Paulo Miguel Pereira de Brito

Aplicação de Métodos Numéricos Adaptativos na Integração de Sistemas Algébrico-Diferenciais Caracterizados por Frentes Abruptas

Departamento de Engenharia Química

Faculdade de Ciências e Tecnologia

Universidade de Coimbra

COIMBRA

1998

APLICAÇÃO DE MÉTODOS NUMÉRICOS ADAPTATIVOS NA INTEGRAÇÃO DE SISTEMAS ALGÉBRICO-DIFERENCIAIS CARACTERIZADOS POR FRENTES ABRUPTAS

Paulo Miguel Pereira de Brito

Tese submetida à Faculdade de Ciências e Tecnologia da Universidade de Coimbra para Mestrado em Engenharia Química, área de Processos Químicos

Trabalho subsidiado pela

JUNTA NACIONAL DE INVESTIGAÇÃO CIENTÍFICA E TECNOLÓGICA
(J. N. I. C. T.)
através de uma Bolsa de Mestrado
PRAXIS XXI / BM / 1771 / 94

COIMBRA

Março de 1998

Aos meus Pais

Agradecimentos

Pela sua importante contribuição no desenrolar deste trabalho gostaria de agradecer:

- ao Prof. Dr. António A. T. G. Portugal, meu supervisor, pela orientação atenta e interessada que soube manter, durante a execução do presente trabalho;
- ao Eng. Belmiro Duarte, pelo interesse e apoio que sempre manifestou;
- a todos os meus amigos no Departamento de Engenharia Química (DEQ), que me acompanharam neste percurso;
- ao Departamento de Engenharia Química (DEQ), na pessoa do seu presidente, Profª. Drª. Maria Margarida L. Figueiredo, pelas condições de trabalho cedidas e pelo interesse demonstrado;
- à Junta Nacional de Investigação Científica e Tecnológica (JNICT) pelo suporte financeiro concedido.

Resumo

O objectivo do presente trabalho consiste no desenvolvimento e estudo de algoritmos adaptativos de integração para sistemas de **Equações Diferenciais Parciais/Algébricas** evolutivas e unidimensionais. Estes algoritmos baseiam-se em estratégias de adaptação espacial da malha, associados a discretizações caracterizadas por aproximações de diferenças finitas.

Os referidos métodos foram escolhidos de forma a representarem dois tipos de estratégias alternativas no tratamento de soluções problemáticas que desenvolvam frentes abruptas e/ou choques móveis e, que, tradicionalmente colocam bastantes dificuldades de integração, através de métodos baseados em malhas, temporal e espacialmente, fixas. Ambos os algoritmos apresentam uma estrutura semelhante, podendo ser divididos em dois estágios distintos.

❶ **Estágio I** ➔ Identificação de subdomínios onde a introdução de uma estratégia de adaptação se revela necessária. Este estágio é equivalente em ambos os algoritmos, sendo efectuado através da comparação dos perfis de solução obtidos pela integração do problema em duas malhas fixas de tamanho diferente (uma malha fina e outra esparsa);

❷ **Estágio II** ➔ Integração dos subproblemas gerados, para cada um dos subdomínios detectados no estágio anterior, pela introdução de um procedimento adaptativo, na resolução do problema original. Neste estágio, as estratégias de adaptação da malha base diferem bastante entre cada um dos algoritmos. Assim, tem-se:

➲ ***Algoritmo de Refinamento*** – caracterizado pelo refinamento sucessivo de cada uma das submalhas detectadas até à satisfação do critério de precisão previamente estabelecido.

➲ ***Algoritmo de Malha Móvel*** – caracterizado pela introdução de uma estratégia de mobilidade nodal dinâmica em cada uma das submalhas referidas anteriormente.

Para cada um dos algoritmos foi desenvolvida uma *package* de carácter geral, de forma a se tornar possível a sua aplicação prática. Estas *packages* foram testadas em diversas condições, tanto para **Equações Diferenciais Parciais** (P.D.E.´s) escalares e vectoriais, como para **Sistemas Mistos Algébrico Diferenciais** (P.D.A.E.´s). A aplicação da discretização espacial a cada modelo transforma o problema original num sistema de **Equações Diferenciais Ordinárias** (O.D.E.´s), que é integrado no tempo por intermédio do integrador implícito DASSL. No que diz respeito à avaliação da solução em abcissas intermédias, em relação às posições nodais das malhas base, foi estudada a influência de dois tipos de interpolação: linear e através de *Splines* cúbicas, tendo-se verificado a melhor adequação das aproximações lineares para perfis com variações do gradiente mais elevadas e bruscas.

A qualidade da performance de cada método foi analisada, tendo como referência os resultados apresentados por *Duarte*[29], baseados na utilização de uma formulação do **Método dos Elementos Finitos Móveis**, quer no que diz respeito à precisão das soluções obtidas, como dos esforços computacionais exigidos.

Em geral, ambos os algoritmos se revelaram robustos na resolução de um variado conjunto de problemas. No entanto, o **Método de Malha Móvel**, mostrou-se particularmente apropriado para problemas que desenvolvem frentes abruptas móveis, onde as magnitudes das derivadas espaciais são especialmente elevadas. No caso do **Método de Refinamento**, os resultados são comparativamente de qualidade inferior, revelando maiores dificuldades de aplicação em modelos de integração mais difícil.

Foram analisados dois procedimentos diferentes para o tratamento das condições fronteiras interiores dos subproblemas gerados, tendo-se concluído que a estratégia adoptada no **Método de Refinamento**, baseada na fixação de condições de *Dirichlet* artificiais, não se revela satisfatória, podendo ocasionar dificuldades acrescidas na integração de modelos específicos. Pelo contrário, a estratégia aplicada no **Método de Malha Móvel**, baseada na simulação da evolução temporal da solução, mostrou-se bastante robusta, em todos os problemas testados.

Desenvolveram-se, igualmente, diversas subrotinas para a avaliação de derivadas de diferentes ordens, através de correlações de diferenças finitas de tipo e grau de precisão muito variados, para malhas arbitrariamente espaçadas, em sistemas coordenados de dimensão máxima três. Consideram-se vários processos para a avaliação dos pesos associados a cada uma destas aproximações.

Índice

Índice de Figuras

Índice de Gráficos

Índice de Tabelas

1. Introdução

A simulação de sistemas físicos descritos por **Equações Diferenciais Parciais** (P.D.E.´s) constitui um campo de pesquisa bastante explorado que pode apresentar muitas dificuldades particularmente se as soluções desenvolvem frentes móveis e abruptas e/ou choques. Estes problemas podem surgir em aplicações físicas tão distintas como o escoamento de fluidos, aerodinâmica, condução térmica, reacção química, combustão e propagação de impulsos electroquímicos. Geralmente, a complexidade matemática dos modelos que descrevem estes fenómenos torna impraticável o desenvolvimento de soluções analíticas. Deste modo, é imperativo recorrer a simulações numéricas que constituem as únicas ferramentas eficientes para solucionar estes casos. No entanto, para sistemas de grande dimensão, as técnicas numéricas desenvolvidas têm que ser suficientemente eficientes de modo a possibilitarem resultados precisos, utilizando tempos computacionais realistas.

As P.D.E.´s que constituem os modelos de simulação, são desenvolvidas a partir de balanços de massa, energia e quantidade de movimento. As derivadas temporais ocorrem em função da dinâmica transiente, enquanto que as derivadas espaciais têm, normalmente, origem em fenómenos convectivos ou difusionais. Podem, igualmente, ser desenvolvidas **Equações Algébricas** (E.A.´s) associadas às P.D.E.´s, que decorrem geralmente de relações de equilíbrio ou de aplicação de princípios termodinâmicos. Para completar o modelo e torná-lo resolúvel, é necessário definir as condições iniciais e fronteira respectivas que têm uma influência determinante na solução.

Muitos dos **Métodos Numéricos** existentes são baseados em **Malhas de Discretização Fixas**. Nestes, todos os pontos da malha (designados por nodos) são tratados do mesmo modo, independentemente do facto de se situarem em zonas de pequena variação da solução, onde poderiam ser removidos sem qualquer risco de perda de precisão desta. Recentemente, houve a preocupação de desenvolver **Métodos de Malha Adaptativa**, onde se procura concentrar os nodos preferencialmente nas zonas de maior actividade da solução. Inicialmente, estes métodos foram concebidos para a resolução de **Problemas de Valor Fronteira** (B.V.P.), descritos por **Equações Diferenciais Ordinárias** (O.D.E.´s), sendo posteriormente generalizados para sistemas evolutivos de dimensão variável. Assim, à medida que a solução evolui no tempo, os nodos são deslocados para as posições com elevados gradientes de solução, sendo o esforço computacional centrado nas regiões onde é mais necessário. As estratégias iniciais redefiniam a malha a partir de critérios baseados na parametrização da solução em intervalos de tempo discretos. No entanto, em anos mais recentes foram apresentados métodos nos quais os nodos se movimentam continuamente com o tempo, de acordo com uma propriedade característica da solução.

No **Capítulo 2** apresenta-se uma classificação geral das P.D.E.´s, sendo enunciados os tipos de problema descritos por estas cujas soluções podem desenvolver frentes móveis e abruptas e descontinuidades. Referem-se, igualmente, os problemas associados à solução numérica: difusão e dissipação. Neste capítulo são referidos os principais métodos aplicados na resolução numérica de P.D.E.´s duma forma resumida.

O **Capítulo 3** constitui um resumo das principais estratégias desenvolvidas para movimentação dos nodos no contexto dos métodos adaptativos, já referidos anteriormente. Não se pretende que esta revisão bibliográfica seja demasiado abrangente ou pormenorizada, mas que dê uma ideia geral dos diferentes algoritmos de redistribuição nodal apresentados.

Pretende-se apenas dar atenção aos algoritmos aplicados em discretizações baseadas em esquemas de **Diferenças Finitas**.

No **Capítulo 4** são desenvolvidas as estratégias adaptativas seleccionadas para estudo no presente trabalho e apresentam-se os respectivos algoritmos, desenvolvidos para a sua aplicação. Cada método considerado baseia-se numa estratégia de manipulação nodal, inserida em cada um dos dois grupos gerais em que se podem dividir os métodos de integração adaptativos mais frequentemente utilizados:

❶ **Estratégia de introdução e remoção de nodos** ⇨ ***Método de Refinamento*:**

Caracterizado pelo refinamento das malhas base na regiões do domínio espacial, onde se verifiquem maiores erros no avanço da solução temporal.

❷ **Estratégia de movimentação dos nodos** ⇨ ***Método de Malha Móvel*:**

Baseado na introdução de estratégias de movimentação nodal dinâmica, para a integração temporal, nas regiões do domínio onde se constatem a ocorrência de estimativas para o erro espacial mais elevadas.

A selecção das subregiões do domínio espacial, onde se revela necessária a aplicação de um procedimento adaptativo, é realizada da mesma forma em cada uma estratégias anteriores, recorrendo-se à comparação das soluções obtidas por integração temporal em duas malhas arbitrárias de espaçamento local distinto. Desta forma, é possível efectuar-se a estimativa do erro associado à discretização espacial, em cada nodo da malha.

Para cada um dos métodos acima referidos, é desenvolvida uma *package* de carácter geral para a resolução de sistemas de **Equações Diferenciais Parciais/Algébricas** (P.D.E./A.´s)

No **Capítulo 5** procede-se à comparação do desempenho numérico de cada um dos métodos considerados, tomando como exemplo a equação víscida de *Burgers* (designada por **Exemplo 1**). Esta equação de carácter não linear, já amplamente estudada por variados autores, origina, em certas condições predominantemente hiperbólicas, uma frente móvel bastante abrupta, constituindo por isso, uma ferramenta eficaz para o teste dos códigos de resolução de P.D.E.´s. Para uma melhor avaliação da performance de cada método, utilizam-se como referência os resultados apresentados por *Duarte*[29], obtidos através da aplicação de uma formulação do **Método de Elementos Finitos Móveis** (M.E.F.M.), com aproximações polinomiais cúbicas de *Hermite*. A comparação entre os desempenhos de cada método considerado, é efectuada essencialmente, a dois níveis: a qualidade dos resultados numéricos, ou seja a sua precisão; o esforço computacional exigido para a sua execução.

No **Capítulo 6** são apresentados e discutidos os resultados obtidos pela aplicação dos métodos referidos anteriormente, na resolução de uma vasta gama de problemas, desde problemas caracterizados por P.D.E.´s escalares até sistemas de equações algébrico-diferenciais. Os exemplos considerados constituem-se como modelos típicos, que tradicionalmente suscitam dificuldades consideráveis na sua resolução. Desta forma, torna-se possível realizar um estudo pormenorizado do desempenho de cada um dos algoritmos, nas condições mais variadas, permitindo a constatação das suas potencialidades e limitações. Os exemplos considerados são apresentados de seguida:

① - P.D.E.´s Escalares:

- **Exemplo 2** – Adsorção num leito fixo **(M.Ref. e M.M. Móvel)**;
- **Exemplo 3** – Reactor pistão difusional **(M.Ref. e M.M. Móvel)**;
- **Exemplo 4** – Combustão **(M.Ref. e M.M. Móvel)**;
- **Exemplo 5** – Difusão, convecção e reacção numa partícula plana **(M.Ref.)**.

② - Sistemas de P.D.E.´s:

- **Exemplo 6** – Propagação de uma chama **(M.Ref. e M.M. Móvel)**;
- **Exemplo 7** – Reactor tubular não isotérmico **(M.Ref. e M.M. Móvel)**;
- **Exemplo 8** – Propagação de ondas **(M.Ref. e M.M. Móvel)**;
- **Exemplo 9** – Excitação de um nervo **(M.Ref. e M.M. Móvel)**;
- **Exemplo 10** – Adsorção num leito fixo **(M.Ref.)**.

③ - Sistemas de P.D.A.E.´s:

- **Exemplo 11** – Reactor tubular não isotérmico **(M.Ref.)**;
- **Exemplo 12** – Coluna de adsorção-reacção **(M.Ref.)**.

O teste ao **Método de Refinamento** é mais exaustivo do que o correspondente ao **Método de Malha Móvel** dinâmico. No entanto, este revela-se mais eficaz do que o anterior (nos exemplos em que é aplicado), possibilitando resultados concordantes com a bibliografia, com tempos de computação razoáveis. Por outro lado, as soluções obtidos a partir do refinamento, são relativamente satisfatórias, na generalidade dos exemplos aplicados, apesar do método revelar algumas dificuldades de desempenho, em determinadas condições.

Finalmente, no **Capítulo 7**, apresentam-se as principais conclusões do trabalho, definindo-se algumas indicações gerais, que possibilitem uma utilização mais eficiente das *packages* desenvolvidas, por parte de um utilizador não familiarizado com as características dos algoritmos respectivos. Enumeram-se, igualmente, diversas propostas potencialmente interessantes, como áreas de estudo futuro.

2. Problemas Físicos e Métodos de Integração

Neste Capítulo apresentam-se alguns dos critérios de classificação mais importantes para P.D.E.´s, assim como as equações cujas soluções desenvolvem tipicamente perfis abruptos móveis e/ou choques. Referem-se resumidamente os principais grupos de métodos utilizados na resolução de equações diferenciais.

2.1 - Classificação de P.D.E.´s

Na formulação matemática da maior parte dos problemas dinâmicos em engenharia são envolvidas taxas de variação em relação a mais de uma variável independente, nomeadamente o tempo e uma ou várias dimensões espaciais. Obtém-se, desse modo equações, designadas por **Equações Diferenciais Parciais**, que é necessário integrar, para a resolução do problema.

Com a aplicação dos princípios de conservação de massa, energia e momento a equação geral,

$$a \cdot \frac{\delta^2 u}{\delta x^2} + b \cdot \frac{\delta^2 u}{\delta x \cdot \delta y} + c \cdot \frac{\delta^2 u}{\delta y^2} + d \cdot \frac{\delta u}{\delta x} + e \cdot \frac{\delta u}{\delta y} + f \cdot u + g = 0 \qquad (2.1)$$

é obtida com bastante frequência sendo designada por **Equação Parcial Diferencial** bidimensional de segunda ordem. Portanto, sendo esta a P.D.E. geral mais comum desenvolvida em problemas de engenharia, é utilizada como base para a classificação de P.D.E.´s. Estas diferenciam-se nas suas propriedades, pela forma como os diversos coeficientes são definidos. Os conceitos apresentados de seguida poderão ser alargados a outros tipos de equações ou sistemas de equações, de diferentes ordens ou dimensões. A ordem de uma P.D.E. é definida pela maior ordem das derivadas nela envolvidas. No caso de um sistema de P.D.E.´s, esta é determinada pela equação de ordem mais elevada que o compõe.

A equação (2.1) pode ser classificada quanto à sua linearidade como:

Linear ⇨ Se todos os coeficientes dependerem apenas das variáveis independentes **x** e **y**.

Quasilinear ⇨ Se os coeficientes forem igualmente definidos em função da variável dependente **u** e/ou os coeficientes **a**, **b** e **c** dependerem das primeiras derivadas de **u**.

Semi-Linear ⇨ Se apenas os termos independentes dependerem da variável dependente **u** (Caso particular da **Equação Quasilinear**).

Não-Linear ⇨ Todos os outros casos.

O grau de não-linearidade de uma P.D.E. encontra-se fortemente relacionado com o grau de dificuldade expectável na resolução do problema e, consequentemente, determina a escolha do método de integração a utilizar. Este conceito pode ser facilmente generalizado para equações ou sistemas de equações de ordem diferente. Outra forma de classificação de P.D.E.´s de segunda ordem do tipo da equação (2.1) é baseada no valor do descriminante $\Delta = b^2 - 4.a.c$. Assim:

$b^2 - 4 \cdot a \cdot c > 0$ ⇨ **P.D.E. Hiperbólica**

$b^2 - 4 \cdot a \cdot c = 0$ ⇨ **P.D.E. Parabólica**

$b^2 - 4 \cdot a \cdot c < 0$ ⇨ **P.D.E. Elíptica**

Esta classificação pode ser igualmente estendida a sistemas de n P.D.E.´s de primeira ordem do tipo:

$$\underline{\underline{A}} \cdot \frac{\delta \underline{u}}{\delta t} + \underline{\underline{B}}^{*} \cdot \frac{\delta \underline{u}}{\delta x} = \underline{c}^{*} \qquad (2.2)$$

em que: $\underline{\underline{A}}$ e $\underline{\underline{B}}^{*}$ são matrizes $n \times n$,
$\underline{u} = [\, u_1 , u_2 , ..., u_n \,]^T$ - vector das variáveis dependentes,
$\underline{c}^{*} = [\, c_1 , c_2 , ..., c_n \,]^T$ - vector dos termos independentes.

Multiplicando (2.2) por $\underline{\underline{A}}^{-1}$ tem-se:

$$\frac{\delta \underline{u}}{\delta t} + \underline{\underline{B}} \cdot \frac{\delta \underline{u}}{\delta x} = \underline{c} \qquad (2.3)$$

Através da diagonalização da matriz $\underline{\underline{B}}$, obtém-se a relação:

$$\underline{\underline{B}} = \underline{\underline{S}} \cdot \underline{\underline{\Lambda}} \cdot \underline{\underline{S}}^{-1} \qquad (2.4)$$

onde $\underline{\underline{S}}$ e $\underline{\underline{\Lambda}}$ são a matriz dos vectores próprios e a matriz diagonal dos valores próprios de $\underline{\underline{B}}$. Ambas as matrizes $\underline{\underline{B}}$ e $\underline{\underline{\Lambda}}$ são quadradas, já que $\underline{\underline{B}}$ apenas é diagonalizável se fôr quadrada, o que implica que $\underline{\underline{\Lambda}}$ também o seja. Obtém-se então:

$$\frac{\delta \underline{u}}{\delta t} + \underline{\underline{S}} \cdot \underline{\underline{\Lambda}} \cdot \underline{\underline{S}}^{-1} \cdot \frac{\delta \underline{u}}{\delta x} = \underline{c} \qquad (2.5)$$

Deste modo, o sistema de P.D.E.´s é:

- **Hiperbólico** ⇨ se os n valores próprios forem reais e diferentes;
- **Parabólico** ⇨ se os n valores próprios forem reais e iguais;

➲ **Elíptico** ⇨ se os n valores próprios forem complexos.

Esta classificação torna-se impossível de realizar se o sistema fôr caracterizado por equações de ordens diferentes e/ou envolver equações algébricas. A semelhança das classificações descritas para sistemas de primeira ordem e equações de segunda ordem é óbvia. Esta relação é muito simples de estabelecer bastando, para tal, transformar a equação geral de segunda ordem (2.1) inicial, no sistema de equações de primeira ordem correspondente, obtendo-se a generalização desta classificação para estes dois casos.

2.2 - Apresentação de P.D.E.´s Típicas que Exibem Soluções com Perfis Abruptos Móveis e/ou Choques

Nesta secção são descritos vários modelos de P.D.E.´s cujas soluções desenvolvem tipicamente frentes abruptas móveis. Tal comportamento pode ser originado pela não-linearidade em termos difusivos, convectivos ou geracionais, quer na própria equação ou nas condições fronteiras, ou simplesmente pela advecção de um perfil inicialmente abrupto. Este tipo de equações resulta, normalmente, da modelização em estado transiente de unidades, cujas propriedades em estado estacionário dependem de dimensões espaciais (Ex: reactores tubulares, colunas de destilação). Por uma questão de simplicidade apenas serão referidos modelos unidimensionais.

Difusão Linear

O exemplo típico deste tipo de equações é o modelo de difusão num meio isotrópico descrito por:

$$\frac{\delta u}{\delta t} = D \cdot \frac{\delta^2 u}{\delta x^2} \qquad (2.6)$$

em que: u - concentração da substância difundida,
D - coeficiente de difusão.

No caso de D ser constante, a equação (2.6) é linear. Em problemas de difusão, a velocidade característica é infinita. Desse modo, a evolução da solução **u(x,t)**, em qualquer ponto genérico **(x,t)**, depende do valor da solução em todo o domínio. Este comportamento define o carácter global dos problemas de difusão. Por outro lado, no caso de problemas hiperbólicos, as características apresentam uma velocidade de avanço finita sendo, deste modo, de natureza local, já que a solução em cada ponto do domínio **(x,t)** é apenas influenciada pelo valor desta nessa região do domínio.

