizar a taxa de ocupação dos recursos computacionais. Aspectos específicos de implementação em ambientes de memória compartilhada e distribuída devem ser analisados em profundidade.

- Uma vez que todos os aspectos acima discutidos não dependem da dimensão do problema, um algoritmo com estas características também poderá resolver sistemas tridimensionais e sistemas com condições de contorno não periódicas;

1. A análise de sensibilidade mostrou que a taxa de crescimento da população de *Chlamydomonas* é o fator mais influente na determinação da produção de biomassa e de lipídios. A taxa de crescimento da população de *Chlamydomonas* deve ser maximizada para obter a maior produção de biomassa e de lipídios.
2. A análise de sensibilidade mostrou que a taxa de conversão de biomassa em lipídios é o fator mais influente na determinação da produção de lipídios. A taxa de conversão de biomassa em lipídios deve ser maximizada para obter a maior produção de lipídios.
3. A análise de sensibilidade mostrou que a taxa de conversão de biomassa em lipídios é o fator mais influente na determinação da produção de lipídios. A taxa de conversão de biomassa em lipídios deve ser maximizada para obter a maior produção de lipídios.

Apêndice A

A.1 Cálculo dos coeficientes γ_l

Pela definição (1.42), tem-se

$$d_j^k = \bar{u}_{2j-1}^{k-1} - \left[\frac{I(x_{2j-1}^{k-1}, U^k) - U_{j-1}^k}{h_{k-1}} \right]. \quad (\text{A.1})$$

Usando interpolação centrada de grau 3,

$$I(x_{2j-1}^{k-1}, U^k) = \tilde{U}_{2j-1}^{k-1} = \frac{9}{16} [U_j^k + U_{j-1}^k] - \frac{1}{16} [U_{j+1}^k + U_{j-2}^k]. \quad (\text{A.2})$$

Então,

$$I(x_{2j-1}^{k-1}, U^k) - U_{j-1}^k = \tilde{U}_{2j-1}^{k-1} - U_{j-1}^k = \frac{9}{16} [U_j^k + U_{j-1}^k] - \frac{1}{16} [U_{j+1}^k + U_{j-2}^k] - U_{j-1}^k. \quad (\text{A.3})$$

Reagrupando os termos, tem-se:

$$\tilde{U}_{2j-1}^{k-1} - U_{j-1}^k = \frac{9}{16} U_j^k - \frac{7}{16} U_{j-1}^k - \frac{1}{16} [U_{j+1}^k + U_{j-2}^k] \quad (\text{A.4})$$

Agora, escrevendo todas as primitivas em função daquela que tiver menor índice, neste caso, U_{j-2}^k , e ainda em função das células complementares, obtem-se:

$$\tilde{U}_{2j-1}^{k-1} - U_{j-1}^k = \frac{9}{16} [U_{j-2}^k + h_k(\bar{u}_{j-1}^k + \bar{u}_j^k)] - \frac{7}{16} [U_{j-2}^k + h_k \bar{u}_{j-1}^k] - \frac{1}{16} [U_{j-2}^k + h_k(\bar{u}_{j-1}^k + \bar{u}_j^k + \bar{u}_{j+1}^k)] - \frac{1}{16} U_{j-2}^k \quad (\text{A.5})$$

Simplificando-se a expressão acima e reescrevendo-a em função apenas das médias celulares, tem-se:

$$\tilde{U}_{2j-1}^{k-1} - U_{j-1}^k = \frac{1}{h_{k-1}} \left(\frac{1}{16} h_k \bar{u}_{j-1}^k + \frac{8}{16} h_k \bar{u}_j^k - \frac{1}{16} h_k \bar{u}_{j+1}^k \right) \quad (\text{A.6})$$

Lembrando das relações de escala entre as malhas,

$$\frac{h_k}{h_{k-1}} = \frac{2^k N_0}{2^{k-1} N_0} = 2, \quad (\text{A.7})$$

tem-se, portanto:

$$\tilde{U}_{2j-1}^{k-1} - U_{j-1}^k = \frac{1}{8}\bar{u}_{j-1}^k + \bar{u}_j^k - \frac{1}{8}\bar{u}_{j+1}^k. \quad (\text{A.8})$$

Substituindo a expressão acima na definição do coeficiente wavelet para média celular,

$$d_j^k = \bar{u}_{2j-1}^{k-1} - \left(\frac{1}{8}\bar{u}_{j-1}^k + \bar{u}_j^k - \frac{1}{8}\bar{u}_{j+1}^k \right). \quad (\text{A.9})$$

Finalmente, colocando (A.9) em termos de somatório, obtém-se a expressão dada no algoritmo de fatoração e reconstrução do Capítulo 1.