Difusão Linear com Termos Geracionais

Os problemas de difusão em que o difundido pode reagir quimicamente, são representados pela adição de um termo geracional à equação de difusão linear:

$$\frac{\delta u}{\delta t} = D \cdot \frac{\delta^2 u}{\delta x^2} + g(u) \qquad (2.7)$$

No caso de g(u) ser fortemente não-linear, as soluções deste tipo de equações desenvolvem, frequentemente, frente abruptas móveis. Se o valor de g(u) fôr bastante elevado, num intervalo pequeno $[u_0 , u_1]$, e relativamente pequeno fora deste intervalo, então, duas frentes abruptas expandir-se-ão em ambas as direcções, a partir de qualquer impulso que exceda o valor crítico u_1. Os termos geracionais podem igualmente estar relacionados com condições fronteira dependentes do tempo.

Difusão Não-Linear

Para muitos casos de aplicações físicas, os coeficientes de difusão dependem da concentração do difundido. Uma relação consideravelmente não-linear pode provocar frentes móveis bastante abruptas. Então, a equação geral tomará a forma:

$$\frac{\delta u}{\delta t} = \frac{\delta}{\delta x}\left(D(u) \cdot \frac{\delta u}{\delta x} \right) \qquad (2.8)$$

Este comportamento ocorre igualmente para modelos de difusão não-lineares acoplados a termos geracionais também não-lineares.

Advecção Linear

Os problemas difusivos referidos anteriormente estão relacionados com equações do tipo parabólico. No entanto, existem exemplos de equações hiperbólicas, onde se desenvolvem frentes abruptas, que podem originar choques correspondentes a descontinuidades físicas. O exemplo mais simples deste tipo de equação é a equação escalar de onda unidimensional:

$$\frac{\delta u}{\delta t} + a \cdot \frac{\delta u}{\delta x} = 0 \qquad (2.9)$$

em que **a** é uma constante e a solução inicial é definida por:

$$u(x,0) = u^0(x) \qquad (2.10)$$

A partir do perfil inicial, a solução movimenta-se com velocidade **a**, mantendo a sua forma inalterável. Assim, qualquer perfil abrupto em $\mathbf{u^0(x)}$ transforma-se numa frente abrupta móvel.

No caso de advecção não-linear, ou seja, para relações de dependência a(u) não-lineares, as frentes podem se formar com o tempo e evoluir para a forma de choques. Este comportamento ocorre mesmo no caso da solução inicial apresentar perfis suaves. O exemplo mais simples deste tipo de equações é a equação invíscida de *Burgers* em que a(u) = u :

$$\frac{\delta u}{\delta t} + u \cdot \frac{\delta u}{\delta x} = 0 \qquad (2.11)$$

Convecção-Difusão

Em muitos processos físicos, os fenómenos convectivos e difusivos ocorrem simultaneamente. Apesar da presença da difusão assegurar que os gradientes da solução sejam finitos, estes podem apresentar valores muito elevados para casos onde a convecção domina. Formam-se assim, quasi-choques, ou seja, frentes bastante abruptas e praticamente descontínuas. O exemplo mais simples deste tipo é a equação de *Burgers*:

$$\frac{\delta u}{\delta t} = -u \cdot \frac{\delta u}{\delta x} + \varepsilon \cdot \frac{\delta^2 u}{\delta x^2} \qquad (2.12)$$

A equação (2.12) apresenta frentes móveis de espessura **O(ε)** para valores de **ε** muito baixos (**ε** << 1).

2.3 - Métodos Numéricos de Resolução de P.D.E.´s

A solução de uma P.D.E. pode ser obtida de duas formas: através da dedução da sua função analítica ou recorrendo a uma aproximação numérica da solução analítica. A exactidão das soluções analíticas constitui uma grande vantagem. No entanto, a sua dedução só se torna possível para o caso de problemas escalares lineares de baixa/média complexidade, o que não acontece para a maioria esmagadora dos problemas relevantes na área da **Engenharia Química**. Deste modo, o caminho que, na maioria dos problemas, é imperioso seguir consiste na obtenção de soluções numéricas.

A aproximação numérica encontra-se associada a um erro que determina a sua qualidade em termos de exactidão. Foram apresentados diversos algoritmos de integração numérica, sucessivamente aperfeiçoados, que constituem vários grupos de métodos numéricos. Cada método representa uma estratégia distinta de resolução do problema. No entanto, estes algoritmos têm em comum, a característica de conversão do problema original, enunciado num espaço definido pelas variáveis independentes de natureza contínua, num problema equivalente num espaço discreto. Deste modo, o problema passa a apresentar um carácter algébrico. A solução é calculada em pontos discretos ao longo do domínio, sendo esta estrutura designada por malha ou grelha.

Em seguida, são descritos duma forma resumida os métodos principais de integração numérica de P.D.E.´s.

2.3.1 - Integral de Convolução[1]

Os diferentes métodos de geração de soluções numéricas de P.D.E.´s não são mais que a aplicação de diversas ferramentas para a resolução do mesmo problema: obtenção duma aproximação numérica da solução com uma exactidão aceitável e utilizando tempos computacionais realistas. Cada uma das estratégias propostas apresentam, necessariamente, vantagens e desvantagens em relação às restantes. Para evidenciar essa semelhança, considera-se a existência de uma solução generalizada para a P.D.E. inicial na forma do operador linear:

$$f(x, t_0 + \tau) = \int_{\varepsilon=-\infty}^{\varepsilon=+\infty} w(\varepsilon) \cdot f(x + \varepsilon, t_0)\, d\varepsilon \qquad (2.13)$$

em que:

$w(\varepsilon)$ - função peso ou de *Kernel* escolhida. Pode ser generalizada de modo a se tornar dependente do tempo e da solução, no caso de equações não-lineares;

τ - incremento no tempo;

ε - variável espacial auxiliar;

$f(x,t_0)$ - condição inicial;

$f(x,t_0+\tau)$ - solução desejada após o intervalo de tempo τ.

Por uma questão de simplicidade a equação (2.13) corresponde ao caso unidimensional. Assim, os métodos de integração diferem entre si pela forma como é definida a função $w(\varepsilon)$. É claro que, de modo a que o problema seja completamente enunciado, torna-se ainda necessário explicitar as devidas condições fronteira. Assim, através do uso repetido da equação (2.13) obtém-se a solução do problema de valor inicial até ao cálculo de $f(x,T)$ para o tempo $T = N.\tau$. É igualmente possível generalizar a equação para o caso de problemas mistos de valor inicial e problemas às condições fronteira.

Como foi referido anteriormente, é necessário converter o problema original para o domínio discreto. Assim, o integral de convolução na forma digital tem a forma:

$$f_j^{n+1} = \sum_{k=-K_M}^{k=+K_M} A_k \cdot f_{j+k}^{n} \qquad k = \pm 1,\ \pm 2,\ .....,\ \pm K_M \qquad (2.14)$$

onde:

A_k - uma função discreta análoga a $w(\varepsilon)$ e dependente do problema no domínio contínuo e do método de conversão utilizado. K_M é definido de forma a que A_{KM} ou A_{-KM} não sejam simultaneamente nulos;

k - índice de contagem com valor máximo K_M;

j - índice espacial de discretização;

n - índice temporal de discretização;

f_j^n - condições iniciais discretas;

f_j^{n+1} - solução após o passo temporal Δt.

Como anteriormente, é necessário definir as condições fronteiras apropriadas do problema.

O integral de convolução na forma digital é igualmente aplicável a algoritmos de integração do tipo implícito (através da introdução de uma segunda função-peso), assim como pode ser generalizado para problemas multi-dimensionais e com duas ou mais variáveis dependentes.

A forma como os coeficientes A_k são avaliados é substancialmente diferente para cada método de integração. Na Tabela 2.1 são apresentados os principais grupos de métodos de resolução de equações diferenciais, resumindo-se as diferentes estratégias de cálculo de A_k para cada um deles.

Tabela 2.1: Resumo dos grupos de métodos de integração principais[1].

Método Computacional	*Método de Cálculo dos Coeficientes da Função de Kernel A_k*
Diferenças Finitas	⇨ *Séries de expansão de Taylor.*
Elementos Finitos	⇨ *Polinómios de interpolação (lineares, quadráticos, cúbicos, etc) ou de Hermite e Funções Spline.*
Espectrais	⇨ *Funções ortogonais (Séries de Fourier, Legendre ou Polinómios de Chebishev).*
Esquemas de Filtro	⇨ *Transformada de Fourier da função de resposta do sistema de equações.*

Os métodos de **Diferenças Finitas** e de **Elementos Finitos** são, sem dúvida, os mais amplamente utilizados e estudados, obtendo-se uma maior informação bibliográfica do que em relação aos restantes.

Métodos de Diferenças Finitas

A raiz do cálculo dos coeficientes da função-peso A_k para os métodos de **Diferenças Finitas** é a expansão em série de *Taylor*. Por exemplo, considerando a sua expressão genérica:

$$f_{j+1}^{n} = f_{j}^{n} + \left(\frac{\delta f}{\delta x}\right)_{j}^{n} \cdot \Delta x + \left(\frac{\delta^2 f}{\delta x^2}\right)_{j}^{n} \cdot \frac{\Delta x^2}{2!} + \left(\frac{\delta^3 f}{\delta x^3}\right)_{j}^{n} \cdot \frac{\Delta x^3}{3!} + \ldots\ldots\ldots \qquad (2.15)$$

Assim, rearranjando a equação anterior obtém-se:

$$\left(\frac{\delta f}{\delta x}\right)_{j}^{n} = \frac{f_{j+1}^{n} - f_{j}^{n}}{\Delta x} - \left(\frac{\delta^2 f}{\delta x^2}\right)_{j}^{n} \cdot \frac{\Delta x^2}{2!} - \left(\frac{\delta^3 f}{\delta x^3}\right)_{j}^{n} \cdot \frac{\Delta x^3}{3!} + \ldots\ldots\ldots \qquad (2.16)$$

Considerando aproximações de primeira ordem, ou seja truncando a expressão (2.16) a partir da segunda derivada, tem-se então:

$$\left(\frac{\delta f}{\delta x}\right)_{j}^{n} = \frac{f_{j+1}^{n} - f_{j}^{n}}{\Delta x} \quad , \quad O_x(\Delta x) \qquad (2.17)$$

Aproximando desta forma todas as derivadas espaciais e temporais, converte-se o problema original num sistema de equações algébricas, a partir do qual, é calculada a aproximação da solução em cada nodo da malha. É nesta metodologia que se baseia o **Método das Linhas** que é, de facto, um dos métodos numéricos de integração de P.D.E.´s mais divulgados, utilizados e, consequentemente, analisados.

Através de manipulação das equações, é possível explicitar os coeficientes A_k correspondentes a este método, para cada posição j e n. Por outro lado, pode-se obter as expressões relativas à conversão para o domínio discreto de derivadas de diferentes ordens, considerando-se várias ordens de aproximação e números de pontos envolvidos, ou seja, vários valores de K_M. *Fornberg* [45,46] desenvolveu algoritmos recursivos para o cálculo do valor dos pesos de aproximação, no caso de malhas de espaçamento arbitrário.

Métodos de Elementos Finitos

O método de **Elementos Finitos** é considerado como pertencente ao grupo de métodos de **Resíduos Pesados**. Neste caso, os coeficientes A_k são definidos através de funções de interpolação usadas para calcular o valor das variáveis independentes em todas as posições não-nodais. A escolha das funções referidas pode variar consideravelmente desde relações lineares até aproximações quadráticas. Podem, igualmente, ser escolhidas expressões de ordem superior (Cúbicas, *Hermite* ou outras funções *Spline*). No entanto, estas requerem mais pontos nodais para a sua definição através de interpolação. Duma maneira geral, pode-se afirmar que os coeficientes de convolução são definidos pelo enunciado do problema no espaço contínuo e os graus dos polinómios de interpolação, através dos quais é realizado o processo de conversão. Esta perspectiva é semelhante à do método anterior, já que as expansões de *Taylor* estão intimamente relacionadas com polinómios de interpolação. Deste modo, as únicas diferenças entre os dois métodos referem-se a vantagens relativas a problemas específicos.

Os outros métodos de **Resíduos Pesados** mais comummente usados podem ser divididos nas classes seguintes:

- Métodos de Subdomínio;
- Métodos de Colocação (Função δ de *Dirac*);
- Métodos de Mínimos Quadrados;
- Métodos de Momentos;
- Métodos de *Galerkin*.

Obviamente que a distinção entre cada um destes métodos se relaciona com o tipo de função-peso escolhida. No entanto, não é o objectivo deste trabalho o aprofundamento da análise deste tipo de métodos.

Métodos Espectrais

Constituem a aplicação do método de *Galerkin* tradicional com funções de tentativa e teste com características ortogonais. Os coeficientes de convolução são calculados, neste caso, a partir de funções ortogonais (Ex: Séries de *Legendre*, Polinómios de *Chebishev*) que efectuam a conversão entre os domínios contínuo e discreto.

Métodos de Esquemas de Filtro

Este método baseia-se na noção de que qualquer *output* dum sistema linear ou quasi-linear pode ser calculado a partir do *input*, se a função resposta fôr conhecida. Neste caso, a

função-peso de conversão passa a ser a transformada de *Fourier* da função resposta do sistema.

Cada um dos quatro grupos de métodos de integração referidos anteriormente é aplicável a qualquer tipo de problema (hiperbólico, parabólico, elíptico ou misto). Porém, a lista apresentada não pretende, de modo algum, ser exaustiva ou demasiado pormenorizada mas, dar uma ideia das principais estratégias propostas e posteriormente desenvolvidas.

2.3.2 - Método das Características

É importante referir um outro tipo de método de integração, que constitui uma perspectiva algo diferente das referidas anteriormente. Nesta estratégia aproveita-se a definição de curva característica do sistema inicial de P.D.E.´s. Pode-se provar que a solução deste sistema, coincide com a solução de um sistema de O.D.E.´s ao longo de determinadas trajectórias no espaço, definidas pelas coordenadas independentes e que se denominam trajectórias ou curvas características. Deste modo, não é necessário recorrer à discretização do problema, pelo menos ao longo duma das coordenadas, não se procedendo a qualquer aproximação, ou seja, não ocorre perda de exactidão nessa direcção. Introduz-se, assim, uma perspectiva *Lagrangiana* na resolução das equações diferenciais, já que, neste caso, os nodos movimentam-se com uma velocidade determinada (velocidade característica) de forma a percorrerem as trajectórias características pretendidas.

Este método é bastante adequado para problemas caracterizados por frentes abruptas móveis e/ou descontinuidades, já que qualquer choque induzido no sistema se propagará ao longo das características, e nunca através destas, evitando-se gradientes elevados no sistema de equações integrado nessas trajectórias. No entanto, o método é pouco geral, ou seja, a sua aplicação tem de ser estudada para cada caso, além de se tornar difícil a avaliação das velocidades características dos nodos (além da integração das O.D.E.´s respectivas) em problemas de alguma complexidade.

2.3.3 - Robustez e Fiabilidade dos Métodos Numéricos

As propriedades definidas para avaliação adequada da performance de cada método numérico, aplicado a um determinado problema são:

- **Estabilidade**,
- **Consistência**,
- **Convergência**.

A estabilidade mede a propagação do erro entre dois passos de integração sucessivos, sendo o critério normalmente usado, a condição de *Von Neumann*.

O conceito de estabilidade pode ser facilmente compreendido através da análise da expansão numérica em série de *Fourier*. Assim, admitindo-se a solução inicial dum problema unidimensional num ponto p, na forma:

$$u_{p,0} = \sum_{n=0}^{N} A_n \cdot e^{i \cdot \beta_n \cdot p \cdot h} \quad , \qquad p = 0,\ 1,\ ....,\ N \tag{2.18}$$

em que:

A_n - amplitude;

β_n.p.h - fase de onda com $\beta_n = \frac{n \cdot \pi}{N \cdot h}$;

N - número de nodos;

h - passo espacial.

A resolução do sistema de N+1 equações permite a obtenção dos coeficientes A_i independentes do tempo. Para estimar o erro de propagação devido ao incremento temporal, considera-se que a solução num ponto p para o tempo q é calculada por:

$$u_{p,q} = e^{i \cdot \beta_n \cdot x} \cdot \xi_q \quad , \qquad p = 0,\ 1,\ ....,\ N \tag{2.19}$$

onde $\xi_q = e^{\alpha \times t}$ com α - constante complexa.

Para que a solução numérica seja estável, é condição necessária que $|\xi_q| \leq 1$, com q pertencente ao domínio temporal.

A consistência mede a influência do incremento das variáveis independentes no comportamento de tendência do erro de truncatura para zero. Por outras palavras, mede a tendência do modelo gerado pelo método para o modelo real.

Finalmente, a convergência mede a tendência da solução numérica para a solução analítica quando o incremento das variáveis independentes tende para zero. O critério de convergência mais utilizado é a condição de *Courant-Friedrich-Levy* que pode ser resumida da forma seguinte:

☞ A solução só é convergente no caso da curva característica da equação, num ponto genérico P para o tempo t^{n+1}, interceptar a recta correspondente ao nível t^n entre os pontos espaciais dos quais depende a fórmula de discretização utilizada.

Estas propriedades são muito importantes na escolha dos métodos numéricos a aplicar em cada problema, assim como determinam, frequentemente, os valores admissíveis para os parâmetros associados aos algoritmos de discretização.

2.3.4 - Dispersão e Dissipação Numéricas

No caso em que a solução numérica de um problema prático, envolvendo P.D.E´s ou sistemas de P.D.E.´s, apresente um comportamento caracterizado pela existência de perfis abruptos e/ou choques, é inevitável a ocorrência de problemas de difusão e dissipação numérica a quando da aplicação de qualquer algoritmo numérico. Estes factores podem ser entendidos através da análise de *Fourier*. Embora o factor de amplificação ξ_q (vd. Equação (2.19)) se mantenha, a amplitude A_i e a fase β_n.p.h (vd. Equação (2.18)) são alteradas pelo método numérico. A dispersão é provocada pelo facto da propagação de cada uma das ondas se efectuar com uma velocidade diferente devido ao desfasamento provocado. Assim, originam-se oscilações na solução numérica (vd. Figura 2.1). Por outro lado, a dissipação resulta da atenuação ou amplificação das ondas, o que provoca o alargamento da frente que se estende por uma região do domínio mais larga.

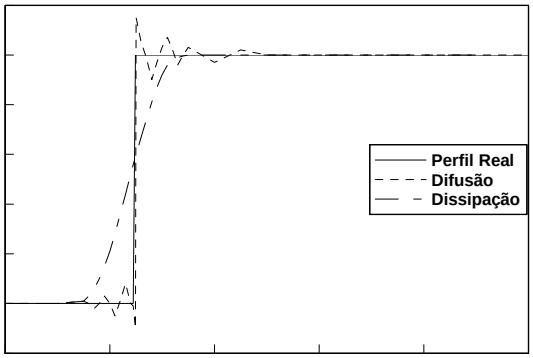

Figura 2.1: Representação de perfis típicos de difusão e dissipação numérica.

É impossível evitar simultaneamente a ocorrência destes dois fenómenos numa simulação numérica. No entanto, é possível, através da adequada parametrização do algoritmo, minimizar os seus efeitos, procurando, na medida do possível, encontrar as condições que evitem a excessiva predominância de um dos efeitos que conduzirá à obtenção de perfis de solução irrealistas.

3. Métodos de Malha Adaptativa

Neste Capítulo apresenta-se uma revisão bibliográfica dos **Métodos de Malha Adaptativa**, juntamente com as respectivas estratégias de mobilidade dos nodos. No entanto, não se pretende que esta revisão seja demasiado exaustiva, mas apenas que constitua uma introdução sucinta ao tema, através da descrição das principais soluções desenvolvidas e adoptadas para a resolução do problema.

A atenção deste estudo centra-se essencialmente nos métodos adaptativos associados a algoritmos de discretização baseados em **Diferenças Finitas**.

3.1 - Introdução

No Capítulo anterior são apresentados os principais métodos desenvolvidos para integração numérica de equações diferenciais (parciais ou ordinárias). No entanto, apenas são referidas estratégias gerais. É óbvio que, a partir de cada perspectiva, desenvolveram-se um grande número de algoritmos distintos, aplicando os conceitos básicos associados a cada método.

É referido igualmente, que o interesse deste trabalho se insere no estudo de P.D.E.´s caracterizadas por frentes móveis e abruptas. Normalmente, todos os esquemas desenvolvidos a partir de uma malha fixa, caracterizados pela constância da malha inicial, revelam-se satisfatórios no caso de problemas que apresentem apenas perfis suaves ou gradientes elevados confinados a zonas do espaço previamente conhecidas e fixas. Nestes casos, uma escolha adequada da malha inicial, concentrando nodos nas zonas críticas, revela-se suficiente para a obtenção da solução com uma exactidão aceitável, usando tempos computacionais razoáveis. No entanto, quando se estudam problemas que desenvolvem frentes móveis e/ou choques, a malha exigível para a implementação de um esquema de malha fixa é, na maioria dos casos, bastante fina. Deste modo, os tempos de computação necessários à obtenção da solução tornam-se incomportáveis. Assim, foram desenvolvidos métodos numéricos que, para além de possibilitarem a integração das equações, se preocupam igualmente em movimentar os nodos, concentrando-os nas zonas de maior actividade da solução (ou seja, nas regiões de gradientes mais elevados). Este grupo de métodos é denominado por **Métodos de Malha Adaptativa**, já que a malha é redefinida ao longo do tempo, adaptando-se às características da solução.

Os algoritmos de mobilidade de nodos podem ser associados a cada um dos métodos numéricos referidos anteriormente. No entanto, a informação recolhida na bibliografia, está geralmente relacionada com a aplicação dos métodos de **Diferenças Finitas** e **Elementos Finitos**, que são os métodos de utilização mais generalizada.

3.2 - Vantagens de Utilização dos Métodos de Malha Adaptativa

A grande vantagem deste tipo de métodos consiste no facto do movimento dos nodos possibilitar uma grande economia em termos de esforço computacional para problemas de

descontinuidades móveis, superfícies de contacto e correntes deslizantes, já que, nestes casos, apenas uma pequena parte do domínio necessita de pequenas separações entre os nodos. Ao contrário, para malhas fixas uniformes é preciso uma menor separação entre nodos de modo a se obter erros de truncatura aceitáveis em zonas de gradientes elevados. No entanto, estas separações são muito menores do que as necessárias para as regiões de baixo gradientes, provocando um grande desperdício de esforço computacional. De qualquer modo, desde que a solução não exiba essas características, os métodos de malha fixa revelam-se perfeitamente satisfatórios já que são menos complexos que os métodos de malha adaptativa.

A resolução apropriada de uma onda de choque requer que a separação nodal na sua vizinhança seja algumas vezes menor que a espessura do choque. Esta espessura está relacionada com o valor dos coeficientes associados a termos difusivos das P.D.E.´s do problema (segundas derivadas espaciais) sendo frequentemente da ordem de grandeza destes. Portanto, se se reduzir os coeficientes de segunda ordem, reduz-se também a espessura do choque. Para se evitar declives infinitos, ou seja, descontinuidades sem significado físico, os coeficientes terão de ser não-nulos. O uso de métodos adaptativos possibilita a utilização de separações nodais muito menores e consequentemente, a aplicação de coeficientes de segunda ordem menores (fisicamente, mais realistas) com ondas de choque bastante mais finas do que pode ser usado, geralmente, no caso de métodos não-adaptativos. É de notar, igualmente, que no caso em que os nodos se movimentem com a frente, podem ser introduzidos passos temporais consideravelmente maiores dos que são possíveis de aplicar para métodos de malha fixa não uniforme.

De modo a ilustrar estas vantagens dos métodos adaptativos, considera-se a solução da equação de *Burgers* unidimensional,

$$\frac{\delta u}{\delta t} = -u \cdot \frac{\delta u}{\delta x} + \varepsilon \cdot \frac{\delta^2 u}{\delta x^2} \qquad (3.1)$$

sujeita às condições fronteira: $\begin{cases} u(0,t) = u_L \\ u(1,t) = u_R \end{cases}$;

e à condição inicial: $u(x,0) = u^0(x)$.

Segundo *Herbst* **[61]** é necessário que na zona do choque o passo espacial h seja aproximadamente igual a ε para que a solução obtida seja estável. Caso contrário δu/δx torna-se oscilatória. Portanto, se se pretender a utilização de um método de malha fixa, é necessário que o espaçamento nodal em todo o domínio seja aproximadamente ε. Pelo contrário, através da aplicação de um método adaptativo, tal não se torna necessário, já que os nodos tenderão a concentrar-se junto ao choque, satisfazendo assim a condição de estabilidade ($h \cong \varepsilon$). Nas outras zonas do domínio, onde o perfil da solução é suave, a malha torna-se mais larga.

3.3 - Revisão Bibliográfica

3.3.1 - Introdução

As simulações numéricas de sistemas de P.D.E.´s consistem, geralmente, na aplicação de um gerador da malha e de um integrador das equações diferenciais. Em qualquer caso, é estabelecida uma ligação do gerador para o integrador. Esta ligação é feita quando a

integração é realizada através da grelha de pontos que cobre o espaço físico. No caso da solução desenvolver gradientes elevados, é necessário igualmente estabelecer uma ligação de comunicação do integrador para o gerador da malha. Assim, esta é alterada de acordo com as características da solução denominando-se adaptativa. Esta revisão bibliográfica pretende debruçar-se de um modo geral sobre as diversas estratégias adaptativas desenvolvidas.

Foram publicadas várias discussões gerais sobre este tipo de problemas tais como as apresentadas por *Thompson et al* [109,110], *Thompson* [107], e *Turkel* [114], que se referem igualmente a todo o campo de geração de grelhas de pontos. Uma grelha gerada numericamente é definida como um conjunto organizado de pontos formado pelas intersecções das linhas de um sistema de coordenadas curvilíneas de ajuste a condições fronteira. No entanto, o interesse deste estudo concentra-se sobre a movimentação dos nodos de modo a satisfazer propriedades da solução e não propriedades fronteira. Foram também apresentadas revisões mais especializadas sobre métodos adaptativos por *Anderson* [6], *Thompson* [108], *Eiseman* [38] e *Hawken et al* [59]. Este último trabalho foca a sua atenção em técnicas adaptativas aplicadas a métodos de integração baseados em **Diferenças ou Elementos Finitos**.

A maioria dos métodos adaptativos desenvolvidos requerem que as equações diferenciais envolvidas apresentem soluções contínuas. No entanto, se as equações forem não-lineares (que constituem o caso mais comum) tenderão a formar perfis descontínuos, e consequentemente, provocar problemas numéricos, se não se introduzirem termos de viscosidade ou análogos, na resolução numérica.

Os métodos adaptativos de movimentação de nodos tendem a apresentar várias características comuns, como [107]:

❶ Um método de ordenação e numeração dos nodos ao longo da região física de interesse;

❷ Um meio de "comunicação" entre os nodos de modo a que a sua distribuição permaneça relativamente regular à medida que se movimentam;

❸ Um meio de representação discreta da solução contínua e de avaliação dos valores discretos com uma exactidão aceitável;

❹ Uma medida do erro associado à discretização efectuada (erro de truncatura);

❺ Um meio de redistribuição dos nodos através da medida do erro, de modo a reduzir os erros cometidos no cálculo da solução numérica.

A maioria dos métodos adaptativos procedem à transferência das variáveis dependentes (vector **A**), descritas pelo sistema de P.D.E.´s, das coordenadas físicas iniciais ($\mathbf{X_i}$, t), para um espaço computacional de coordenadas ($\boldsymbol{\xi_i}$, t) no qual os nodos são igualmente espaçados. Deste modo, as incógnitas do sistema computacional são $\mathbf{A}(\mathbf{X_i}$, t) e $\mathbf{X_j}(\boldsymbol{\xi_i}$, t). Cada P.D.E. é transformada num conjunto de O.D.E.´s no tempo (uma para cada nodo) através da aplicação de um método de discretização (Ex: **Diferenças Finitas**, **Elementos Finitos**). Estas O.D.E.´s podem ser escritas na forma:

$$\frac{d\,A^k}{d\,t} = F^k(t) \qquad (3.2)$$

onde $\mathbf{A^k}$ - valor da solução no nodo k.

Os detalhes da computação de $\mathbf{F^k}$ são definidos pelo método de discretização particular escolhido, e contém os termos adicionais ($\delta \mathbf{X_i}$ / δt . $\delta \mathbf{A}$ / $\delta \mathbf{X_i}$) resultantes da transformação para o sistema coordenado computacional.

As O.D.E.´s podem ser integradas através de métodos de diferenças finitas com passo temporal, explícitos ou implícitos. Normalmente, a aplicação de métodos implícitos permite a utilização de passos no tempo bastante maiores que os métodos explícitos, sem causar instabilidade na solução. No entanto, requerem a resolução de equações matriciais, aumentando consideravelmente o tempo computacional de integração. De qualquer modo revelam-se essenciais para problemas que necessitem de aproximações mais apertadas entre os nodos.

Através da derivada temporal de $\mathbf{A^k}$ em cada nodo é possível calcular o valor de $\mathbf{A^k}$ para o próximo passo. Como (d$\mathbf{A^k}$ / dt) pode variar consideravelmente de nodo para nodo, provocando o aumento e decaimento da solução em zonas distintas do domínio, o sistema de O.D.E.´s é frequentemente *stiff*.

Os métodos de malha adaptativa dividem-se em dois grandes grupos:

➲ **Métodos de Redistribuição Nodal Estática**;

➲ **Métodos de Redistribuição Nodal Dinâmica**.

Os M.N.R.E. caracterizam-se pelo facto da malha ser redefinida em intervalos de tempo previamente fixados a partir de estimativas de erro local, com critérios de equidistribuição. Incluem dois tipos de abordagem distintos:

➲ ***Redistribuição Nodal*** ⇨ os nodos são posicionados em cada intervalo de tempo definido, mantendo-se constante o seu número.

➲ ***Refinamento da Malha*** ⇨ são adicionados ou suprimidos nodos em posições intermédias da malha inicial. Esta estratégia consiste simplesmente na relaxação ou refinamento da malha através do uso de nodos suplementares.

O algoritmo de obtenção da solução é completamente independente do algoritmo de redefinição da malha. No caso de alteração desta, efectua-se a transferência entre malhas, calculando-se os valores da solução na nova malha por interpolação, utilizando a informação proveniente da anterior.

Por outro lado, nos M.R.N.D. os dois algoritmos são interdependentes, avaliando-se a solução e a malha conjuntamente e continuamente ao longo do tempo. O movimento nodal pode seguir vários critérios, tais como:

⇨ Equidistribuição do integral de medidas de erro baseadas em propriedades características da solução;

⇨ Minimização do integral de indicadores de erro local;

⇨ Pseudoforças de atracção e repulsão entre nodos.

Uma desvantagem dos **Métodos Dinâmicos** é referida por *Dwyer et al* [36], baseando-se no facto da computação simultânea das posições dos nodos e da solução transformar um problema linear num não-linear ou, frequentemente, dificultar a resolução dum problema inicialmente não-linear. Porém, para problemas *stiff*, a utilização de **Métodos Estáticos** pode provocar instabilidade se os passos espaciais não forem bastante reduzidos.

3.3.2 - Métodos Numéricos de Malha Adaptativa

A maior parte dos métodos adaptativos tem sido desenvolvida para aplicação conjunta com métodos de integração de **Diferenças Finitas** ou **Elementos Finitos**. Para este último método, enunciado formalmente no princípio dos anos 60, foi apresentada a primeira versão adaptativa por *Miller e Miller* [83,84] em 1981, que foi denominada por **Método de Elementos Finitos Móveis** (M.E.F.M.). Esta formulação foi posteriormente alterada e melhorada por diversos autores, tendo sido realizados diversos trabalhos que incluem revisões dos principais métodos, nomeadamente por *Duarte* [29], *Johnson* [72] e *Hawken et al* [59]. O objectivo deste trabalho centra-se, no entanto, sobre as estratégias mais importantes aplicadas a métodos de **Diferenças Finitas** que são enumeradas em seguida.

3.3.2.1 - Métodos Adaptativos Aplicados a Problemas de Valor Fronteira (B.V.P.´s)

Os trabalhos iniciais no campo de malhas adaptativas de redistribuição estática foram aplicados a problemas de **Valor Fronteira** (B.V.P.´s) a dois pontos, sendo baseados nos trabalhos de *De Boor* [26] sobre colocação *spline* de ordem variável.

Pereyra e Sewell [89] apresentam uma análise da selecção de malhas para a solução de problemas B.V.P.´s baseada directamente no trabalho de *De Boor* [26]. A solução de um problema B.V.P. pode ser simbolizada por:

$$F(\underline{Y}) = 0 \qquad (3.3)$$

onde $\underline{Y}$ - vector de funções de uma variável real definida no intervalo finito [a,b].

Uma malha variável para a solução de (3.3) é definida como uma partição π de [a,b] que satisfaça:

$$a \le x_1 \le \quad \cdots \quad \le x_{N+1} \le b$$

com comprimentos de malha definidos por: $h_i = x_{i+1} - x_i$

Para qualquer método de diferenças finitas estável, o erro de truncatura satisfaz a relação:

$$\tau_i = h_i^n \cdot T(x_i) + O(h^{n+1}) \qquad i = 1, \ldots\ldots, N \qquad (3.4)$$

onde o vector-função T(x) é independente da partição π.

Deste modo, tenta-se definir uma malha com o menor número de pontos possível e que verifique:

$$\| \tau \|_{P,P} = \text{tol} \qquad (3.5)$$

sendo tol uma tolerância pré-definida.

Para a definição da partição π que satisfaça (3.5) *Pereyra e Sewell* [89] utilizam a expressão desenvolvida por *De Boor* [26]:

$$h_i^{l+1} \cdot \left| \frac{d^{l+1}u}{dx^{l+1}}(\xi_i) \right| = E \qquad i = 1,\ 2,\ \ldots..\ ,\ N-1 \qquad (3.6)$$

em que: $\xi_i =]x_i, x_{i+1}[$, $h_i = x_{i+1} - x_i$ e E - constante.

Obtém-se, assim, uma equidistribuição da norma $\| \tau \|_{P,P}$ da forma,

$$h_i \cdot \| \tau_i \|_P^P = \text{cte} = \tilde{E} \qquad i = 1,\ \ldots..\ ,\ N \qquad (3.7)$$

e portanto: $\tilde{E} = \dfrac{\| \tau \|_{P,P}^P}{N}$

No entanto, $\tilde{E}$ é dependente da partição π escolhida. No entanto, admitindo:

$$\| T(x) \|_P \le M_1 \qquad \text{e} \qquad \varepsilon = \frac{M_1}{k^{1/\sigma}} \qquad (3.8)$$

onde $\sigma = \dfrac{P}{(n \cdot P + 1)}$, π satisfaz a condição $\dfrac{h}{\min_i h_i} \le k$ e considerando a função g(x) de modo que:

$$g(x) = \max\left(\| T(x) \|_P , \varepsilon \right) \qquad \text{e} \qquad Y_i = h_i^n \cdot g(x_i) \qquad (3.9)$$

define-se uma malha assimptoticamente ou aproximadamente equidistribuida se:

$$h_i \cdot Y_i^P = E \cdot (1 + O(h)) \qquad i = 1,\ \ldots..\ ,\ N \qquad (3.10)$$

para uma constante positiva E, que é independente da partição π.

A introdução de Y_i evita a ocorrência de comprimentos de malha elevados e problemas numéricos associados a valores pequenos de $\| \tau_i \|$.

Denny e Landis [27] resolvem adaptativamente problemas B.V.P. a dois pontos por utilização de uma medida de erro calculada através dos termos principais de cálculo do erro de truncatura, obtidos por aproximações de diferenças finitas a três pontos da O.D.E.. Estes termos são diferenciados em relação à coordenada nodal de modo a se deduzir uma fórmula de diferenças finitas, para as posições nodais, que minimize o erro de truncatura. Os sistemas de

equações para a posição dos nodos e para a solução são resolvidos alternadamente. No entanto, este procedimento não concentra os pontos nas zonas de gradientes elevados, mas nas regiões onde os gradientes são menores, o que não é de modo nenhum desejável[108].

Gough et al [54] utilizam um algoritmo adaptativo para resolver B.V.P.´s a dois pontos. A O.D.E. unidimensional para o componente k da variável dependente é transformada do espaço físico x para o domínio ξ, caracterizado por uma separação nodal uniforme. A medida do erro é calculada através das derivadas de ordem m de A_k e x em relação a ξ:

$$E_m = \sum_{k=1}^{K} \left(\frac{1}{R_k} \cdot \left| \frac{d^m A_k}{d \xi^m} \right|^2 + \frac{\lambda}{x_f - x_0} \cdot \left| \frac{d^m x}{d \xi^m} \right|^2 \right) \qquad (3.11)$$

onde:

A_k - componente k do vector solução A;
$(x_f - x_0)$ e R_k - alterações máximas de x e A, respectivamente, em todo o domínio ξ;
λ - constante de peso (geralmente atribui-se-lhe o valor de K).

A transformação de x para ξ é obtida pela resolução da O.D.E. correspondente à minimização do integral de E_m para o domínio ξ. A discretização efectuada baseia-se em diferenças finitas centrais, sendo o sistema de equações obtido resolvido através do método iterativo de *Newton-Raphson*.

White [117] apresenta um método adaptativo para resolução de problemas B.V.P. a dois pontos, da forma:

$$u' = f(x,u), \qquad x \in [0,1]; \qquad b\big(u(0), u(1)\big) = 0 \qquad (3.12)$$

em que u, f e b são vectores. Este método baseia-se na definição de um critério de equidistribuição que requer, ao longo da malha, a satisfação da relação:

$$\xi(x_i) - \xi(x_{i-1}) = \lambda \qquad i = 1, \ldots\ldots, N \qquad (3.13)$$

para uma constante λ. A transformação de variáveis aplicada tem a forma:

$$\xi(x) = \frac{\int_{x_0}^{x} m(u,x)\, dx}{\theta} \qquad (3.14)$$

com $\theta = \int_{x_0}^{x_f} m(u,x)\, dx$

Deste modo, a medida de erro m(u,x) é equidistribuida ao longo do domínio ξ. As O.D.E.´s obtidas na grelha ξ são expressas na forma de diferenças finitas centrais. Utilizam-se várias tipos de funções monitor m, nomeadamente: a função comprimento de arco da solução $m(u,x) = \left[1 + \left\| f\big(u(x),x\big) \right\|_2^2 \right]^{1/2}$; erros de truncatura locais e alterações de nodo para nodo, da solução.

A transformação do problema B.V.P. inicial para as coordenadas do monitor $\xi(x)$ resulta no sistema de equações seguinte:

$$\begin{cases} \dfrac{d\,u}{d\,\xi} = \theta \cdot \dfrac{f(u,x)}{m(u,x)} \\ \dfrac{d\,x}{d\,\xi} = \theta \cdot \dfrac{1}{m(u,x)} \\ \dfrac{d\,\theta}{d\,\xi} = 0 \end{cases} \tag{3.15}$$

Obviamente torna-se necessário definir as condições fronteira adequadas, para o sistema transformado.

Num trabalho posterior, *White* **[118]** estende a aplicação do método a problemas dependentes do tempo, pela resolução do problema B.V.P. transformado pela equidistribuição da função monitor na malha regular, definida pelas coordenadas (ξ , t). As equações para x e θ neste sistema coordenado são:

$$\left|\frac{d\,x}{d\,\xi}\right| + \left\|\frac{d\,u}{d\,\xi}\right\|^2 - \theta^2 = 0 \tag{3.16}$$

e

$$\frac{d\,\theta}{d\,\xi} = 0 \tag{3.17}$$

As equações (3.16) e (3.17) são resolvidas juntamente com o sistema de P.D.E.´s do modelo, através de um método iterativo de *Newton-Raphson*, resultando num sistema não-linear de equações para a solução e a malha equidistribuida. Os valores das variáveis A, x e ξ utilizados em cada passo temporal correspondem aos valores médios entre o passo actual e o seguinte, enquanto que as derivadas em relação a ξ são aproximadas por diferenças centrais. Este algoritmo insere-se claramente no grupo de métodos adaptativos de redistribuição nodal dinâmica.

Ablow e Schechter **[2]** aplicam uma estratégia muito semelhante à de *White* **[117]**, usando a medida de erro:

$$E = 1 + \lambda \cdot |\Omega| \tag{3.18}$$

em que: λ - constante de peso;
Ω - curvatura circular da solução calculada por:

$$\Omega = \frac{d^2A \big/ d\,x^2}{\left[1 + \left[dA \big/ d\,x\right]^2\right]^{3/2}} \tag{3.19}$$

No entanto, os valores de A e x devem ser normalizados para aproximadamente a mesma magnitude. Desse modo, obtém-se uma segunda O.D.E. para além da correspondente ao problema B.V.P., no domínio ξ, através da fórmula de equidistribuição:

$$E^2 \cdot \left[\left[\frac{d\,x}{d\,\xi} \right]^2 + \left[\frac{d\,A}{d\,\xi} \right]^2 \right] = 1 \qquad (3.20)$$

As duas equações são resolvidas simultaneamente de forma similar ao proposto por *White* **[117]**.

Russell e Christiansen **[100]** desenvolveram uma estratégia de malha adaptativa para a solução de problemas B.V.P., baseado na equidistribuição de um monitor adequado. Para tal, as equações que definem as posições nodais na grelha, usam a informação obtida pelo cálculo mais recente da solução do problema. Portanto, os dois procedimentos são independentes, mas trocam informação entre si, sendo o processo de cálculo iterativo, repetindo-se até se obter convergência e se verificar a tolerância especificada. Os tipos de função monitor considerados são: comprimento de arco; erros de truncatura local e resíduos da solução do B.V.P. calculada por colocação.

Observa-se que quando o problema é resolvido desta forma, a solução é obtida em algumas regiões do domínio, antes do problema total ser completamente resolvido. Assim, os autores apresentam um processo de divisão do domínio em subregiões denominadas de "aceitáveis" e "não-aceitáveis". Após a solução de uma determinada partição ser considerada "aceitável", na iteração seguinte, as equações diferenciais apenas são avaliadas nas restantes partições do domínio, utilizando as condições fronteira obtidas pelo cálculo da solução nas regiões "aceitáveis". Deste modo, todo o esforço computacional é concentrado sobre as zonas catalogadas como "não-aceitáveis", em cada iteração. O processo de partição tem algumas limitações mas demonstra-se eficaz na redução do esforço computacional.

Chen **[24]** centra a sua atenção em algumas estratégias de adaptação de malha, de um ponto de vista matemático, desenvolvendo uma nova teoria para a equidistribuição do erro. Este método baseia-se na conjugação de estratégias de monitorização de erros de interpolação e de erros de truncatura locais, para a resolução de problemas B.V.P e, consequente extensão a modelos de valor inicial e fronteira uni- e bidimensionais. Os problemas B.V.P. considerados são definidos pelo operador não-linear:

$$N(u) = 0\,, \qquad x \in [a, b] \qquad (3.21)$$

numa malha de N+1 pontos Π da forma:

$$a = x_0 < x_1 < \quad \cdots \quad < x_N = b\,.$$

Os métodos de selecção da malha são associados a minimização de erros, e podem ser definidos pela dedução de uma função monitor positiva M(u,x), de modo a que a malha Π gerada para a variável x, corresponda a uma malha uniforme, nas novas coordenadas ξ. Assim, tem-se que:

$$\xi = \xi(x) = \int_a^x \rho(u, y)\, dy \qquad (3.22)$$

onde,

$$\rho(u,x) = \frac{M(u,x)}{\int_a^b M(u,y)\,dy} = \xi'(x) \tag{3.23}$$

As expressões (3.22) e (3.23) representam o princípio de equidistribuição proposto por *White* **[117]** e implicam que:

$$\xi(x_j) - \xi(x_{j-1}) = c_1 \equiv 1/N \qquad j = 1, \cdots, N \tag{3.24}$$

e consequentemente,

$$\int_{x_{j-1}}^{x_j} \rho(u,y)\,dy = c_1 \tag{3.25}$$

O autor compara duas perspectivas distintas para o desenvolvimento de estratégias de dedução dos monitores óptimos para a obtenção de uma malha, que possibilitem uma solução óptima. A primeira baseia-se na estimativa de erros de truncatura desenvolvida por *Pereyra e Sewell* **[89]**, apresentada anteriormente.

Por outro lado, é apresentado um método proposto por *Carey e Dinh* **[23]**, centrado na dedução de um monitor óptimo para polinómios de interpolação, de forma a se obter erros de interpolação mínimos. Assim, pretende-se a obtenção de um monitor óptimo M, que possibilite a minimização do erro e = u - v_h, sendo v_h a estimativa da solução calculada pela definição da semi-norma H^m de v:

$$|v|_m^2 = \int_a^b \left[v^{(m)}\right]^2 dx \tag{3.26}$$

Desta forma, obtém-se a desigualdade:

$$|e|_m^2 \le \sum_{i=1}^{N} \left(\frac{h_i}{\pi}\right)^{2\cdot(j+1-m)} \int_{I_i} \left[u^{(j+1)}\right]^2 dx \tag{3.27}$$

onde j ≥ m. Assim, para um monitor arbitrário que satisfaça a equação (3.25), prova-se que, sendo j = k:

$$|e|_m^2 \le \frac{1}{(\pi \cdot N)^{2\cdot(k+1-m)}} \int_a^b \frac{\left[u^{(k+1)}\right]^2}{[\rho]^{(2\cdot(k+1-m))}}\,dx\left(1 + O(h)\right) \tag{3.28}$$

Por minimização do termo da direita de (3.28) em função de ξ, obtém-se a solução da equação de *Euler*, que conduz ao monitor óptimo:

$$M = \left[u^{(k+1)}\right]^{2/\,[2\cdot(k+1-m)+1]} \tag{3.29}$$

Podem ser desenvolvidas generalizações ao método de *Carey e Dinh* [23] por aplicação de outras normas de erro, como por exemplo, as semi-normas de *Sobolev*-(l_1,l_2).