$$d_j^k = \bar{u}_{2j-1}^{k-1} - \left(\bar{u}_j^k + \sum_{l=1}^1 \left(-\frac{1}{8} \right) (\bar{u}_{j-l}^k - \bar{u}_{j+l}^k) \right). \quad (\text{A.10})$$

Considerando-se $\bar{r} = r - 1$, com r sendo o grau do interpolador para a primitiva, tem-se a seguinte tabela para os valores de γ_l , obtidos de forma análoga ao caso descrito acima:

$$\begin{aligned} \bar{r} = 3 &\Rightarrow \gamma_1 = -\frac{1}{8} \\ \bar{r} = 5 &\Rightarrow \gamma_1 = -\frac{22}{128} \quad \gamma_2 = -\frac{3}{128} \end{aligned} \quad (\text{A.11})$$

E assim, para as demais escolhas do interpolador da primitiva de $u(x)$, tem-se associados os valores de γ_l .

Referências Bibliográficas

- [1] Arandiga, F.; Donat, R.; Harten, A. *Multiresolution based on weighted Averages of the Hat function II: Non-linear Reconstruction Techniques*. SIAM, J. Sci. Comput., vol 20, n. 3, 1053-1093, 1999.
- [2] Arandiga, F.; Donat, R.; Harten, A. *Multiresolution Based on Weighted Averages of the Hat Function I: Linear Reconstruction Techniques*. SIAM J. Numer Analysis, vol 36, n. 1, 160-203, 1998.
- [3] Bertoluzza, S.; Naldi, G.; Ravel, J.C. *Wavelet methods for the numerical solution of boundary value problems on the interval*, Wavelets: Theory, Algorithms and Applications, C.K.Chui, L.Montefuso, L.Puccio, eds. Academic Press, Nova Iorque, 425-448, 1994.
- [4] Berger, M.;Oliger, J. *Adaptive Mesh Refinement for Hyperbolic Partiall Differential Equations*. Journal of Computational Physics. 1984.
- [5] Bihari, B.; Harten, A. *Application of Generalized Wavelets: An Adaptive Multiresolution Scheme*. Journal of Computational and Applied Mathematics. 61, 275-321. 1995.
- [6] Brezis, H. *Analisis Funcional: Teoria y aplicaciones*. Madrid Alianz, 1984.
- [7] Cohen, A. *Wavelet methods in numerical analysis*. Université Pierre et Marie Curie, Paris.
- [8] Delbourgo, R.; Gregory, J.A. *Shape preserving piecewise rational interpolation* SIAM J. Sci. Statist. Comput., 6, 967-976, 1985.
- [9] Deslauriers, G.; Dubuc, S. *Symmetric Iterative Interpolation Process*. Constructive Appoximation, Springer Verlag, New York, 1989.
- [10] Domingues, M. O. *Análise Wavelet na Simulação Numérica de Equações Diferenciais Parciais com Adaptatividade Espacial*. Tese de Doutorado em Matemática Aplicada, Universidade Estadual de Campinas, IMECC, 2001.

- [11] Donat, R. *Studies on Error Propagation for certain Nonlinear Approximations to hyperbolic Equations: Discontinuities in Derivatives*. SIAM J. Numer. Analysis, Vol 31, n 3, 665-679, 1994.
- [12] Dubuc, S. *Interpolation through an Iterative Scheme*. Journal of Mathematical Analysis and Applications, 114, 185-204, 1986.
- [13] Garcia, C. I. *Métodos de refinamento local em modelos globais de previsão numérica do tempo* Tese de doutorado. Instituto de Matemática e Estatística, USP, 2001.
- [14] Gottlieb, S.; Shu, C.-W. *Total Variation Diminishing Runge-Kutta Schemes*. ICASE Technical Report, TR96-50, Julho 1996.
- [15] Harten, A. *Multiresolution Algorithms for the Numerical Solution of Hyperbolic Conservation Laws*. Communication on Pure and Applied Mathematics, Vol XLVIII, 1305-1342, 1995.
- [16] Harten, A. *Discrete Multiresolution Analysis and Generalized Wavelets*. CAM Report 92-08, University of California, 1992.
- [17] Harten, A.; Yad-Shalom, I. *Fast Multiresolution Algorithms for Matrix-Vector Multiplication* ICASE Report 92-55, 1992.
- [18] Harten, A. *ENO Schemes with Subcell Resolution*. Journal of Computational Physics, 83, 148-184, 1989.
- [19] Harten, A.; Engquist, B.; Osher, S.; Chakravarthy, S. *Uniformly High Order Accurate Essentially Non-oscillatory Schemes, III*. Journal of Computational Physics, 71, 231-303, 1987.
- [20] Holmström, M. *Solving Hyperbolic PDE's Using Interpolating Wavelets*. Report 189, Departmente of Scientific Computing, Uppsala University, Sweden, 1996.
- [21] Holmström, M. *Wavelet Based Methods for Time Dependent PDE's*. PhD Thesis, Faculty of Science and Technology, Uppsala, Sweden, 1997.
- [22] Jameson, L. *On the wavelet optimized finite difference method*. Tech. report ICASE 94-9, NASA Langely Research Center, Haptom, VA, 1994.
- [23] Jameson, L. *A wavelet optimized, very high order adaptive grid and order numerical method*. SIAM J. Sci. Comput., 19, 1980-2031, 1998.
- [24] Kaibara, M. *Análise de Multi-resolução para Leis de Conservação em Malhas Adaptativas*. Tese de Doutorado, Universidade Estadual de Campinas, 2000.
- [25] Kujit, F.; van Damme, R. *Shape preserving interpolatory subdivision schemes for nonuniform data*. to be printed.