Através do aproveitamento destes dois tipos de estratégias, *Chen* [24] desenvolve um novo método para a estimativa do monitor, procurando aperfeiçoar as expressões de limite do erro a partir das quais são deduzidos os monitores anteriores. Para tal, considera a norma L_2 da qual obtém dois limites para o erro, que são minimizados simultaneamente. Assim, o primeiro limite é definido pela equação (3.18) com m = 0:

$$|e|_2^2 \leq \frac{1}{(\pi \cdot N)^{2\cdot(k+1)}} \int_a^b \frac{\left[u^{(k+1)}\right]^2}{[\rho]^{(2\cdot(k+1))}} dx\left(1+O(h)\right) \tag{3.30}$$

Inicialmente, introduz-se a estimativa:

$$\int_{x_{i-1}}^{x_i} e^2\, dx = c^2 \cdot h_i^{(2k+3)} \cdot T^2\left(u\left(x_{i-1/2}\right)\right)\left(1+O(h)\right) + t.o.e. \tag{3.31}$$

em que t.o.e. corresponde a termos de ordem elevada e portanto, desprezáveis.

Por outro lado, para uma função ξ com um monitor M, tem-se que:

$$h_i = \frac{1}{N \cdot \rho\left(x_{i-1/2}\right)} \cdot \left(1+O(h_i)\right) + t.o.e. \tag{3.32}$$

Deste modo, obtém-se o segundo limite definido por:

$$|e|_2^2 \leq \frac{c^2}{N^{2\cdot(k+1)}} \int_a^b \frac{T^2}{\rho^{(2\cdot(k+1))}} dx\left(1+O(h)\right) + t.o.e. \tag{3.33}$$

A minimização da combinação dos dois termos da direita das expressões (3.30) e (3.33), através de todas as funções admissíveis de ξ, é realizada por resolução da equação de *Euler*, obtendo-se a função monitor óptima:

$$M = \left\{\left[u^{(k+1)}\right]^2 + T^2\right\}^{1/\left[2\cdot(k+1)+1\right]} \tag{3.34}$$

3.3.2.2 - <u>Métodos Adaptativos de Redistribuição Nodal Estática</u>

Pierson e Kutler [92] resolveram problemas unidimensionais através da transformação das coordenadas físicas (X,t) para o domínio computacional (ξ,t), sendo a medida de erro definida pela terceira derivada da solução em relação a ξ. A discretização das P.D.E.´s é realizada por aplicação de diferenças finitas centrais e a integração temporal é efectuada por intermédio de um método de diferenças finitas implícito.

A movimentação dos nodos é executada após alguns passos temporais, de modo a minimizar o integral da medida de erro. A transformação das variáveis é obtida escrevendo X como um conjunto de séries finitas de *Chebishev* em ξ. Os coeficientes dos polinómios de *Chebishev* são calculados através da minimização referida anteriormente, sujeita as restrições

máximas e mínimas para o comprimentos da malha e resolvidos através de um método simplex.

Klopfer e McRae **[75]** utilizam para a resolução de problemas unidimensionais, uma medida de erro baseada no termo principal do erro de truncatura da P.D.E. transformada nas coordenadas computacionais (ξ,t). O espaçamento dos nodos é uma função linear da medida de erro E, de modo que:

$$\frac{\delta X}{\delta \xi} \alpha \left(1 + \frac{E}{E_{MAX}} \right)$$

O procedimento de movimento nodal é repetido após cada passo temporal. As derivadas temporais das coordenadas nodais X^k são calculadas a partir das alterações nas posições nodais, sendo incluídas nas P.D.E.´s da solução A. Introduz-se, igualmente, um esquema de viscosidade artificial de modo a evitar problemas de instabilidade. Por outro lado, devido ao uso do método preditivo-correctivo explícito de *MacCormack* **[80]**, impõe-se que o espaçamento mínimo seja superior a um décimo do máximo, de forma a evitar problemas excessivos de *stiffness*.

Sanz-Serna e Christie **[102]** apresentam uma estratégia adaptativa que aproveita o método de *White* **[118]** como ponto de partida. A função comprimento de arco S, é aproximada pelo somatório da quantidade $\Delta S = \left[[\Delta X]^2 + [\Delta A]^2 \right]^{1/2}$, em todos os nodos, onde ΔX e ΔA são as alterações na coordenada espacial e na solução entre nodos adjacentes, respectivamente.

As novas posições dos nodos são escolhidas de modo a que a alteração de S seja igual de nodo para nodo, sendo a solução transferida para a nova grelha por interpolação. Introduz-se, igualmente, uma limitação máxima para a razão de ΔX entre nodos adjacentes, pela especificação de um parâmetro β, de modo que:

$$[\Delta A]^2 \leq \beta \qquad (3.35)$$

em todos os intervalos. O procedimento de adaptação nodal é alternado com a operação de integração temporal que é efectuada através de um método implícito de diferenças finitas.

Revilla **[99]** apresenta um procedimento semelhante ao de *Sanz-Serna e Christie* **[102]** substituindo $[\Delta A]^2$ pelo valor absoluto da alteração de ΔA, entre intervalos adjacentes.

Dwyer e colaboradores **[30-36]** resolvem P.D.E.´s bidimensionais e dependentes do tempo através da transformação do sistema de coordenadas espaciais (X_1,X_2,t), para o sistema coordenado de equidistribuição do integral da medida de erro (ξ_1,ξ_2,t). Esta é definida em função de derivadas em relação ao comprimento de arco Z, ao longo de curvas no espaço físico, onde ξ_1 e ξ_2 se mantém constantes ($dZ^2 = [dX_1]^2 + [dX_2]^2$):

$$E(Z) = 1 + \lambda_1 \cdot \left| \frac{\delta A}{\delta Z} \right| + \lambda_2 \cdot \left| \frac{\delta^2 A}{\delta Z^2} \right| \qquad (3.36)$$

onde: λ_1 e λ_2 - coeficientes de peso.

Após a transformação das variáveis, a P.D.E. é resolvida através de métodos de diferenças finitas directos e implícitos, alternadamente. A transformação para cada curva de ξ_2 constante, no espaço físico, é dada pela expressão:

$$\xi_1(X_1, X_2, t) = \frac{\int_0^Z E(z)\,dz}{\int_0^{Z_{MAX}} E(z)\,dz} \tag{3.37}$$

Os nodos são mantidos estacionários entre vários passos temporais, até que a medida do erro seja calculada, já que se verifica a ocorrência de oscilações, se esta fôr tomada após cada passo. Deste modo, a equação (3.37) é resolvida através de quadratura numérica, definindo-se os valores de equidistribuição de ξ_1, sendo a solução transferida para a nova grelha por interpolação. Pode ser aplicado um procedimento semelhante baseado na transformação em relação a curvas de ξ_1 constante.

Apesar das versões do método aplicadas a problemas unidimensionais se terem revelado satisfatórias, para o caso bidimensional podem ocorrer oscilações na solução devido ao desenvolvimento de *skewness* crescente em algumas regiões da malha. No entanto, *Dwyer et al* **[33]** sugerem que o processo de *Potter e Tuttle* **[94]** pode ser usado para reduzir esse problema. *Potter e Tuttle* **[94]** utilizam um movimento nodal ao longo de linhas de ξ_1 constante, de forma definir as curvas de ξ_2 constante nas quais uma função especificada de ξ_1 satisfaz a equação de *Laplace*. Assim, o sistema coordenado curvilíneo gerado é ortogonal. *Dwyer et al* **[33]** depararam-se igualmente com problemas, em casos bidimensionais, quando a adaptação é permitida nas fronteiras.

Brackbill e Saltzman **[18,19]**, *Brackbill* **[17]** e *Saltzman e Brackbill* **[101]** procedem à extensão dum método proposto por *Winslow* **[122]**, de geração de grelhas para problemas adaptativos fronteira, através da transformação dum problema singular, das coordenadas físicas (X_1,X_2,t) para o sistema numérico (ξ_1,ξ_2,t). Esta transformação é baseada numa medida de erro definida por:

$$E = \lambda_S \cdot E_S + \lambda_0 \cdot E_0 + \lambda_V \cdot E_V \tag{3.38}$$

com λ_S, λ_0 e λ_V - constantes de peso da ordem da unidade.

Deste modo, a transformação de coordenadas é determinada pela soma pesada de três contribuições (ou medidas) distintas:

⇨ A medida de suavidade da transformação:

$$E_S = \left[\nabla\xi_1\right]^2 + \left[\nabla\xi_2\right]^2 \tag{3.39}$$

⇨ A medida de ortogonalidade da grelha no espaço físico:

$$E_0 = \left[\nabla\xi_1 \cdot \nabla\xi_2\right]^2 \qquad \text{ou} \qquad E_0 = \left[\nabla\xi_1 \cdot \nabla\xi_2\right]^2 \cdot J^3 \tag{3.40}$$

A segunda expressão de (3.40) contribui para a atribuição de maior peso às zonas em que o Jacobiano da transformação apresente um valor elevado. Estas regiões correspondem a maiores separações nodais no domínio físico, sendo J calculado por:

$$J = \frac{\delta X_1}{\delta \xi_1} \cdot \frac{\delta X_2}{\delta \xi_2} - \frac{\delta X_1}{\delta \xi_2} \cdot \frac{\delta X_2}{\delta \xi_1} \tag{3.41}$$

⇨ A terceira medida E_V é definida por:

$$E_V = W \cdot J \tag{3.42}$$

onde W é uma medida de erro de truncatura da solução obtida por,

$$W = \left| \frac{\nabla A}{A} \right|^Q \tag{3.43}$$

em que Q é um parâmetro que, na prática, toma o valor de dois ou quatro. O Jacobiano da transformação tende a ser reduzido em zonas de W elevado se E_V fôr equidistribuido. Portanto, o refinamento da malha ocorrerá em zonas caracterizadas por medidas de erro elevadas.

Os nodos são mantidos estacionários entre vários passos de avaliação da solução A das P.D.E.´s transformadas. Para tal, utiliza-se um método de diferenças finitas explícito, enquanto que o sistema de O.D.E.´s para as coordenadas espaciais é resolvido por iteração de *Jacobi*. Este sistema é obtido pela minimização do integral:

$$\int E \, dV = \iint E \cdot J \, d\xi_1 \, d\xi_2 \tag{3.44}$$

através de cálculo variacional. O integral de volume de E é transformado para as coordenadas numéricas. A transferência da solução da grelha anterior para a nova é realizada por interpolação ou pela resolução de uma equação de transporte, na forma conservativa.

Kreis et al **[76]** concluíram que a performance do método de *Brackbill e Saltzman* **[18,19]** pode ser melhorada, em problemas específicos, através da substituição da equação (3.42) pela relação seguinte, no cálculo da medida do erro:

$$E_V = W \big/ J \tag{3.45}$$

Bell e Shubin **[13]**, aplicam um método de grelha adaptativa a problemas unidimensionais, inspirado na formulação variacional da geração elíptica de grelhas, proposta por *Saltzman e Brackbill* **[101]**. Assim, o objectivo consiste na minimização do integral:

$$\int_0^1 \left[x_\xi^2 + \lambda_a \cdot w(\xi) + \lambda_b \cdot \left(x_\xi - x_\xi^* \right)^2 \right] d\xi$$

onde λ_a e λ_b - parâmetros de peso;

$w(\xi)$ - função de concentração nodal definida de forma a verificar a relação:

$$0 \le w(\xi) \le 1$$

x_ξ^* - valor de x_ξ no passo temporal anterior.

As derivadas x_ξ são aproximadas por diferenças finitas. O primeiro termo promove a equidistribuição de Δx em todo o domínio, enquanto o segundo tende a diminuir Δx em zonas de $w(\xi)$ elevado. Finalmente, procura-se, com a introdução do terceiro termo, manter uma variação suave da grelha, ao longo do tempo.

De modo a determinar numericamente $x(\xi)$, considera-se primeiro a equação de *Euler* para o integral:

$$\left(1+\lambda_a \cdot w+\lambda_b\right) \cdot w_{\xi\xi} + \lambda_a \cdot w_\xi \cdot x_\xi - \lambda_b \cdot x_{\xi\xi}^* = 0 \qquad (3.46)$$

Através de um esquema de diferenças finitas e definindo as condições fronteira adequadas, obtém-se um sistema linear tridiagonal, para o cálculo das posições nodais. É igualmente possível efectuar a integração directa de (3.46), obtendo-se a relação:

$$x(\xi) = \int_0^1 \frac{k+\lambda_b \cdot x_\xi^*(\xi)}{1+\lambda_a \cdot w(\xi)+\lambda_b} d\xi \qquad (3.47)$$

onde k é definido de forma a que $x(\xi = 1) = 1$.

A função $w(\xi)$ é escolhida de modo a assumir valores elevadas em zonas onde se pretendem comprimentos de malha Δx reduzidos. Para tal, normalmente, utiliza-se a técnica de equidistribuição de uma medida de erro que, neste caso, é definida por,

$$E = \int_0^1 \left| u_h(x) - u_e(x) \right| dx \qquad (3.48)$$

em que:

u_h - solução aproximada (consistente com os esquemas de diferenças finitas aplicados);

u_e - solução exacta.

As funções consideradas são: $w(\xi) = u_\xi^2$; $w(\xi) = |u_\xi|$. No entanto, a experiência numérica demonstrou uma melhor performance do método para o segundo caso.

Os autores estudam a aplicação de dois métodos de discretização das equações diferenciais: esquema de *Godunov*, um método de viscosidade artificial.

Verwer et al **[116]**, seguindo um trabalho anterior de *Blom et al* **[16]**, apresentam um método que consideram intermédio entre as perspectivas de redistribuição nodal estática e dinâmica. O avanço da solução na grelha, para problemas unidimensionais, é executado por um esquema de *Lagrange* de passo temporal, numa malha trapezoidal espaço-tempo. A estratégia divide-se em dois estágios:

❶ ***Estágio Predictivo*** - Procede-se à computação da grelha do nível de tempo seguinte (n+1), através de um passo de *Euler* implícito, numa malha fixa. A solução obtida é

introduzida no algoritmo de redistribuição de *De Boor* [26], que gera os pontos da grelha, através da equidistribuição de uma função monitor escolhida. Esta equidistribuição define implicitamente a transformação de coordenadas x = x(ξ,t).

❷ ***Estágio de Integração*** - Efectua-se o cálculo da aproximação da solução do nível de tempo n+1, através de um esquema de *Cranck-Nicholson Lagrangiano*, utilizando as novas posições nodais.

A transformação das variáveis é baseada na expressão proposta por *White* [117], recorrendo ao monitor de segunda derivada:

$$M(\xi, t) = \alpha + \sqrt{|u_{xx}(\xi, t)|} \tag{3.49}$$

Através da equidistribuição de M, a transformação inversa x = x(ξ,t). apresenta a forma:

$$\int_{x_i}^{x_{i+1}} M(\xi, t)\, d\xi = \theta(t) \cdot \left[\xi(x_{i+1}, t) - \xi(x_i, t)\right] = \theta(t)/m \qquad 0 \le i \le m - 1 \tag{3.50}$$

Adicionalmente, *Verwer et al* [116] propõem a aplicação de algoritmos de passo temporal e número de nodos (m) variáveis, de forma a aumentar a eficiência da integração. Para tal, desenvolveram uma estratégia de tentativa e erro para avaliação do passo máximo utilizável, baseado na avaliação de erros de truncatura locais. Por outro lado, procedem à introdução de uma função monitor de erro espacial, para o cálculo de m em cada passo. Assim, m é definido da forma:

$$m = \left[m_{var}\right] + m_{flat} + 1 \tag{3.51}$$

onde

$$m_{var} = \frac{\int_{x_L}^{x_R} \sqrt{|u_{xx}|}\, d\xi}{\sqrt{TOLS}} \tag{3.52}$$

e m_{flat} - parâmetro pré-definido;
TOLS - tolerância especificada.

Deste modo, fixando $\alpha = \dfrac{m_{flat} \cdot \sqrt{TOLS}}{x_R - x_L}$, a propriedade de equidistribuição implica que:

$$\int_{x_i}^{x_{i+1}} \sqrt{|u_{xx}(\xi, t)|}\, d\xi = \theta(t)/m - \left(x_{i+1} - x_i\right) \cdot \alpha \le \theta(t)/m \le \sqrt{TOLS} \tag{3.53}$$

A definição de m pela equação (3.51) garante que a quantidade $(\Delta x)^2.|u_{xx}|$ seja inferior à tolerância TOLS em todos os subintervalos.

Métodos Adaptativos de Refinamento da Malha:

Hyman e Naughton **[70]** apresentaram um método de refinamento da malha que implementa um critério de equidistribuição, para a resolução de problemas gerais caracterizados por sistemas de P.D.E.´s da forma:

$$F(u, x, t) = 0 \tag{3.54}$$

Juntamente com as condições iniciais e fronteiras apropriadas é definido um monitor de erro de truncatura local. Esta função mede a qualidade da aproximação numérica do operador diferencial F. A expressão de cálculo do erro de truncatura envolve derivadas da função u, que são aproximadas por técnicas de diferenças finitas, podendo, no entanto, ser bastante complexas para o caso de problemas não-lineares. De forma a evitar a avaliação directa do erro de truncatura, *Hyman e Naughton* **[70]** comparam as aproximações ao operador F obtidas numa grelha fina e noutra mais larga, utilizando para tal um processo de extrapolação de *Richardson*. Assim, a função monitor é definida por:

$$m^0_{i,j} = \frac{\left| F_h(x,t)_{i,j} - F_{2h}(x,t)_{i,j} \right|}{\|F\|} \tag{3.55}$$

onde: i - contador espacial;
j - contador temporal;
F_h - aproximação numérica a F em (x_i , t_i) utilizando a grelha fina;
F_{2h} - aproximação numérica a F em (x_i , t_i) utilizando a grelha larga.

O monitor m^0 representa a exactidão da aproximação de F, mas não indica necessariamente se a precisão da aproximação da solução u é aceitável ou se a variação dos comprimentos de malha adjacentes é suficientemente suave. Portanto, é necessário definir um monitor de solução m^S e um monitor de regularidade de malha m^R. O primeiro pode ser escolhido como um monitor de comprimento de arco e a regularidade da malha é verificada por fixação de limites (máximo e mínimo) na variação do comprimento da malha ou obrigando a relação entre comprimentos de malha adjacentes a não exceder um valor pré-determinado. Deste modo, a malha adaptativa é gerada por equidistribuição de uma combinação linear dos monitores referidos, da forma:

$$m = \alpha(x,t) \cdot m^0 + \beta(x,t) \cdot m^S + \gamma(x,t) \cdot m^R \tag{3.56}$$

A escolha dos parâmetros α, β e γ depende dos objectivos do cálculo, podendo variar espacialmente. Através da equação (3.56) constrói-se uma malha aproximadamente equidistribuida que, em cada ponto (x_i , t_i) satisfaz a inequação:

$$m^- \le m_{i,j} \le m^+ \tag{3.57}$$

sendo: $m^+ = \overline{m} \cdot r$, $m^- = \overline{m} / r$ e

$\overline{m}$ - valor médio do monitor em todo o domínio;
r - parâmetro constante.

A equidistribuição é implementada partindo de uma malha inicial larga e uniforme, com posterior identificação das zonas da malha onde a relação (3.57) não é verificada. Estas são sucessivamente refinadas até que todo o domínio satisfaça a condição (3.57).

Berger e Oliger **[15]** utilizam um algoritmo de refinamento de grelha para o cálculo da solução de problemas transientes bidimensionais, descritos por P.D.E.´s hiperbólicas, através de técnicas de diferenças finitas. A malha de base é refinada por sobreposição sucessiva de submalhas de forma rectangular, localmente uniformes e com orientação arbitrária. A nova grelha gerada não tem necessariamente que coincidir com as anteriores, como acontece na maioria dos algoritmos de refinamento.

O critério de refinamento baseia-se na estimativa do erro de truncatura local a partir da técnica proposta por *Hyman e Naughton* **[70]**. O algoritmo de *Berger e Oliger* **[15]** é adaptativo no espaço e no tempo, mantendo constante a razão entre os passos temporal e espacial em todas as grelhas. Esta estratégia de decomposição do domínio permite a utilização de diferentes aproximações de diferenças finitas para submalhas distintas. No entanto, a maior desvantagem deste algoritmo reside no facto da técnica de divisão do domínio resultar na manipulação de estruturas de dados bastante complexas para as zonas de intercepção de grelhas.

Gropp **[57]** apresenta um método de combinação entre um algoritmo de refinamento de malhas localmente uniformes (L.U.M.R.) e estratégias de malhas móveis, baseado nas ideias de *Berger e Oliger* **[15]** e principalmente de *Gropp* **[56]**, já que, neste caso, não se permite a orientação arbitrária das grelhas. O método desenvolvido, denominado por algoritmo L.U.M.R. móvel (M.L.U.M.R.) é aplicado a problemas bidimensionais.

O algoritmo de refinamento gera uma série de níveis de grelhas de pontos G_l, em que a inicial (G_0) coincide com o domínio espacial do problema. Cada nível é constituído por grelhas rectangulares (G_{li} para a grelha i do nível l). Por outro lado, se uma grelha G_{li} se insere em $G_{l-1,j}$ então todas as G_{li} se inserem em $G_{l-1,j}$ e nenhuma se situa em qualquer outra grelha do nível l-1. Assim, a grelha $G_{l-1,j}$ é designada por *parent* de G_{li}, e consequentemente G_{li} será *child* de $G_{l-1,j}$. Uma estrutura de grelha simples é apresentada na Figura 3.1.

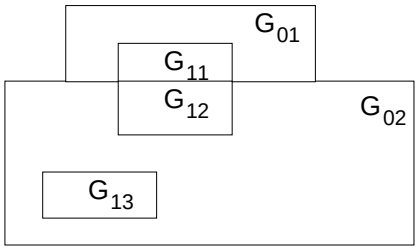

Figura 3.1: Esquema de grelhas com um nível de refinamento crescente.

Cada grelha contém uma malha uniforme de pontos com separação espacial h_l e consequentemente passo temporal k_l. Como é óbvio, há situações onde os mesmos pontos são definidos mais do que uma vez para grelhas distintas (p. ex. pontos fronteira). No entanto, a aplicação do algoritmo assegura que a estes pontos apenas seja atribuído um valor da solução. A cada grelha é atribuída, igualmente, uma velocidade de deslocamento, com a qual todos os pontos se movimentam uniformemente.

O algoritmo de redefinição da malha é aplicado entre vários passos temporais no nível l, por verificação da necessidade de introdução de malhas mais finas. Este procedimento é realizado em três passos:

❶ Marcação dos pontos da malha de nível l onde é necessário proceder a refinamento. A selecção é realizada por estimativa do erro de truncatura local nos pontos ou por outro método *ad hoc* baseado no comportamento da solução.

❷ Rodear as regiões marcadas por zonas de tampão, de forma a isolar o interior das grelhas refinadas, das grelhas mais largas. Este passo é necessário devido à estratégia de sobreposição das grelhas, permitindo a introdução de um maior número de passos temporais entre cada definição da malha.

❸ Análise dos pontos marcados de maneira a determinar as novas grelhas refinadas, que são inicializadas a partir dos valores das anteriores.

Para o cálculo das velocidades das grelhas, *Gropp* **[57]** usa um método não automático que solicita ao utilizador a especificação desta em função dos valores e da posição da grelha.

Cada nível consiste assim, num conjunto de grelhas que, possivelmente, se interceptam entre si. As fronteiras entre uma grelha refinada e a sua *parent* são obtidas por interpolação linear em relação aos valores anteriores.

A partir do trabalho de *Berger e Oliger* **[15]**, *Berger e Colella* **[14]** desenvolvem um método adaptativo de refinamento de malha (A.M.R.). Este método baseia-se, como em métodos anteriores, na utilização de uma sucessão de grelhas rectangulares encaixáveis entre si, nas quais as P.D.E.´s são discretizadas. Para isso são definidos, tal como em *Gropp* **[56]**, uma série de níveis l = 1, ... , l_{MAX} onde cada grelha G_{ik} apresenta um espaçamento h_i e é definida por:

$$G_l = \bigcup_k G_{lk} \tag{3.58}$$

No caso particular do primeiro nível (l=1), tem-se que $G_1 = \bigcup_k G_{1k} = D$, sendo D o domínio do problema. O facto das novas grelhas se encaixarem mutuamente, não implica que não possa existir sobreposição destas, ou seja, para o mesmo nível:

$$G_{lj} \cap G_{lk} \neq 0 \qquad j \neq k \tag{3.59}$$

Por outro lado, as grelhas correspondentes a diferentes níveis devem ser “devidamente encaixáveis”, o que significa:

⇨ Uma grelha fina começa e acaba nos cantos de uma célula da grelha mais larga do nível acima.

⇨ Deve existir pelo menos uma célula do nível l-1, de uma grelha l-1, a separar uma célula de nível l de outra de nível l-2 em todas direcções cardiais, a não ser que a célula se situe na fronteira física do domínio.

O refinamento da grelha é efectuado quer em relação ao espaço, como ao tempo, de modo a se manter constante a razão entre os passos.

Em cada uma das grelhas aplica-se um esquema de diferenças finitas explícito. Se não houver necessidade de refinamento, o processo consistirá na simples integração temporal numa única malha. Caso contrário, cada grelha é definida separadamente (tendo o seu próprio vector-solução) e é avançada no tempo independentemente das outras, exceptuando a definição das condições fronteira respectivas. No entanto, o avanço da integração para o tempo $t+\Delta t$ só é efectuada após todas as grelhas mais finas terem sido integradas até o tempo t.

O critério de refinamento é aplicado após intervalos de tempo especificados e baseia-se em estimativas de erro de truncatura locais. Portanto, este procedimento indica uma lista dos pontos da malha larga com estimativas de erro mais elevadas, assinalando as regiões onde é necessária uma malha mais fina.

Bell et al **[12]** apresentam um aperfeiçoamento do algoritmo A.M.R. de *Berger e Colella* **[14]** para problemas hiperbólicos tridimensionais. Para tal, alteram a estratégia de refinamento da malha, essencialmente em duas vertentes:

☞ **Geração da grelha** ⇨ O procedimento simples de bissecção na direcção mais longa das células "inaceitáveis" é substituído por um algoritmo que procura saliências nas zonas de transição entre as zonas marcadas e não marcadas. Considerando o caso bidimensional, por razões de simplicidade, definem-se, em regiões de pontos marcados, as funções Σ_x e Σ_y, associadas a cada uma das direcções coordenadas:

$$\Sigma_x = \int_y f(x,y)\,dy \tag{3.60}$$

e

$$\Sigma_y = \int_x f(x,y)dx \tag{3.61}$$

onde f - função contínua.

Se qualquer uma das quantidades Σ_x e Σ_y contém um valor nulo numa determinada direcção, o rectângulo que define a região pode ser fraccionado nessa direcção. Se tal não acontecer, pode-se definir uma saliência através do posicionamento do cruzamento de maior amplitude do zero, das segundas derivadas das funções Σ_x e Σ_y. No caso da existência de mais que um cruzamento de máxima amplitude, escolhe-se o mais próximo do centro do rectângulo original.

☞ **Estimativa do Erro** ⇨ Neste caso, os autores procedem à adição de um termo de origem espacial no processo de estimativa do erro de truncatura, obtido por extrapolação de *Richardson*. De facto, a medida de erro adicional poderá identificar estruturas anteriormente não consideradas.

Arney e Flaherty **[10]** desenvolvem um algoritmo de refinamento baseado no trabalho de *Berger e Oliger* **[15]**. Os autores pretendem combinar um esquema de refinamento celular, sem a definição de submalhas de orientação arbitrária, com um algoritmo de movimentação da malha base inspirado em *Arney e Flaherty* **[11]**, o que promoverá a combinação entre o refinamento da malha e o fenómeno dinâmico. Outra diferença em relação ao método de

Berger e Oliger [15] consiste no facto das submalhas geradas apresentarem a forma poligonal e não rectangular.

O processo de introdução das submalhas finas divide-se em quatro passos:

❶ Localização para cada grelha de nível l dos nodos onde o indicador de erro ultrapasse a tolerância estabelecida (nodos "intoleráveis").

❷ Associação dos nodos "intoleráveis" nas zonas respectivas.

❸ Isolamento das zonas referidas por regiões tampão de forma a reduzir problemas com as condições iniciais e fronteiras nas interfaces das malhas finas e largas.

❹ Refinamento celular das regiões assinaladas.

As submalhas são obtidas por bissecção do passo temporal e dos lados de cada célula da grelha original que intercepte a região "inaceitável", tornando-se deste modo numa malha espaço-tempo poligonal.

A integração é efectuada através de um esquema de diferenças finitas de *MacCormack* [80] sendo os indicadores de erro avaliados por extrapolação de *Richardson*.

Trompert e Verwer [112] propõem um algoritmo estático de refinamento semelhante aos anteriores, para a resolução de P.D.E.´s parabólicas bidimensionais. Este método pertence à classe de algoritmos de malha localmente uniforme (L.U.M.R.). Deste modo, a partir de uma malha inicial, são geradas sucessivamente subgrelhas cada vez mais finas nas regiões de maior actividade espacial, através da bissecção das células das grelhas anteriores (refinamento celular). Assim, inicializa-se um novo problema de valor inicial e fronteira para cada subgrelha, procedendo-se à integração de maneira consecutiva desde as grelhas de maior até às de menor nível. Os valores iniciais são definidos por interpolação dos valores de grelhas do nível anterior (ou seja, grelhas com grau de refinamento imediatamente mais baixo) ou, quando possível, de subgrelhas do passo temporal anterior. As fronteiras internas das regiões assinaladas são tratadas como fronteiras de *Dirichlet* e os valores são interpolados a partir da submalha anterior. A geração das submalhas é controlada pela estimativa de erros de truncatura locais e, consequentemente, pela especificação da tolerância pretendida.

Portanto, o esquema não difere muito de outros já referidos, podendo resumir-se nos passos seguintes:

❶ Integração da grelha mais larga de base até ao tempo $T+\Delta t$; o passo Δt máximo é obtido por um teste de erro temporal.

❷ Redefinição da grelha através da estimativa do indicador de erro utilizando o valor da solução obtido no passo de integração; as regiões "não-aceitáveis" são assinaladas e isoladas, enquanto que a integração das zonas não refinadas é considerada completa até ao nível de tempo $T+\Delta t$.

❸ Retorno ao tempo T e procede-se à interpolação dos valores da solução inicial para a grelha mais fina obtida na redefinição; torna-se necessário especificar as condições fronteira entre grelhas de níveis diferentes.

❹ Integração da grelha fina até $t \leq T+\Delta t$; poderá ser necessário proceder a nova adaptação temporal com um valor do passo inferior a Δt; quando $T+\Delta t$ fôr atingido, as soluções nos pontos da nova grelha fina são injectados na grelha anterior, obtendo-se uma nova grelha base para o passo temporal seguinte.

Como foi referido anteriormente, a grande desvantagem deste tipo de métodos consiste na grande complexidade das estruturas de dados entre cada nível de refinamento, associadas a crescentes divisões do domínio e à constante redefinição das posições nodais, das soluções respectivas e conjugação dos vários níveis de grelhas. Deste modo, torna-se inevitável o armazenamento de enormes quantidades de dados.

Num trabalho posterior, *Trompert e Verwer* **[113]** aplicam um algoritmo bastante semelhante ao mesmo tipo de problemas. No entanto, neste caso, a preocupação centra-se no estudo da integração temporal pela utilização dum método implícito de *Euler*, em substituição do método explícito de *Runge-Kutta-Chebyshev* usado em **[112]**. Além disso, *Trompert e Verwer* **[113]** procedem a uma análise de erro detalhada de forma a demonstrar a estabilidade e convergência do método para alguns tipos de P.D.E.´s.

Lee e Tsuei **[79]**, inspirados por um trabalho de *Acharya e Monkalled* **[3]** apresentam um método híbrido de combinação entre os conceitos de refinamento local e de movimento global da grelha. De forma a demonstrar a sua utilidade, o método desenvolvido é aplicado a problemas bidimensionais de fluxo incompressível de recirculação laminar.

Ao contrário de *Acharya e Monkalled* **[3]**, os autores efectuam o assinalamento das regiões com base na solução adaptativa, em vez da solução da grelha inicial. Deste modo, obtém-se uma diminuição das regiões de erros elevados e uma redução do número necessário de níveis de refinamento. Por outro lado, *Acharya e Monkalled* **[3]** utilizam uma distribuição de funções de peso para estimar a medida de erro, em substituição do método de extrapolação de *Richardson*. Além disso, aplicam um procedimento mais sofisticado para reconhecer as regiões a refinar, o que origina a formação de estruturas de dados mais complexas. Por fim, o método de geração de grelhas baseia-se num princípio variacional, através da resolução de duas equações de *Poisson* para obtenção da grelha adaptativa nas regiões refinadas. A aplicação deste procedimento pode-se tornar problemática para casos tridimensionais, onde a poupança do esforço computacional é um factor importante a ter em conta.

Portanto, *Lee e Tsuei* **[79]** usam uma estratégia de movimentação da grelha baseada na equidistribuição de uma função peso W_i calculada em relação aos gradientes das propriedades do fluxo. A função peso é definida por modificação da utilizada por *Dwyer et al* **[36]** e, para a direcção x, tem a forma,

$$W_{i,j} = 1 + \sum_{N} b_N \cdot \left[\left| \phi_\xi \right|_{i,j} + \sum_{k=1,k\neq i}^{m} \left| \varphi_\xi \right|_{k,j} \cdot x \cdot C_t \cdot e^{(-|i-k|)} + \sum_{l=1,l\neq j}^{n} \left| \phi_\xi \right|_{i,l} \cdot x \cdot C_t \cdot e^{(-|j-l|)} \right] \tag{3.62}$$

em que: b - parâmetro de peso;
N - número dos gradientes de propriedades consideradas;
ϕ_ξ - gradiente da propriedade ϕ ao longo da coordenada ξ;
C_t - coeficiente de acoplamento.

Assim, o algoritmo de refinamento híbrido pode ser dividido em duas fases distintas: a fase de obtenção da solução numa grelha inicial adaptativa gerada pela equidistribuição da função definida em (3.62), para todo o domínio; a fase de refinamento das regiões

caracterizadas por indicadores de erro elevados e consequente integração do modelo, a partir da nova grelha. O procedimento de refinamento é bastante semelhante aos referidos anteriormente, seguindo os estágios gerais de definição de fronteiras, interpolação da solução nas interfaces entre grelhas com diferentes graus de refinamento e avanço temporal da integração em todas as grelhas.

A estimativa do erro é realizada através de extrapolação de *Richardson* e por um método directo de estimativa do erro de truncatura local, baseado nos erros associados aos termos convectivos, num sistema de coordenadas curvilíneo, definido por:

$$T_E = T_{E1} + T_{E2} + T_{E3} \tag{3.63}$$

onde T_{E1}, T_{E2} e T_{E3} são medidas de erro dependentes de parâmetros do modelo, do Jacobiano e da transformação de variáveis.

A integração temporal é efectuada através de um algoritmo SIMPLE, com aplicação de esquemas *upwind* e centrais de segunda ordem para a discretização dos termos convectivos e difusivos, respectivamente.

Ewing et al **[42,43]** procedem ao estudo e desenvolvimento de esquemas de diferenças finitas aplicados na implementação de métodos de refinamento local no tempo e no espaço semelhantes aos referidos anteriormente. Estes métodos são utilizados na resolução de modelos parabólicos. Os esquemas deduzidos baseiam-se em fórmulas *backward* de *Euler* implícitas, tendo os autores estudado a sua estabilidade, assim com as respectivas análises do erro.

Smooke e Koszykowski **[105]** desenvolveram um algoritmo adaptativo de diferenças finitas aplicado a problemas instáveis de combustão caracterizados por frentes de chama. Estes problemas são modelizados através de sistemas de P.D.E.´s a uma dimensão espacial esquematizados por:

$$\underline{u}_t = f\left(x, t, \underline{u}, \underline{u}_x, \underline{u}_{xx}\right), \qquad a \le x \le b \tag{3.64}$$

com as condições inicial e fronteiras:

$$\begin{aligned} &g_1\left(a, t, \underline{u}(a), \underline{u}_x(a)\right) = 0 \\ &g_2\left(b, t, \underline{u}(b), \underline{u}_x(b)\right) = 0 \\ &\underline{u}(x,0) = \underline{r}(x) \qquad a \le x \le b \end{aligned} \tag{3.65}$$

A equação (3.64) é discretizada no tempo (t) e no espaço (x) aplicando diferenças finitas centradas e atrasadas ao sistema de equações não-lineares resolvido, em cada nível de tempo, por um método iterativo de *Newton* modificado. Por outro lado, o algoritmo adaptativo baseia-se na equidistribuição aproximada de uma função peso positiva em toda a malha, para cada nível temporal. A função peso escolhida representa a diferença entre a solução e os gradientes dos componentes da solução numérica v_i num determinado nível, no qual a malha gerada deve satisfazer as desigualdades seguintes:

$$\int_{x_j}^{x_{j+1}} \left| \frac{d\, v_i}{d\, x} \right| d\, x \le \delta \cdot \left| \max_{(a,b)} v_i - \min_{(a,b)} v_i \right| \qquad j = 0, \cdots, M-1$$
$$\int_{x_j}^{x_{j+1}} \left| \frac{d^2\, v_i}{d\, x^2} \right| d\, x \le \gamma \cdot \left| \max_{(a,b)} \frac{d\, v_i}{d\, x} - \min_{(a,b)} \frac{d\, v_i}{d\, x} \right| \qquad i = 1, \cdots, N \tag{3.66}$$

onde γ e δ - parâmetros inferiores a um.

Os valores de v_i e dv_i/dx, etc, são estimados a partir da solução numérica pelo uso de aproximações de diferenças finitas. De modo a promover a regularização da malha, ou seja, não permitir gradientes elevados entre comprimentos de malha adjacentes que podem provocar problemas de exactidão e de convergência do método de *Newton*, impõe-se, igualmente, a verificação da condição:

$$1/A \le \frac{h_j}{h_{j-1}} \le A \qquad j = 2,\ 3,\ \cdots, M \tag{3.67}$$

em que: $h_i = x_{i+1} - x_i$;
A - constante superior ou igual a um.

O algoritmo de refinamento estático pode ser resumido da forma seguinte:

❶ ⇨ Resolução da forma discreta da equação (3.64) na malha definida para o tempo t^n de modo a calcular a aproximação numérica v^n;
❷ ⇨ Testar as condições (3.66) para os n componentes de v;
❸ ⇨ Caso ❷ não seja verificado num intervalo da malha, inserir um novo nodo no ponto médio desse intervalo;
❹ ⇨ Testar a condição (3.67) e inserir um nodo no ponto médio dos intervalos que não a verifiquem;
❺ ⇨ Efectuar a interpolação da solução numérica para a nova malha e regressar ao passo ❶.

O processo é iterativo, sendo repetido até que todos os pontos da grelha verifiquem as condições (3.66) e (3.67). Nesse caso, o procedimento é avançado para o cálculo da solução no nível de tempo seguinte.

O método de refinamento estático da malha apresentado anteriormente é adequado para o caso de problemas onde a actividade espacial da solução varia consideravelmente com o tempo. No entanto, no caso de frentes estáveis, o problema pode ser resolvido através de uma malha com um número fixo de pontos que acompanhem a frente. Acoplam-se desse modo as equações de movimentação dos nodos ao esquema iterativo anterior, excluindo-se a introdução de novos nodos. No algoritmo de redistribuição dinâmica proposto por *Smooke e Koszykowski* **[105]**, os pontos da malha movem-se com velocidades obtidas por extrapolação linear de níveis de tempo anteriores. A localização dos M pontos da malha para o tempo t^{n+1} é calculada por:

$$\frac{d\, x_j^n}{d\, t} = \frac{x_j^n - x_j^{n-1}}{\Delta\, t^n} \qquad j = 1,\ 2,\ \cdots, M-1 \tag{3.68}$$

Integrando a equação (3.68) obtém-se as posições dos pontos da malha para o tempo t^{n+1} através de:

$$x_j^{n+1} = \frac{x_j^n - x_j^{n-1}}{\Delta t^n} \cdot \Delta t^{n+1} + x_j^n \tag{3.69}$$

A solução numérica no tempo t^n é interpolada para a nova malha no tempo t^{n+1} e utilizada como *input* no método iterativo de *Newton* para avaliação da solução para t^{n+1}. Os problemas resultantes da possibilidade de cruzamento dos nodos ou da sua saída do domínio são evitados através de simples reordenamento e colocação do nodo no ponto médio do intervalo entre o ponto fronteira e o nodo mais próximo, respectivamente.

Altas e Stephenson [5] apresentam um método automático adaptativo de geração de grelhas para problemas bidimensionais. O procedimento de geração baseia-se na utilização de erros de quadratura para estimar E, a variação da função da solução u(x,y) e pode ser resumido da forma seguinte: inicialmente o domínio é dividido em células, obtendo-se uma grelha uniforme; de seguida, calcula-se o valor de E para cada célula quadrada do domínio; cada célula que apresente valores de E superiores a uma determinada tolerância ε é dividida por bissecção dos seus lados, obtendo-se quatro subcélulas.

O cálculo de E para a célula S definida pelos vértices (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j+1}) e (x_{i+1}, y_{j+1}) é efectuado pela utilização da regra de quadratura trapezoidal. Esta, T_1, aplicada a u(x,y) na célula S leva à obtenção do termo de erro:

$$\iint_S u(x,y)\,ds - T_1 = E_{T1} \tag{3.70}$$

com,

$$T_1 = (x_{i+1} - x_i) \cdot (y_{j+1} - y_j) \cdot \left[u(x_i, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) + u(x_{i+1}, y_{j+1})\right] / 4 \tag{3.71}$$

Por outro lado, se a célula S tivesse sido dividida em quatro subcélulas, a soma das aplicações da regra trapezoidal a cada subcélula deverá dar:

$$\iint_S u(x,y)\,ds - T_2 = E_{T2} \tag{3.72}$$

com,

$$T_2 = (x_{i+1} - x_i) \cdot (y_{j+1} - y_j) \cdot \left[u(x_i, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) + u(x_{i+1}, y_{j+1}) + 2 \cdot \left(u(x_{i+1/2}, y_j) + u(x_{i+1}, y_{j+1/2}) + u(x_{i+1/2}, y_{j+1}) + u(x_i, y_{j+1/2})\right) + 4 \cdot u(x_{i+1/2}, y_{j+1/2})\right] / 16 \tag{3.73}$$

Por subtracção da equação (3.71) por (3.73), obtém-se:

$$E = |T_1 - T_2| = (x_{i+1} - x_i)\cdot(y_{j+1} - y_j)\cdot[3\cdot(u(x_i, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) + u(x_{i+1}, y_{j+1}))$$
$$- 2\cdot(u(x_{i+1/2}, y_j) + u(x_{i+1}, y_{j+1/2}) + u(x_{i+1/2}, y_{j+1}) + u(x_i, y_{j+1/2})) - 4\cdot u(x_{i+1/2}, y_{j+1/2})]/16 \quad (3.74)$$

A avaliação de E necessita dos valores da solução u no centro e nos pontos médios das arestas das células. Estes valores são calculados por interpolação das soluções conhecidas nos vértices de cada célula.

Portanto, o algoritmo proposto pode ser dividido nos passos seguintes:

❶ ⇨ Início do esquema, utilizando as subregiões obtidas numa grelha uniforme;
❷ ⇨ Avaliar E em cada subregião através da equação (3.74);
❸ ⇨ Subdividir em quatro subregiões, as regiões onde a quantidade E, calculada em ❷, seja superior à tolerância ε;
❹ ⇨ Obter, na nova malha, a aproximação da solução ou utilizar os valores interpolados de soluções anteriores;
❺ ⇨ Continuar os passos ❷ a ❹ até que o valor de E seja inferior à tolerância em todas as regiões do domínio.
❻ ⇨ Resolver o problema na grelha final.

O procedimento de geração da grelha é combinado com esquemas de diferenças finitas em malhas não uniformes, desenvolvidos pelos autores. Estes esquemas são aplicados para a discretização e posterior integração do problema.

Fung et al [48] propõem um procedimento adaptativo, aplicável a problemas multidimensionais evolutivos e não-evolutivos, para a melhoria da precisão da solução numérica duma grelha fixa base, através da introdução das aproximações do erro de truncatura, nos subdomínios de refinamento. As regiões de refinamento são identificadas através da estimativa do erro de truncatura local, em cada nodo.