- [26] Kuijt, F.; van Damme, R. *Monotonicity preserving interpolatory subdivision schemes*. J. Comput. Appl. Math., 101(1-2), 203-229, 1999.
- [27] Kurganov., A.; Tadmor, E. *Solution of Two-Dimensional Riemann Problems of Gas Dynamics without Riemann Problem Solvers*. Numerical Methods in Partial Differential Equations. to appear. 2000
- [28] Lax, P.; Liu, X.-D. *Solution of Two-Dimensional Riemann Problems of Gas Dynamics by Positive Schemes* . SIAM Journal of Scientific Computing, 19, 2, 319-340, 1998.
- [29] Le Veque, R. *Numerical Methods for Conservation Laws*. Birkhäuser Verlag. 1992.
- [30] Le Veque, R. *Nonlinear Conservation Laws and Finite Volume Methods for Astrophysical Fluid Flows* . Computational Methods for Astrophysical Fluid Flow. Springer-Verlag.
- [31] Liu, X.-D.; Osher, S. *Convex ENO High Order Multi-Dimensional Schemes without Field by Field Decomposition or Staggered Grids* . Journal of Computational Physics, 141, 1-27, 1998.
- [32] Nielsen, O. *Wavelets in Scientific Computing* , PhD Thesis, Technical University of Denmark, 1998.
- [33] Osher, S.; Shu, C.-W. *Efficient Implementation of Essentially Non-oscillatory Shock-Capturing Schemes,II*. Journal of Computational Physics, 83, 32-78, 1989.
- [34] Percival, Walden. *Wavelet and Time Series*. Cambridge Press, 2000.
- [35] Roma, A., Peskin, C., Berger, M. *An Adaptive Version of the Immersed Boundary Method* Journal of Computational Physics, 153, 509-534, 1999.
- [36] Plantevin, F. *Wavelets on irregular meshes* . Advances in Computational Mathematics , 4, 293-329, 1995.
- [37] Schulz-Rinne, C. W. *Classification of the Riemann problem for two-dimensional gas dynamics* . SIAM J. Math. Anal.24, 76-88, 1993.
- [38] Schulz-Rinne, C.; Collins, J.; Glaz, H. *Numerical Solution of the Riemann Problem for Two-Dimensional Gas Dynamics* . SIAM Journal of Scientific Computing, 14, 6, 1394-1414, 1993.
- [39] Shu, C.-W. *Essentially Non-Oscillatory and Weighted Essentially Non-Oscillatory Schemes for Hyperbolic Conservation Laws*. ICASE Report, No 97-65, NASA, 1997.
- [40] Shu, C.-W.; Osher, S.; *Efficient Implementation of Essentially Non-oscillatory Shock-Capturing Schemes*. Journal of Computational Physics, 77, 439-471, 1988.
- [41] Sod, G. *Numerical Methods in Fluid Dynamics*. Cambridge University Press, Cambridge, 1985.

- [42] Stoer, J.; Bulirsch, *Introduction to Numerical Analysis*. Second Edition, Springer Verlag, 1993.
- [43] Strikwerda, J. *Finite Difference Schemes and Partial Differential Equations* Chapman e Hall, 1989.
- [44] Sweldens, W. *Building your own Wavelets at Home*. <http://cm.bell-labs.com/cm/ms/who/wim/papers>.
- [45] Zhang, T.; Chen, G.-Q.; Yang, S. *On the 2-D Riemann problem for the compressible Euler equations, I. Interaction of shocks and rarefaction waves*. Discrete Continuous Dynam. Systems, 1, 555-584, 1995.
- [46] Zhang, T.; Zhen, Y. *Conjecture on the structure of solutions of the Riemann problem for two-dimensional gas dynamics systems*. SIAM J. Math. Anal., 21, 593-630, 1990.