Partindo de uma equação diferencial geral,

$$L\varphi = 0 \quad (3.75)$$

o operador L da função φ está relacionado com um operador de diferenças L_h (em que h corresponde ao espaçamento da grelha) de φ e o respectivo erro de truncatura TE, através da relação:

$$L\varphi = L_h\cdot\varphi_h + TE(\varphi, h) \quad (3.76)$$

Assim, a equação discreta a resolver, correspondente à aplicação em (3.76) de uma técnica de diferenças finitas é:

$$L_h\cdot\varphi_h + TE(\tilde{\varphi}, h) = 0 \quad (3.77)$$

A função argumento $\tilde{\varphi}$ de TE pode ser diferente da solução φ_h. Apenas no caso da solução exacta estar disponível, esta satisfaria (3.77) exactamente, com $\varphi = \varphi_h = \tilde{\varphi}$. Tal implica que o valor exacto de TE pode ser calculado pela aplicação do operador L_h à solução:

$$TE(\varphi, h) = -L_h \cdot \varphi \tag{3.78}$$

e a resolução do sistema,

$$L_h \cdot \varphi_h = -TE(\varphi, h) = L_h \cdot \varphi \tag{3.79}$$

proporciona a solução exacta nos pontos nodais. Portanto, para se obter uma melhoria da precisão da solução numérica, não é necessário alterar a estrutura da grelha base mas, apenas, considerar valores melhorados de TE, em cada nodo.

De uma forma analítica, TE é definido por todos os termos de derivadas de ordem elevada, truncados nas expansões de *Taylor*, a quando da dedução das fórmulas de discretização. Portanto, se forem conhecidos mais valores adjacentes para uma função, maiores ordens de derivadas podem ser computadas. Essa sequência pode ser executada da forma:

$$TE_{h/N} = TE(\varphi_{h/N}, h) = -L_h \cdot \varphi_{h/N} \tag{3.80}$$

onde h/N, se relaciona com valores baseados numa grelha de tamanho h/N.

Assim, o processo de refinamento pode ser resumido da forma seguinte: inicialmente, considera-se uma grelha base, não necessariamente uniforme, de tamanho local h, para a solução φ_h. O valor de TE é fixado em zero (TE=0) e a equação (3.76) é resolvida para φ_h, com uma precisão pré-definida ε. Considera-se φ_h como uma solução refinada de uma grelha 2h, formada pela omissão dos nodos intermédios da grelha base, há excepção dos nodos fronteira e dos imediatamente adjacentes a estes. Deste modo, o erro de truncatura é estimado, por utilização da fórmula $L_{2h}\varphi_h$ em cada ponto da malha h. As regiões onde o TE estimado ($-L_{2h}\varphi_h$) é superior ao valor de ε são identificadas. Nas regiões não seleccionadas, os valores de TE são anulados, sendo introduzidas zonas tampão refinadas onde, igualmente TE=0, de forma a garantir uma maior suavidade na transição da solução. Informações como: o tamanho dos subdomínios onde se procede à injecção dos TE; a evolução das condições fronteira, e diferentes parâmetros como o factor de refinamento N de cada subregião, são transmitidas ao integrador de grelha fina $L_{h/N}\varphi_{h/N}=0$, de forma a se obter a solução localmente refinada $\varphi_{h/N}$. Estas soluções são utilizadas no cálculo das aproximações de TE ($-L_h\varphi_{h/N}$) para a ciclo de refinamento seguinte, até que a relação seguinte seja verificada em todo o domínio:

$$\left| L_h \varphi_{h/N} - L_h \varphi_{h/M} \right| < \varepsilon \qquad \text{para qualquer } M > N \tag{3.81}$$

A maior vantagem da injecção do erro de truncatura, em relação a outros métodos de refinamento, consiste na dispensa de armazenamento das soluções correspondentes às malhas finas. Os valores aproximados de TE indicam a qualidade da solução local e se os refinamentos necessários podem ser obtidos com menor esforço.

Oliveira et al **[88]** desenvolvem dois algoritmos de refinamento, baseados no método das linhas, que aplicam na resolução numérica dum modelo correspondente ao balanço energético a um sistema de leito fixo. Ambos os algoritmos envolvem um procedimento de adaptação temporal, baseado em extrapolações de *Richardson*, associado a uma estratégia de adaptação espacial.

A adaptação espacial envolve o refinamento do domínio espacial, de forma à obtenção de diversos subdomínios, constituídos pela junção dos nodos que não satisfaçam o seguinte teste:

$$\exists i : \left| {}^{i}TE \right| > \varepsilon \qquad (3.82)$$

onde:

ε - tolerância estabelecida;

${}^{i}TE$ - componente i do vector TE, que pode apresentar diferentes significados, conforme são executadas as localizações de primeiro nível ou de níveis superiores:

⇨ ***Primeira Localização*** – estimativa do erro de truncatura espacial;
⇨ ***Localizações Seguintes*** – diferença entre duas aproximações sucessivas do erro de truncatura espacial.

Os refinamentos sucessivos consistem na construção de subproblemas, definidos a partir do problema global, associados a subdomínios com nível de refinamento crescente, de espaçamento de malha h/N, N=2, 2^2, 2^3, … .

A diferença principal entre cada algoritmo proposto consiste na forma como é construída a solução actualizada, correspondente ao tempo final de cada passo, em cada nível de refinamento. Esta é utilizada para a definição dos subdomínios de refinamento seguintes. Assim, tem-se que:

Algoritmo R.S.A. (*Refined Solution Algorithm*) ➔ a solução actualizada entre cada ciclo de refinamento n é a solução de nível imediatamente anterior n-1, onde os componentes, respeitantes aos subdomínios onde o refinamento foi executado, são substituídos pela solução refinada, calculada com $h/2^n$.

Algoritmo S.T.I.A. (*Spatial Truncation Error Injection*) ➔ as soluções localmente refinadas são utilizadas para o cálculo de estimativas de erro de truncatura espacial, cujos valores são injectados na solução previamente calculada. É óbvio que este procedimento apenas é executado nas regiões onde o refinamento se revelou necessário.

O procedimento de refinamento sucessivo é aplicado até que os valores de TE verifiquem a tolerância estabelecida ε, na totalidade do domínio.

O algoritmo proposto por *Nowak et al* **[87]** destina-se à resolução de sistemas de equações diferenciais parciais parabólicas unidimensionais, através de um controlo adaptativo espacial e temporal.

A adaptatividade é baseada na estimativa detalhada dos erros associados às discretizações espaciais e temporais. Deste modo, os autores consideram que o erro total, introduzido pela discretização do domínio global, pode ser definido por três contribuições distintas, da forma seguinte:

$$\in = \in_t + \in_{MIX} + \in_x \qquad (3.83)$$

em que:

$\in$ - erro total;

$\in_t$ - erro introduzido pela discretização temporal;

$\in_x$ - erro associado à discretização espacial;

$\in_{MIX}$ - erro misto que depende simultaneamente das discretizações espacial e temporal.

Como cada uma das diferentes partes do erro se podem cancelar entre si, o controlo do erro total ($\in$) não é aconselhável. Assim, os autores procedem à avaliação separada de cada um dos componentes ($\in_t$, $\in_x$ e $\in_{MIX}$), de maneira a desenvolverem um algoritmo robusto. Portanto, são deduzidas fórmulas para a estimativa apropriada de cada um dos erros referidos anteriormente ($\boldsymbol{\varepsilon}_t$, $\boldsymbol{\varepsilon}_x$ e $\boldsymbol{\varepsilon}_{MIX}$), baseadas na comparação da integração temporal, efectuada por um procedimento de extrapolação de *Richardson*, em duas malhas de tamanhos distintos (uma malha fina e outra esparsa).

A partir da aproximação dos diferentes erros considerados, estabelece-se um algoritmo de controlo global do domínio, em cada passo da integração, baseado em normas de raízes quadradas médias pesadas, definidas por:

$$\boldsymbol{\varepsilon}_t = \left\|\boldsymbol{\varepsilon}_t\right\| = \sqrt{\frac{2}{n \cdot m} \cdot \sum_{i=1}^{n/2} \sum_{j=1}^{m} \left(\frac{\boldsymbol{\varepsilon}_{t,i,j}}{w_{i,j}} \right)^2} \leq tol_t \tag{3.84}$$

$$\boldsymbol{\varepsilon}_x = \left\|\boldsymbol{\varepsilon}_x\right\| = \sqrt{\frac{2}{n \cdot m} \cdot \sum_{i=1}^{n/2} \sum_{j=1}^{m} \left(\frac{\boldsymbol{\varepsilon}_{x,i,j}}{w_{i,j}} \right)^2} \leq tol_x \tag{3.85}$$

onde:

m - número de equações diferenciais parabólicas;

n - número de nodos da malha;

tol_x - precisão requerida no espaço;

tol_t - precisão requerida no tempo;

$w_{i,j}$ - valores dos pesos escolhidos de modo a verificarem a condição:

$$w_{i,j} = \max\left(\left|u_{i,j}\right|, \left|u_j^t\right|\right) \tag{3.86}$$

em que: u_j^t - valor prescrito pelo utilizador.

Recomenda-se que o u_j^t escolhido seja da mesma ordem da magnitude máxima esperada para $u_{i,j}$. Deste modo,

$$u_j^t = \max\left(\left|u_j^{esperado}(x,t)\right|\right) \tag{3.87}$$

Em simultâneo com o controlo global, é testado se os erros máximos locais,

$$\boldsymbol{\varepsilon}_{x,max}^{loc} = \max_i \left(\sqrt{\frac{1}{m} \cdot \sum_{j=1}^{m} \left(\frac{\boldsymbol{\varepsilon}_{x,i,j}}{w_{i,j}} \right)^2} \right) \tag{3.88}$$

$$\boldsymbol{\varepsilon}_{t,max}^{loc} = \max_{i}\left(\sqrt{\frac{1}{m}\cdot\sum_{j=1}^{m}\left(\frac{\boldsymbol{\varepsilon}_{t,i,j}}{w_{i,j}}\right)^2}\right) \qquad (3.89)$$

ao longo da coordenada espacial, são inferiores a $\xi_x\ tol_x$ e $\xi_t\ tol_t$, respectivamente. ξ_x e ξ_t são factores heurísticos, que podem variar entre 10 e 100, assegurando que não existem erros locais arbitrariamente altos em regiões limitadas do domínio. No que diz respeito à definição das tolerâncias, é conveniente que estas verifiquem as relações seguintes: $tol_t \le tol_x$ e $0.001 \le tol_x \le 0.01$.

Após a execução completa de um passo temporal (verificação simultânea das equações (3.84) e (3.85) e $\boldsymbol{\varepsilon}_x^{loc} \le \xi_x\ tol_x$; $\boldsymbol{\varepsilon}_t^{loc} \le \xi_t\ tol_t$), a malha espacial é redefinida e um novo passo temporal ΔT é escolhido, para o passo de integração seguinte.

O procedimento de redefinição da malha é executado pela comparação entre a estimativas de erro local espacial $\boldsymbol{\varepsilon}_{x,i}^{loc}$, com uma tolerância $\overline{tol}$, fixada por:

$$\overline{tol} = tol_x \qquad (\text{se } \boldsymbol{\varepsilon}_{x,MAX}^{loc} \le tol_x) \qquad (3.90)$$

e

$$\overline{tol} = \sqrt{\boldsymbol{\varepsilon}_{x,MAX}^{loc}\cdot tol_x} \qquad (\text{se } \boldsymbol{\varepsilon}_{x,MAX}^{loc} > tol_x) \qquad (3.91)$$

Se o erro em qualquer nodo da malha fôr maior que $\overline{tol}$, os intervalos envolvidos são subdivididos uma vez. Caso contrário, se $\boldsymbol{\varepsilon}_{x,i}^{loc} < 1/6\ \overline{tol}$, este nodo é cancelado, tomando em atenção que nunca se cancelam simultaneamente dois nodos adjacentes. O procedimento de redefinição é completado por um passo de suavização da malha, através da fixação de uma relação máxima de dois, entre os tamanhos de intervalos consecutivos.

A fixação do passo temporal seguinte é realizada, tendo em conta que os valores óptimos não deverão ser muito diferentes dos utilizados no passo anterior, assumindo que a dinâmica do problema não varia muito entre estes dois passos. No entanto, é necessário ter em atenção que o procedimento de redefinição local da malha a modificou. Assim, é necessário avaliar a estimativas do erros associadas à nova malha, em relação às anteriormente obtidas com a malha não redefinida. Tendo em conta esta transformação, *Nowak et al* **[87]** desenvolvem um procedimento de estimativa dos valores óptimos do passo global ($\Delta\tilde{T}$) e dos passos temporais $\Delta\tilde{T}_k$, k = 1, …, q, (referentes aos q passos de extrapolação no tempo), para a nova malha redefinida.

Kulkarni e Dudukovic **[77]** apresentam um algoritmo implícito de diferenças finitas espacial e temporalmente adaptativas, que possibilita uma identificação eficaz de frentes mássicas e térmicas móveis, na simulação de reactores de leito fixo, onde se processam reacções fortemente exotérmicas.

O refinamento da malha e consequente colocação nodal, baseia-se na análise da magnitude das primeiras e segundas derivadas espaciais, através da verificação dos critérios seguintes:

Primeira Derivada: $$\frac{y_i^{n+1} - y_{i-1}^{n+1}}{y_i^{n+1}} \times 100 \le \delta_1 \qquad (3.92)$$

Segunda Derivada: $$\left.\frac{\delta^2 y}{\delta z^2}\right|_i^{n+1} \times \frac{\left[\left(z_{i+1}^{n+1} - z_i^{n+1}\right) \times \left(z_i^{n+1} - z_{i-1}^{n+1}\right)\right]^2}{4} \leq \delta_2 \quad (3.93)$$

em que: δ_1 e δ_2 - tolerâncias especificadas pelo utilizador.

Os autores verificam que o critério de segunda derivada conduz a resultados mais precisos e uma maior robustez do algoritmo. Este facto deve-se a que apenas nas regiões onde a segunda derivada é mais elevada, a solução tem tendência para se tornar instável, gerando oscilações nos perfis. Assim, este critério permite uma colocação de nodos suficientes nas extremidades das frentes e, deste modo, reduz a ocorrência de dispersão numérica. Ao contrário, as zonas onde a primeira derivada é muito elevada correspondem somente aos perfis praticamente verticais das frentes, onde apenas são necessários alguns nodos para a uma simulação apropriada.

Por outro lado, a adaptação temporal é sempre realizada por um controlo da primeira derivada temporal. Assim, o tamanho do passo é determinado pelo critério seguinte:

$$\frac{y_i^{n+1} - y_i^n}{y_i^n} \times 100 \leq \delta_t \quad (3.94)$$

e

$$t^{n+1} - t^n \leq t_{MAX} \quad (3.95)$$

com: δ_t - tolerância pré-definida;

t_{MAX} - tamanho máximo do passo temporal, fixado em função das características físicas do problema.

Kulkarni e Dudukovic **[77]** verificam, igualmente, a inadequação da aplicação de interpolações de ordem elevada nas regiões críticas do domínio, onde as segundas derivadas são muito elevadas. Este tipo de interpolações introduzem oscilações artificiais nos perfis, conduzindo a resultados pouco precisos. Assim, são utilizadas interpolações lineares, sempre que a malha é refinada.

Fraga e Morris **[47]** desenvolvem um novo método de refinamento espacial que aplicam a equações não-lineares unidimensionais dispersivas de onda. Este tipo de problemas caracterizam-se pela formação de perfis da solução que apresentam uma ou várias ondas solitárias móveis designadas por *solitons*. Estes fenómenos têm a propriedade de não mudarem de forma quando colidem entre si, não sendo, no entanto exclusivos deste tipo de equações, podendo ocorrer em muitos outros problemas da área da matemática e da física.

O método baseia-se na descrição geométrica da solução e difere significativamente dos anteriores, já que não recorre a qualquer estimativa de erro ou equidistribuição de uma quantidade particular. A estratégia pode ser resumida da forma seguinte: procede-se, inicialmente, à localização de cada *soliton* da solução; de seguida, introduz-se uma malha fina que constitua o suporte de cada *soliton*, preenchendo-se os intervalos entre malhas finas, com pontos mais espaçados. Cada um dos subintervalos é discretizado uniformemente.

A localização da posição dos *solitons* é efectuada através da definição de um valor máximo S_{MIN} para a solução. Assim, qualquer solução acima deste valor é considerada como fazendo parte de um *soliton*. Este procedimento localiza os *solitons*, mas não define completamente o seu domínio. Assim, os intervalos:

$$I_j = \left[a_j, b_j\right] \qquad j = 1, \cdots, n_S \tag{3.96}$$

em que a_j - começo do *soliton*;
b_j - fim do *soliton*;
n_S - número de *solitons*.

são estendidos para cada um dos lados, por uma quantidade proporcional ao tamanho de cada intervalo. Assim, os novos intervalos são definidos por:

$$I'_j \leftarrow \left[a_j - \beta \cdot \left|I_j\right|, b_j + \beta \cdot \left|I_j\right|\right] \qquad j = 1, \cdots, n_S \tag{3.97}$$

onde β - valor calculado através das condições iniciais.

No entanto, os intervalos I'_j têm de ser ainda mais estendidos, já que, devido aos aspectos evolutivos do problema, os *solitons* podem movimentar-se entre cada passo temporal. Assim,

$$I'_j \leftarrow \left[a_j - \text{buffer} \cdot \beta \cdot \left|I_j\right|, b_j + \text{buffer} \cdot \beta \cdot \left|I_j\right|\right] \qquad j = 1, \cdots, n_S \tag{3.98}$$

em que: buffer - constante dependente do problema que varia entre um e três.

Com as sucessivas extensões dos intervalos, é possível a ocorrência de sobreposição de intervalos. Assim, após cada extensão, cada grupo de intervalos sobrepostos é combinado num intervalo maior, antes do procedimento avançar para o próximo passo.

Então, a nova malha é definida por um conjunto de submalhas contíguas que preenchem o domínio. Cada intervalo, definido anteriormente, corresponde a uma submalha uniforme, com uma discretização espacial de comprimento h_{goal}. Os espaços entre cada intervalo são também discretizados uniformemente, introduzindo-se N_{GAP} pontos em cada um deles. Assim, o número total de pontos é calculado por:

$$N = N_{GAP} + \sum_{j=1}^{n_S} \left[\frac{\left|I'_j\right|}{h_{goal}} + N_{GAP} \right] + 1 \tag{3.99}$$

A principal vantagem deste método é a sua simplicidade e o facto de apenas serem adicionados pontos nas zonas onde estes são verdadeiramente necessárias. Como apenas se processa um refinamento espacial, qualquer esquema de integração temporal pode ser aplicado. De qualquer modo, é possível conjugar qualquer método de adaptação temporal com este método.

3.3.2.3 - Métodos Adaptativos de Redistribuição Nodal Dinâmica

Kansa et al **[73]** desenvolveram uma estratégia de movimentação dos nodos de modo que as P.D.E.´s, quando transformadas para o domínio numérico, dependam minimamente das derivadas espaciais. De um modo geral, o sistema transformado é escrito da forma:

$$\left.\frac{\delta A_k}{\delta t}\right|_{\xi} = G_k - \frac{\delta F_k}{\delta X} + \frac{dX}{dt}\cdot\frac{\delta A_k}{\delta X} \tag{3.100}$$

em que:

A_k - componente k da solução;
G_k - função dependente da solução e da coordenada espacial X;
F_k - função de X, da solução e do gradiente da solução.

Por outro lado, é definida uma velocidade (V_k) para cada componente da solução pela expressão:

$$V_k = \frac{\delta F_k / \delta X}{\delta A_k / \delta X} \tag{3.101}$$

Se os nodos se movimentarem da forma $dX/dt = V_k$, então a P.D.E. correspondente ao componente k reduz-se a:

$$\left.\frac{\delta A_k}{\delta t}\right|_{\xi} = G_k \tag{3.102}$$

Num problema múltiplo, geralmente, a velocidade V_k não é a mesma para todos os componentes, devido a imposições numéricas. Assim, as velocidades são definidas de modo a minimizar a equação,

$$E = \sum_{k=1}^{K}\left[\frac{dX}{dt} - V_k\right]^2 \tag{3.103}$$

que é aproximada através de polinómios de colocação a três pontos para o cálculo dos gradientes. Em cada passo, alterna-se a resolução da equação (3.102), através de um método de diferenças finitas, incorporando termos de viscosidade, com a computação das velocidades nodais. De forma a se fixar uma separação mínima entre os nodos, de modo a impedir o seu cruzamento, estes são redefinidos por intermédio de interpolação após cada passo temporal. Também se aplicam interpolações de modo a impedir uma concentração excessiva de nodos junto a choques.

Ghia et al **[51]** apresentam um método bastante semelhante ao proposto por *Kansa et al* **[73]**, no qual os nodos são movimentados de forma a que a P.D.E., no domínio numérico, tenha uma dependência mínima das derivadas espaciais de 1ª. ordem. No domínio espacial, o modelo unidimensional de problemas monocomponentes pode ser representado por:

$$\left.\frac{\delta A}{\delta t}\right|_{X} = \mu\cdot\frac{\delta^2 A}{\delta X^2} - F(X,A)\cdot\frac{\delta A}{\delta X} - G(X,A)\cdot A - g[X] \tag{3.104}$$

onde μ - constante de valor muito reduzido.

Quando transformada para as coordenadas computacionais, a P.D.E. toma a forma:

$$\left[\left.\frac{\delta A}{\delta t}\right|_{\xi} + G\cdot A + g\right]\cdot\left[\frac{\delta X}{\delta \xi}\right]^2 - \mu\cdot\frac{\delta^2 A}{\delta \xi^2} = R\cdot\frac{\delta A}{\delta \xi}\cdot\left[\frac{\delta X}{\delta \xi}\right]^{-1} \tag{3.105}$$

com:

$$R = \left[\left.\frac{\delta X}{\delta t}\right|_{\xi} - F\right]\cdot\left[\frac{\delta X}{\delta \xi}\right]^2 - \mu\cdot\frac{\delta^2 X}{\delta \xi^2} \tag{3.106}$$

Na equação (3.105) as derivadas espaciais de 1ª. ordem são isoladas no termo da direita. A estratégia adaptativa baseia-se na movimentação dos nodos de modo que se verifique a relação:

$$\left[\left.\frac{\delta X}{\delta t}\right|_{\xi} - F\right]\cdot\left[\frac{\delta X}{\delta \xi}\right]^2 - \mu\cdot\frac{\delta^2 X}{\delta \xi^2} = 0 \tag{3.107}$$

A equação (3.107) é discretizada através de diferenças *upwind* para $\delta X/\delta\xi$ e resolvida para a velocidade de cada nodo por aplicação de uma técnica implícita. As velocidades nodais são utilizadas para o cálculo de R (ou seja, do resíduo da equação (3.106)) que é introduzido na equação (3.105). São usadas diferenças finitas centrais para a discretização das equações (3.105) e (3.106) enquanto que o cálculo das derivadas temporais da solução é efectuado por intermédio da técnica implícita referida. O uso de diferenças *upwind* na primeira parte do algoritmo confere estabilidade à solução. Os autores apresentam também uma generalização do método para aplicação na resolução de problemas a duas dimensões.

Anyiwo **[9]** utiliza um método adaptativo que envolve uma transformação de variáveis em dois estágios. Inicialmente, a P.D.E. é transformada das coordenadas físicas (X_i,t) para coordenadas ortogonais curvilíneas (S_i,t), passando destas, posteriormente, para o espaço computacional (ξ_i,t) de nodos equidistribuidos. A medida de erro é definida por,

$$E = e^{(\gamma)} = e^{\left(\sum_{i=1}^{I}\gamma_i\right)} \tag{3.108}$$

ou seja, E é um produto dos factores $E_i = e^{(\gamma_i)}$, sendo γ_I uma forma de quantificar a deformação da malha em cada direcção coordenada, que é calculada pela expressão:

$$\gamma_i = \lambda^1\cdot\alpha_i^1\cdot\Omega_i + \lambda^2\cdot\left|\alpha_i^2 + \beta_i^2\right|\cdot\Omega_i + \lambda^3\cdot\sigma^3\cdot\Omega_i \tag{3.109}$$

onde: λ^1, λ^2, λ^3 - constantes de peso;
α_i^2, β_i^2 - curvatura circular e torsão de curvas de S_i constante nas coordenadas físicas;

$$\alpha_i^1 = 1/N_i \qquad \text{e} \qquad \Omega_i = 1/\ln(N_i - 1) \tag{3.110}$$

com: N_i - número de pontos na direcção i nas coordenadas (ξ_i,t).
e ainda,

$$\sigma^3 = \left|\delta_i^3\right| + \left|\alpha_i^3 + \beta_i^3\right| \tag{3.111}$$

em que:
δ_i^3, α_i^3 e β_i^3 - declive, curvatura circular e torsão da solução A nas coordenadas (S_i,t).

A transformação das coordenadas (S_i,t) para (ξ_i,t) é realizada fazendo com que ao longo de cada direcção S_i se verifique a relação:

$$\frac{\delta S_i}{\delta \xi_i} = \frac{\int_C E_i(s_i)\, ds_i}{N_i \cdot E_i(S_i)} \tag{3.112}$$

A equação (3.112) representa uma equidistribuição de E entre todos os nodos, sendo $dS_i/d\xi_i$ proporcional entre curvas de S_i constante no espaço físico.

A transformação entre as coordenadas (X_i,t) e (S_i,t) é obtida através de:

$$dS_i = \cos(\theta_{ij}) \cdot dX_i \tag{3.113}$$

onde: θ_{ij} - ângulo entre as direcções coordenadas S_i e X_j restringido pela relação,

$$\cos(\theta_{jk}) \cdot \cos(\theta_{ik}) = 0 \qquad \text{se} \qquad i \neq j \tag{3.114}$$

de modo que as coordenadas (S_i,t) sejam ortogonais.

O cálculo de θ_{ij} no interior do domínio físico, é efectuado por uma fórmula de interpolação pesada da solução. Ao longo das direcções ξ_I, calcula-se θ_{ij} em função da variação de $dS_i/d\xi_I$, promovendo uma fusão suave com o comportamento angular das fronteiras do domínio físico.

O algoritmo proposto pode ser resumido da forma seguinte: usa-se a medida de E para calcular as derivadas $dS_i/d\xi_i$ da primeira transformação em cada nodo. Estes valores são usados para avaliar os novos valores de θ_{ij} e portanto, efectuar a segunda transformação de variáveis. As P.D.E.´s são transformadas para o domínio de (ξ_i,t) e resolvidas por um esquema de diferenças finitas para o próximo passo temporal. Finalmente, usam-se as novas soluções e transformações para calcular a nova medida do erro.

Introduz-se um método adicional para controlo da adaptação de forma a restringir o movimento nodal e manter uma transformação suave, por inserção de um valor não-nulo de dX_i/dt nas P.D.E.´s transformadas. Este valor é definido de modo a satisfazer a equação de transformação no espaço (ξ_i,t),

$$\left.\frac{\delta X_i}{\delta t}\right|_{\xi} = \nabla\left[-c \cdot U \cdot X_i + \mu \cdot \nabla X_i\right] \tag{3.115}$$

em que: c - constante positiva superior ou igual a um;
μ - coeficiente de viscosidade;
U - velocidade advectiva para a solução física.

Yanenko e colaboradores [123-125] desenvolveram um método adaptativo baseado na transformação de variáveis para problemas bidimensionais, através da aplicação duma medida de erro calculada pela soma de várias contribuições,

$$E = \lambda_C \cdot E_C + \lambda_a \cdot E_a + \lambda_V \cdot E_V \tag{3.116}$$

onde: E_C - medida do desvio da conformidade da transformação:

$$E_a = \left| \frac{\delta \xi_1}{\delta X_1} - \frac{\delta \xi_2}{\delta X_2} \right|^2 + J^R \cdot \left| \frac{\delta \xi_1}{\delta X_2} - \frac{\delta \xi_2}{\delta X_1} \right|^2 \tag{3.117}$$

E_a - medida do grau de movimentação dos nodos com o meio, ou seja, a "*Lagrangidade*" da transformação:

$$E_a = \left| U_1 - \frac{\delta X_1}{\delta t} \right|^2 + \left| U_2 - \frac{\delta X_2}{\delta t} \right|^2 \tag{3.118}$$

sendo (U_1,U_2) a velocidade do meio, ao passo que $(\delta X_1/\delta t , \delta X_2/\delta t)$ corresponde à velocidade da grelha no espaço físico.

E_V - medida da variação da solução:

$$E_V = W \cdot J^Q \tag{3.119}$$

em que: W - função pesada dos gradientes dos componentes da solução A.

Através da equidistribuição de E_V, o Jacobiano da transformação tenderá a reduzir-se em zonas de W elevado. Deste modo, a grelha será refinada em regiões de medida de erro elevadas.

Efectuando a minimização do integral de E ao longo de X_1, X_2 e t obtém-se a P.D.E. dependente do espaço e do tempo que representa a transformação. Esta equação é resolvida simultaneamente com a P.D.E. da solução A, por aplicação de um método de diferenças finitas para a integração temporal.

Inspirados pelo método automático de geração de grelhas de *Winslow* [122], *Hindman e Spencer*[63] desenvolveram um método adaptativo unidimensional, a partir de uma transformação de variáveis, que verifique a equação de *Poisson* unidimensional:

$$\frac{\delta^2 P}{\delta X^2} = P \tag{3.120}$$

Na equação (3.120), P = P(ξ) é uma função definida de modo a que apresente valores elevados nas zonas onde se pretende concentrar os nodos. O método de estimativa das velocidades nodais requer a resolução da equação:

$$\frac{\delta^2 X}{\delta \xi^2} + \left[\frac{\delta X}{\delta \xi}\right]^3 \cdot P = 0 \qquad (3.121)$$

sujeita às condições fronteira: $X(\xi^0) = X^0$ e $X(\xi^N) = X^N$. Esta expressão é obtida através do cálculo de segunda derivada, em relação a ξ, de:

$$\xi(X,t) = c^{te} \cdot \int_{X^0}^{X} E\, dt \qquad (3.122)$$

em que o integrando é uma medida de erro, definida com base no trabalho de *Dwyer et al* **[33]**, da forma:

$$E = 1 + \lambda \cdot W \qquad (3.123)$$

onde $\sqrt{W} = \delta A / \delta \xi$ corresponde ao declive da solução de um problema unidimensional teste no domínio numérico. A comparação entre a equação (3.121) e a segunda derivada de (3.122) leva à relação:

$$P = \frac{1}{E} \cdot \frac{\delta E}{\delta \xi} \cdot \left[\frac{\delta X}{\delta \xi}\right]^{-2} = \frac{\lambda}{1 + \lambda \cdot W} \cdot \frac{\delta W}{\delta \xi} \cdot \left[\frac{\delta X}{\delta \xi}\right]^{-2} \qquad (3.124)$$

Hindman e Spencer **[63]** deduzem igualmente uma forma alternativa para a função P, através da minimização do integral:

$$\int_{X^0}^{X^N} \left[\left[\frac{\delta \xi}{\delta X}\right]^2 + \lambda \cdot W \cdot \frac{\delta \xi}{\delta X}\right] dx$$

Este procedimento é, por sua vez, baseado no trabalho de *Brackbill* **[17]** sendo o primeiro termo relacionado com a suavidade da transformação, enquanto o segundo termo promove a concentração de nodos em regiões de elevado W. Através de cálculo variacional, obtém-se a P.D.E. da transformação,

$$-2 \cdot \left[\frac{\delta X}{\delta \xi}\right]^{-1} + \lambda \cdot W \cdot \left[\frac{\delta X}{\delta \xi}\right]^2 = c^{te} \qquad (3.125)$$

que é diferenciada em ordem a ξ, obtendo-se a função:

$$P = \frac{\lambda \cdot \frac{\delta W}{\delta \xi} \cdot \frac{\delta X}{\delta \xi}}{\left(2 + 2 \cdot \lambda \cdot W \cdot \left[\frac{\delta X}{\delta \xi} \right]^3 \right)} \tag{3.126}$$

Por aplicação do método adaptativo fronteira de *Hindman et al* **[62]** procede-se à diferenciação em ordem ao tempo da equação (3.121), evitando assim, a sua resolução directa. Obtém-se:

$$\frac{\delta^2 \dot{X}}{\delta \xi^2} + 3 \cdot \frac{\delta \dot{X}}{\delta \xi} \cdot \left[\frac{\delta X}{\delta \xi} \right]^2 \cdot P + \left[\frac{\delta X}{\delta \xi} \right]^3 \cdot \dot{P} = 0 \tag{3.127}$$

onde $\dot{X}$ - velocidade nodal.

Esta equação é aproximada através de diferenças centrais em cada nodo. A expressão de diferenças centrais para $\dot{P}$ é expandida em termos de $\dot{A}$ e $\dot{X}$, introduzindo-se a P.D.E. do problema para eliminar os valores de $\dot{A}$. As equações resultantes são resolvidas em cada nodo para o cálculo das velocidades.

Após a transformação da P.D.E. original para o sistema coordenado computacional, aplica-se um algoritmo de passo temporal duma forma alternada entre a P.D.E. transformada e as equações matriciais das velocidades nodais. Utilizam-se para tal, métodos implícitos de primeira e segunda ordem ou o método explícito predictor-corrector de segunda ordem de *MacCormack* **[80]**. O posicionamento inicial dos nodos também deverá satisfazer a equação (3.121).

Hindman e Spencer **[63]** discutem igualmente a aplicação teórica do seu método a problemas bidimensionais.

Matsuno e Dwyer **[82]** desenvolvem um método geral, eficiente e preciso, para aplicação de técnicas de geração de grelhas elípticas. Para tal, estudam uma perspectiva diferente de utilização da equação de *Poisson* (equação (3.120)) para uma dimensão, com a qual pretendem deduzir novas formas para a função P(ξ) que define as propriedades da adaptação em função da solução. A aplicação de equações de *Poisson* resulta na relação seguinte para a transformação de variáveis:

$$x_{\xi\xi} = -\left(P \Big/ J^2 \right) \cdot x_\xi \tag{3.128}$$

em que $J = \xi_x$ é o Jacobiano da transformação.

Por outro lado, define-se uma função peso w de modo a que apresente valores elevados em que as variações de x são reduzidas. Desse modo, combinando a equidistribuição de w com a equação (3.128) obtém-se a relação:

$$x_{\xi\xi} = -\left(w_\xi \Big/ w \right) \cdot x_\xi \tag{3.129}$$

onde $P = -J^2 \cdot \left(w_\xi / w \right)$

A resolução da equação (3.129) impõe o estabelecimento de restrições a P que dependem da escolha de w. Assim, através da discretização por diferenças centrais de (3.129), verifica-se que a razão entre comprimentos de malha adjacentes r_i deverá ser positiva para qualquer nodo i, o que implica:

$$\left| \left(P / J^2 \right)_i \right| = \left| \left(w_\xi / w \right)_i \right| < 2 \tag{3.130}$$

conjugada com a introdução de restrições para r_i da forma:

$$1/K \le r_i \le K \tag{3.131}$$

A escolha da função de peso w adequada é bastante importante, revelando-se muito útil a aplicação de uma função dependente da primeira derivada da solução. Assim, os autores consideram a função:

$$w = \left[1 + b_1 \cdot \left(f_x \right)^2 \right]^{1/2} \tag{3.132}$$

onde a constante de peso b_1 pode, à partida, ser limitada superiormente através da introdução das restrições (3.130) e (3.131) já referidas.

No caso de utilização de algoritmos de integração robustos pode, igualmente, ser introduzida uma adaptação de segunda derivada da forma:

$$w = w_1 \cdot w_2 = \left[1 + b_1 \cdot \left(f_x \right)^2 \right]^{1/2} \cdot \left[1 + b_2 \cdot \left(f_{xx} \right)^2 \right]^{1/2} \tag{3.133}$$

que, por definição de restrições equivalentes para os termos w_1 e w_2, permite a estimativa dos valores máximos admissíveis para as constantes de peso b_1 e b_2.

Apesar dos autores apresentarem a dedução do método para casos unidimensionais, o objectivo principal consiste na extensão do procedimento a problemas tridimensionais, para os quais a equação de *Poisson* correspondente é definida por:

$$\begin{aligned} \xi_{xx} + \xi_{yy} + \xi_{zz} &= P(\xi, \eta, \zeta) \\ \eta_{xx} + \eta_{yy} + \eta_{zz} &= Q(\xi, \eta, \zeta) \\ \zeta_{xx} + \zeta_{yy} + \zeta_{zz} &= R(\xi, \eta, \zeta) \end{aligned} \tag{3.134}$$

onde $P = w_\xi \Big/ \left[\left(s_\xi \right)^2 \cdot w \right]$, $Q = w'_\eta \Big/ \left[\left(s'_\eta \right)^2 \cdot w' \right]$ e $R = w''_\zeta \Big/ \left[\left(s''_\zeta \right)^2 \cdot w'' \right]$.

A estratégia de desenvolvimento do algoritmo é idêntica à do caso anterior, baseando-se no método de *Winslow* **[122]**. As funções w, w´ e w´´, que contêm informação sobre os critérios de adaptação, são escolhidas de maneira semelhante ao longo das três direcções de

comprimento de arco s, s´ e s´´, associadas às coordenadas computacionais ξ, η e ζ, respectivamente:

$$\begin{aligned} w &= \left[1 + b_1 \cdot \left(F_s\right)^2\right]^{1/2} \cdot \left[1 + b_2 \cdot \left(F_{ss}\right)^2\right]^{1/2} \\ w' &= \left[1 + b_1' \cdot \left(F_{s'}\right)^2\right]^{1/2} \cdot \left[1 + b_2' \cdot \left(F_{s's'}\right)^2\right]^{1/2} \\ w'' &= \left[1 + b_1'' \cdot \left(F_{s''}\right)^2\right]^{1/2} \cdot \left[1 + b_2'' \cdot \left(F_{s''s''}\right)^2\right]^{1/2} \end{aligned} \qquad (3.135)$$

No entanto, para casos multidimensionais, é necessário considerar igualmente a questão da ortogonalidade da grelha, que impõe uma restrição significativa à adaptação. Para tal, introduz-se uma função de peso P_G, que irá competir com os termos de adaptação. As constantes b_1 e b_2 associadas à primeira e segunda derivadas, são definidas de forma semelhante ao caso unidimensional.

Petzold **[91]** utiliza uma perspectiva mais directa para o cálculo das velocidades nodais, generalizando o trabalho de *Hyman* **[68]** através da minimização da medida:

$$I = \sum_k \omega \cdot \left[\dot{X}^k\right]^2 + \left\|\dot{A}\right\|^2 \qquad (3.136)$$

em que: ω - constante de peso.

O segundo termo corresponde a uma função da velocidade nodal, por transferência do modelo para um domínio onde os nodos se mantêm estacionários, e aproximando espacialmente as P.D.E.´s por diferenças finitas, de forma a obter-se:

$$\dot{A}^k = \dot{X}^k \cdot \frac{\delta A^k}{\delta X} + f^k \cdot \left[A^k, \frac{\delta A^k}{\delta X}, \frac{\delta^2 A^k}{\delta X^2}\right] \qquad (3.137)$$

A equação para a velocidade do nodo k é deduzida por diferenciação de I em relação à velocidade desse nodo. Tem-se, assim:

$$\omega \cdot \dot{X}^k + \dot{A}^k \cdot \frac{\delta A^k}{\delta X} = 0 \qquad (3.138)$$

O objectivo consiste em minimizar a taxa de variação temporal da solução e da posição nodal de modo a que se possam aplicar passos maiores. Adiciona-se igualmente, um termo extra de forma a impedir o cruzamento entre nodos. Este termo é definido pela minimização de uma função *penalty* auxiliar, que apresenta valores elevados em zonas onde a separação nodal tende para zero. O modelo transformado e o sistema de equações para o movimento nodal formam um sistema de O.D.E.´s que é resolvido por um integrador implícito. Por outro lado, após cada passo temporal a grelha é redistribuída por ajuste do espaçamento nodal de forma a que a relação seguinte seja verificada:

$$\Delta X \cdot \left\|\frac{\delta A}{\delta X}\right\| + \left[\Delta X\right]^2 \cdot \left\|\frac{\delta^2 A}{\delta X^2}\right\| < \text{tol} \qquad (3.139)$$

O número de nodos é igualmente ajustado, de modo a ser o mínimo possível a verificar a tolerância fixada. A nova grelha é definida através da comparação entre a grelha antiga e uma grelha de referência, de forma a que a maioria dos nodos não se movimente, sendo apenas adicionados e retirados. Assim, a interpolação necessária à transferência da solução para a nova malha é efectuada somente num número reduzido de nodos, em cada passo temporal.

Verwer et al **[115]** apresentam uma estratégia de movimento da grelha baseada no princípio de equidistribuição nodal, juntamente com a aplicação de técnicas de suavização no espaço e no tempo propostas por *Dorfi e Drury* **[28]**. A suavização espacial assegura uma relação constante entre os comprimentos de malha adjacentes, enquanto a suavização temporal possibilita uma evolução suave da malha.

Para problemas enunciados da forma geral,

$$u_t = f(u, x, t), \qquad x_L < x < x_R,\ t > 0 \tag{3.140}$$

com as condições inicial e fronteiras:

$$\begin{aligned} &u(x,0) = u^0(x) && x_L < x < x_R \\ &b(u, x, t) = 0 && x = x_L, x_R,\ t > 0 \end{aligned} \tag{3.141}$$

efectua-se a discretização espacial recorrendo a diferenças finitas centradas. O sistema de O.D.E.´s resultante é resolvido através de um integrador temporal.

Considerando o conjunto de nodos móveis:

$$x_L = X_0 < \quad \cdots \quad < X_i(t) < X_{i+1}(t) < \quad \cdots \quad < X_{N+1} = x_R$$

que é introduzido, para as trajectórias $x(t) = X_i(t)$, na expressão da derivada total

$$u' = x' \cdot u_x + u_t = X_i' \cdot u_x + f(u, X_i(t), t) \qquad 1 \le i \le N \tag{3.142}$$

Aproximando as derivadas por diferenças finitas obtém-se:

$$U_i' = X_i' \cdot \frac{U_{i+1} - U_i}{X_{i+1} - X_i} + F_i \tag{3.143}$$

onde F_i é a aproximação numérica de f(u,x,t), baseada em diferenças finitas centrais de 3ª. ordem.

Passando para o sistema matricial tem-se:

$$\underline{U}' = \underline{X}' \cdot \underline{\underline{D}} + \underline{F} \tag{3.144}$$

com $t > 0$ e $U(0)$ conhecido.

O objectivo consiste na criação de uma grelha dependente do tempo e, implicitamente da solução u, que verifique a equação de equidistribuição espacial:

$$\frac{\eta_{i-1}}{M_{i-1}} = \frac{\eta_i}{M_i} \qquad 1 \le i \le N \tag{3.145}$$

onde: $\eta_i = 1/(X_{i+1} - X_i)$ - valores da concentração nodal;

M_i - valor discreto da função monitor m(u) no nodo i.

Definindo a função monitor $m(u) = (\alpha + u_x^2)^{1/2}$ obtém-se para o domínio discreto:

$$M_i = \left[\alpha + \left(\frac{U_{i+1} - U_i}{X_{i+1} - X_i} \right)^2 \right]^{1/2} \tag{3.146}$$

em que: α - constante (se α=1, o monitor é a função comprimento de arco).

O sistema (3.144) é resolvido simultaneamente com a equação (3.145) utilizando um integrador implícito.

O movimento nodal é regularizado de forma a suprimir oscilações ou distorções na malha. A distorção a nível espacial é impedida através da substituição das concentrações nodais pelas suas correspondentes numericamente "anti-difusas". Assim:

$$\begin{aligned}
&\tilde{\eta}_0 = \eta_0 - k \cdot (k+1) \cdot (\eta_1 - \eta_0) \\
&\dots \\
&\tilde{\eta}_i = \eta_i - k \cdot (k+1) \cdot (\eta_{i+1} - 2 \cdot \eta_i + \eta_{i-1}) \\
&\dots \\
&\tilde{\eta}_N = \eta_N - k \cdot (k+1) \cdot (\eta_{N-1} - \eta_N)
\end{aligned} \tag{3.147}$$

onde k - constante.

Porém, o critério de equidistribuição é mantido,

$$\frac{\tilde{\eta}_{i-1}}{M_{i-1}} = \frac{\tilde{\eta}_i}{M_i} \qquad 1 \le i \le N \tag{3.148}$$

Deste modo, a razão entre as concentrações nodais consecutivas está limitada por:

$$\frac{k}{k+1} \le \frac{\tilde{\eta}_{i-1}}{\tilde{\eta}_i} \le \frac{k+1}{k} \tag{3.149}$$

A introdução do procedimento de suavização temporal consiste na substituição do sistema de equações algébricas (3.148) pelos sistema de equações diferenciais seguinte:

$$\frac{\tilde{\eta}_{i-1} + \tau \cdot \tilde{\eta}'_{i-1}}{M_{i-1}} = \frac{\tilde{\eta}_i + \tau \cdot \tilde{\eta}'_i}{M_i} \qquad \tau > 0,\ 1 \le i \le N \tag{3.150}$$

O uso de derivadas da concentração nodal previne que o movimento da grelha apenas se ajuste aos novos valores do monitor. Deste modo, a grelha é forçada a ajustar-se durante o intervalo de tempo τ, dos anteriores para os novos valores do monitor. Assim, a escolha de um valor adequado para o parâmetro τ é bastante importante.

Rai e Anderson **[96,97]** e *Anderson e Rai* **[8]** desenvolveram um método para modelos uni- e bidimensionais baseado em pseudo-forças de atracção e repulsão entre os nodos. Nesta perspectiva, um nodo atrai os adjacentes se as medidas de E_i correspondentes forem superiores à média e repele-os em caso contrário. Os autores utilizam várias formas de E_i tais como: $|\delta A/\delta\xi_i|$; $|(\delta A/\delta\xi_i)/(\delta X_i/\delta\xi_i)|$; $|\delta^2 A/\delta\xi_i^{\,2}|$; e $|\delta X_i/\delta\xi_i|+\lambda|\delta A/\delta\xi_i|$.

A dependência temporal de X_i é calculada em cada nodo pelo somatório das pseudo-forças entre nodos, da forma seguinte: a partir de um problema bidimensional resolvido numa grelha rectangular, considera-se um espaço numérico de $N \times M$ nodos. A derivada temporal de ξ_1 num ponto fixo no espaço físico é definida para cada nodo de coordenadas (k,q) por,

$$\left.\frac{\delta \xi_1^{\,kq}}{\delta t}\right|_X = -\lambda_1 \cdot \sum_{m=1}^{M}\left[\sum_{n=k+1}^{N}\frac{E_1^{\,nm} - E_1^{\,avm}}{r^Q} - \sum_{n=1}^{k-1}\frac{E_1^{\,nm} - E_1^{\,avm}}{r^Q}\right] \qquad (3.151)$$

onde: $E_1^{\,nm}$ - medida do erro E_1 no nodo (n,m);

$E_1^{\,avm}$ - média da medida do erro E_1 ao longo de uma linha de ξ_1 constante no espaço físico;

r - distância na grelha numérica entre os nodos (k,q) e (n,m);

Q - constante positiva.

Se Q apresentar um valor baixo a aplicação da equação (3.151) provocará que, mesmo nodos distantes se influenciem entre si, o que é uma forma de suavização do movimento nodal.

Aplica-se uma equação equivalente para o cálculo de $\delta\xi_2/\delta t$:

$$\left.\frac{\delta \xi_2^{\,kq}}{\delta t}\right|_X = -\lambda_2 \cdot \sum_{n=1}^{N}\left[\sum_{m=q+1}^{M}\frac{E_2^{\,nm} - E_2^{\,avn}}{r^Q} - \sum_{m=1}^{q-1}\frac{E_2^{\,nm} - E_2^{\,avn}}{r^Q}\right] \qquad (3.152)$$

A transformação entre (X_i,t) e (ξ_i,t) é utilizada para computar as velocidades nodais respectivas a partir das derivadas temporais de ξ_1 e ξ_2. As constantes λ_1 e λ_2 são ajustadas em cada passo temporal de modo que as velocidades não excedam um valor limite máximo. Desta forma, as O.D.E.´s correspondentes ao movimento nodal são resolvidas simultaneamente com as O.D.E.´s do modelo por aplicação de método explícito de *MacCormack* **[80]**.

Greenberg **[55]** resolveu uma P.D.E. simples transformada para o sistema de coordenadas computacional (ξ,t). A equação é resolvida através de um método explícito de diferenças finitas centrais. Após cada passo temporal, aplicam-se também diferenças centrais para o cálculo dos gradientes locais da solução como medidas de erro E^n. Estas são utilizadas na definição de constantes k^{nm} aplicadas na dedução de O.D.E.´s para:

$$\Delta X^n = X^n - X^{n-1} \qquad (3.153)$$

Assim, a O.D.E.:

$$\frac{d\left(\Delta X^n\right)}{d\,t} = \sum_{m=1}^{N} k^{mn} \cdot \Delta X^m - \sum_{m=1}^{N} k^{nm} \cdot \Delta X^n \qquad (3.154)$$

assegura que a soma $\sum_{n=1}^{N} \Delta X^n$ permaneça constante. As constantes k^{nm} são escolhidas de maneira a que a separação nodal diminua em regiões caracterizadas por medidas de erro elevadas e vice-versa. Assim, k^{nm} tenderá a decrescer se a medida do erro fôr equidistribuida. A equação (3.154) é linearizada e resolvida analiticamente para determinar a distribuição nodal em cada passo temporal.

Madsen **[81]** apresenta um método adaptativo que aplicou a um sistema de duas P.D.E.´s unidimensionais. A equação definida para o cálculo das velocidades dos nodos é bastante semelhante à desenvolvida por *Klopfer e McRae* **[75]** e pode ser representada da forma:

$$\frac{d\left(\Delta X^n\right)}{d\,t} = \left\langle E \right\rangle - E^n \qquad (3.155)$$

onde: E^n - valor de uma medida do erro E, definida por uma relação pesada de vários componentes, entre o nodo n e o nodo n-1;

$\left\langle E \right\rangle$ - valor médio de E^n em todos os intervalos.

Cada componente para o cálculo de E^n pode consistir no valor absoluto duma estimativa do erro de truncatura espacial da aproximação do termo da direita das P.D.E.´s ou no valor absoluto de uma das grandezas seguintes: a variação, o declive ou a curvatura de cada componente da solução no intervalo definido entre os nodos n-1 e n. De forma a suavizar a distribuição nodal, inclui-se também o valor $|\Delta X^n|$ no cálculo de E^n. As constantes de peso são fixadas de modo que à medida que E^n varie no tempo, $\left\langle E \right\rangle$ se mantenha constante. A aproximação excessiva entre os nodos é impedida através do ajuste do valor de E^n correspondente.

O sistema de P.D.E.´s é aproximado recorrendo a diferenças finitas centrais e é resolvido em simultâneo com as equações de movimentação da malha, através de uma versão modificada do integrador-O.D.E. implícito de *Gear* **[49]**.

Winkler et al **[121]** aplicaram um método adaptativo para a determinação de fluxos gasosos e campos de radiação electromagnética na formação de estrelas que utiliza a expressão seguinte para a colocação nodal:

$$\eta^n - \eta^{n-1} = 0 \qquad (3.156)$$

onde:

$$\eta^n = \omega_x \cdot \Delta\alpha^n + \omega_{xz} \cdot \frac{\left[\Delta\alpha^n\right]^2}{\Delta\alpha^{n+1} - \Delta\alpha^{n-1}} + \Omega_x \cdot \Delta\beta^n + \Omega_{xz} \cdot \frac{\left[\Delta\beta^n\right]^2}{\Delta\beta^{n+1} - \Delta\beta^{n-1}} + \\ + \omega_M \cdot \frac{M^n - M^{n-1}}{M_{scale}} + \Omega_M \cdot \frac{M^n - M^{n-1}}{M^n + M^{n-1}} + \sum_k \Omega_k \cdot E_k^n \qquad (3.157)$$

Quando a equação (3.156) é satisfeita, cada termo da equação (3.157) é mais ou menos equidistribuida ao longo da malha na proporção do valor relativo das constantes de peso ω_x, ω_{xz}, Ω_x, Ω_{xz}, ω_M, Ω_M e Ω_k. As separações nodais normalizadas $\Delta\alpha^n$ e $\Delta\beta^n$ são definidas por:

$$\Delta\alpha^n = \frac{X^n - X^{n-1}}{X_{scale}} \qquad \text{e} \qquad \Delta\beta^n = \frac{X^n - X^{n-1}}{X^n + X^{n-1}} \qquad (3.158)$$

e M^n corresponde à variável de massa *lagrangiana*.

A equação (3.157) contém medidas de erro associadas à solução geralmente definidas por:

$$E_k^n = \frac{\left|G_k^n - G_k^{n-1}\right|}{G^n + G^{n+1}} \qquad (3.159)$$

onde G_k^n - valor de várias propriedades dos gases ou do campo de radiação electromagnética no nodo n.

A normalização resulta na equidistribuição da variação do logaritmo de G_k^n. Algumas expressões para medidas de erro não contêm denominadores sendo, nestes casos, equidistribuida a variação de G_k^n ao longo da grelha.

O segundo e quarto termos da expressão (3.157) actuam como resistência ao cruzamento nodal em alguns problemas.

As equações do modelo são discretizadas através de diferenças finitas e resolvidas conjuntamente com as equações de movimento nodal, usando um integrador O.D.E. implícito.

Winkler et al **[119]** introduzem na equação (3.157) o termo adicional:

$$+\omega_{MAX} \cdot \left[\frac{\Delta\alpha^n}{\Delta\alpha_{MAX}}\right]^Q + \omega_{MIN} \cdot \left[\frac{\Delta\alpha_{MIN}}{\Delta\alpha^n}\right]^Q + \Omega_{MAX} \cdot \left[\frac{\Delta\beta^n}{\Delta\beta_{MAX}}\right]^Q + \Omega_{MIN} \cdot \left[\frac{\Delta\beta_{MIN}}{\Delta\beta^n}\right]^Q$$

de modo a limitar a variação da separação nodal e do logaritmo de X^n entre limites máximos e mínimos especificados. Os autores também retiram a necessidade de utilização de valores absolutos nos termos da equação (3.157), substituindo-os pelo quadrado das expressões correspondentes.

Por outro lado, alteram a forma da expressão (3.156) para:

$$\eta^n - \eta^{n-1} - \omega_t \cdot \frac{d\,X^n}{d\,t} = 0 \qquad (3.160)$$

como forma de obtenção de suavização temporal. O peso temporal ω_t apenas é não-nulo se as medidas de erro decrescerem com o tempo, sendo proporcional ao quadrado do somatório das variações fraccionadas das medidas.

A partir dos trabalhos de *Winkler et al* [119-121], *Dorfi e Drury* [28] desenvolveram um método adaptativo semelhante que aplica equações de *Euler* unidimensionais contendo termos de viscosidade adicionais. Para tal definem uma medida de erro com k componentes da forma,

$$E^n = \left[1 + \sum_{k=1}^{K} \left[\frac{\beta^n}{\alpha_k^n} \cdot \frac{\Delta A_k^n}{\Delta X^n} \right]^2 \right]^{1/2} \tag{3.161}$$

em que: $\Delta A_k^n = A_k^n - A_k^{n-1}$ - variação da componente k da solução;

β^n, $\alpha_k{}^n$ - factores de escala associados à coordenada espacial e à solução, respectivamente.

Os nodos são movimentados de forma a que a relação $\beta^n/\Delta X^n$ seja proporcional ao valor médio de E^n no espaço ou no tempo, ou seja:

$$\frac{\tilde{\eta}^n}{E^n} = \frac{\tilde{\eta}^{n-1}}{E^{n-1}} \tag{3.162}$$

onde $\tilde{\eta}^n$ é a quantidade suavizada no tempo definida por:

$$\tilde{\eta}^n = \eta^n + \frac{\tau}{\Delta t} \cdot \left[\eta^n - \eta^n{}_{old} \right] \tag{3.163}$$

Esta equação corresponde na prática, a uma equação diferencial temporal da forma:

$$\eta^n = \frac{\beta^n}{\Delta X^n} - \gamma \cdot (\gamma + 1) \cdot \left[\frac{\beta^{n+1}}{\Delta X^{n+1}} - 2 \cdot \frac{\beta^n}{\Delta X^n} + \frac{\beta^{n-1}}{\Delta X^{n-1}} \right] \tag{3.164}$$

O valor de γ é seleccionado de forma a que ΔX^n não varie mais que 20-30 % de nodo para nodo, sendo $\eta^n{}_{old}$ definido como o valor de η^n no passo temporal anterior. Como já foi referido anteriormente, a escolha do valor de τ é determinante, porque para valores reduzidos, possibilita um tempo de resposta rápido à custa da perca de concentração dos nodos, no caso de cruzamento entre duas ondas. Porém, um τ elevado reduz a perca de concentração mas, diminui a capacidade dos nodos acompanharem frentes móveis. A diferença entre os valores actuais e anteriores de η^n, dividida por Δt pode ser expressa em termos das velocidades nodais.

Dorfi e Drury [28] aplicam uma técnica semelhante à utilizada por *White* [118] e *Winkler et al* [120]. Para tal, o sistema de P.D.E.´s do modelo é diferenciado e resolvido com a equação (3.162), através de um método iterativo de *Newton-Raphson*.

Finalmente, os autores sugerem a generalização do seu método a problemas multidimensionais, sugerindo a definição de E^n e η^n como tensores.

Hyman e Larrouturou [69] desenvolveram várias equações para grelhas móveis de forma a reduzir os erros de truncatura e aumentar a eficiência dos métodos numéricos de

resolução de P.D.E.´s. Para tal, apresentam vários tipos de estratégias relacionadas com a integração temporal e espacial, respectivamente:

⇨ **Distribuição dinâmica da variação temporal (T.V.):**

Através desta perspectiva pretende-se reduzir os erros de truncatura no tempo de modo a que seja possível aplicar maiores passos e se facilite a resolução de equações implícitas. Desta forma, os autores utilizaram o procedimento de *Hyman* **[68]** de modo a minimizar a variação temporal de U e X recorrendo à resolução de:

$$\min_{\dot{X}_i}\left[\left\|\dot{u}_i\right\|_j^2 + \alpha\cdot\dot{X}_i^2\right] = \min_{\dot{X}_i}\left[\sum\omega_j\cdot\left(F_i + U_x\left(X_i\right)\cdot\dot{X}_i\right)^2 + \alpha\cdot\dot{X}_i^2\right] \quad j = 1,\ \cdots,\ N_{eq} \tag{3.165}$$

onde ω_j, α- constantes de peso.

A equação (3.165) é quadrática e através da sua resolução obtém-se:

$$\dot{X}_i = -\sum\omega_j\cdot F_i\cdot U_x\left(X_i\right) \Big/ \left[\alpha + \sum\omega_j\cdot U_x\left(X_i\right)^2\right] \tag{3.166}$$

Esta estratégia é igualmente extensível a duas dimensões, podendo as equações para $\dot{X}$ e $\dot{Y}$ ser facilmente deduzidas.

⇨ **Malha equidistribuida (E.M.):**

Neste caso, as velocidades nodais são calculadas de forma a que os nodos se movimentem no sentido da redução dos erros de truncatura espaciais. *Hyman e Larrouturou* **[69]** apresentam duas estratégias para o desenvolvimento das expressões de movimentação nodal:

Método da equação elíptica - Neste caso, a função da malha, ou seja a função monitor, deve satisfazer a equação,

$$\dot{m}_{i+1/2} = \frac{\beta}{\tau}\cdot\left(\overline{m} - m_{i+1/2}\right) \tag{3.167}$$

onde: $m_{i+1/2}$ - valor da função monitor no ponto $X_{i+1/2} = 1/2\ (X_i + X_{i+1})$;
$\overline{m}$ - valor médio de m;
β - tempo de relaxação em relação à escala temporal τ.

Aproximando a expressão (3.167) e introduzindo as velocidades nodais definidas por:

$$\frac{\Delta\dot{X}_{i+1/2}}{\Delta X_{i+1/2}} = \frac{\beta}{\tau}\cdot\frac{\overline{m} - m_{i+1/2}}{m_{i+1/2}} \tag{3.168}$$

obtém-se o sistema linear para X:

$$m_{i+\frac{1}{2}} \cdot \frac{\Delta \dot{X}_{i+\frac{1}{2}}}{\Delta X_{i+\frac{1}{2}}} - m_{i-\frac{1}{2}} \cdot \frac{\Delta \dot{X}_{i-\frac{1}{2}}}{\Delta X_{i-\frac{1}{2}}} = \frac{\beta}{\tau} \cdot \left(m_{i+\frac{1}{2}} - m_{i-\frac{1}{2}} \right) \tag{3.169}$$

Deste modo, os nodos movimentam-se para as regiões onde o valor que a função da malha assume é mais elevado, mantendo-se estacionários se e só se a malha equidistribuir a função $m_{i+1/2} = \overline{m}$.

Método de aproximação linear - Nos casos onde há dúvidas quanto à validade da equação (3.168), os autores desenvolvem uma estratégia totalmente diferente para explicitar $\dot{X}$ de $\dot{m}$ na equação (3.167), recorrendo à função de malha simples:

$$m_{i+\frac{1}{2}} = a_0 \cdot \Delta X_{i+\frac{1}{2}} + \sum a_1 \cdot \left| \Delta U_{i+\frac{1}{2}} \right| + \sum a_2 \cdot \left| \Delta U_{x\ i+\frac{1}{2}} \right| \tag{3.170}$$

correspondente à equidistribuição duma aproximação da primeira e segunda derivadas espaciais, se ΔX fôr pequeno.

Por manipulação e aproximação das equações, com consequente substituição em (3.167), obtém-se a equação da malha:

$$\begin{aligned} &\left[a_0 + \sum_{eq} a_1 \cdot \left| U_x\left(X_{i+1}\right) \right| + \sum_{eq} a_2 \cdot \left| U_{xx}\left(X_{i+1}\right) \right| \right] \cdot \dot{X}_{i+1} - \\ &\left[a_0 + \sum_{eq} a_1 \cdot \left| U_x\left(X_{i+1}\right) \right| + \sum_{eq} a_2 \cdot \left| U_{xx}\left(X_{i+1}\right) \right| \right] \cdot \dot{X}_i = -\frac{\beta}{\tau} \cdot \left(m_{i+\frac{1}{2}} - \overline{m} \right) \end{aligned} \tag{3.171}$$

que pode ser resolvida directamente para $\dot{X}$.

Para definição da escala de tempo τ, os autores utilizam a expressão:

$$\tau^{-1} = \max\left\{ \tau^{-1}{}_{MAX}, \min_{eq} \min_{i} \left[\frac{\left| F_i(U) \right|}{\max\left(\left| U_i \right|, U_{floor} \right)} \right] \right\} \tag{3.172}$$

onde τ_{MAX} - constante que impede a anulação de τ, quando a solução atinge o estado estacionário;

U_{floor} - valor mínimo da solução pré-definido.

Os autores concluem que a aplicação simultânea dos métodos T.V. e E.M. é mais eficiente do que a sua utilização separada. Para tal, calculam as velocidades nodais através da função E.M. espacial - $\dot{X}^S$, e dos erros de minimização - $\dot{X}^t$. As velocidades combinadas $\dot{X}^b$ são definidas, então por:

$$\dot{X}^b = \theta_i \cdot \dot{X}^t + \left(1 - \theta_i\right) \cdot \dot{X}^S \tag{3.173}$$

Os factores de peso θ_i são escolhidos de forma a equilibrar os erros no espaço e no tempo da forma:

$$\theta_i = g_i^t \Big/ \left(g_i^t + g_i^S\right) \tag{3.174}$$

onde g_i^t, g_i^S - ganhos de exactidão aplicando as estratégias T.V. e E.M. respectivamente.

Cada um dos ganhos é definido pelas expressões:

$$g_i^t = \left|\left(U_t\right)_i - \dot{U}_i\right| = \left|F_i - \left(F_i + \dot{X}_i\left(U_x\right)_i\right)\right| = \left|\dot{X}_i^t\left(U_x\right)_i\right| \tag{3.175}$$

e

$$g_i^S = 1/2 \cdot \left(g_{i+1/2}^S + g_{i-1/2}^S\right) \tag{3.176}$$

com

$$g_{i+1/2}^S = \left|\dot{m}_{i+1/2}\left(\dot{X}=0\right) - \dot{m}_{i+1/2}\left(\dot{X}=\dot{X}^S\right)\right| = \frac{\beta}{\tau}\cdot\left|m_{i+1/2} - \overline{m}\right| \tag{3.177}$$

Por outro lado, é estabelecida uma limitação na variação do espaçamento da malha de forma a promover a regularização desta. Impõe-se, assim que:

$$\left|\Delta\dot{X}_{i+1/2} \Big/ \Delta X_{i+1/2}\right| \le \delta/\tau \tag{3.178}$$

em que δ - constante adimensional.

Inspirados pelo trabalho de *Ren e Russell* **[98]**, *Huang et al* **[64,65]** apresentam e deduzem vários tipos de P.D.E.´s relacionadas com o movimento da malha (denominadas M.M.P.D.E.´s - *Moving Mesh Partial Differential Equations*). Estes trabalhos baseiam-se em equações desenvolvidas a partir do princípio de equidistribuição, que são estudadas tanto do ponto de vista teórico como numérico. *Ren e Russell* **[98]** concluem que a equidistribuição implica a resolução de uma P.D.E. no domínio de coordenadas computacionais. Algumas das equações são originalmente deduzidas por *Huang et al* **[65]**, enquanto outras relacionam-se com a aplicação de estratégias desenvolvidas por outros autores. Assim, partindo de uma função monitor M(x,t), o princípio de equidistribuição (E.P.) pode ser expresso na forma integral por:

$$\int_0^{x(\xi,t)} M(x,t)\,dx = \xi\cdot\theta(t) \tag{3.179}$$

onde $\theta(t)$ é definida pela equação de *White* **[117]**.

Através de diferenciação simples ou dupla da equação (3.179) obtêm-se as duas formas diferenciais de E.P.:

$$M\left(x(\xi,t),t\right)\cdot\frac{\delta\, x(\xi,t)}{\delta\,\xi} = \theta(t) \tag{3.180}$$

e

$$\frac{\delta}{\delta\,\xi}\left\{M\left(x(\xi,t),t\right)\cdot\frac{\delta\, x(\xi,t)}{\delta\,\xi}\right\} = 0 \tag{3.181}$$

Estas equações que não contêm o termo de velocidade nodal, são denominadas E.P.´s quasi-estáticas (Q.S.E.P.´s). Através de manipulação destas equações *Huang et al* **[65]** deduzem vários tipos de M.M.P.D.E.´s. As equações mais importantes estão resumidas na Tabela 3.1 juntamente com as referências em que cada uma se baseia.

Tabela 3.1: Resumo das principais M.M.P.D.E.´s deduzidas por *Huang et al*[65].

M.M.P.D.E.	Equação	Ref[a].
1	$\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta\dot{x}}{\delta\xi}\right)+\frac{\delta}{\delta\xi}\left(\frac{\delta M}{\delta\xi}\cdot\dot{x}\right)=-\frac{\delta}{\delta\xi}\left(\frac{\delta M}{\delta t}\cdot\frac{\delta x}{\delta\xi}\right)$	**[65]**
2	$\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta\dot{x}}{\delta\xi}\right)+\frac{\delta}{\delta\xi}\left(\frac{\delta M}{\delta\xi}\cdot\dot{x}\right)=-\frac{\delta}{\delta\xi}\left(\frac{\delta M}{\delta t}\cdot\frac{\delta x}{\delta\xi}\right)-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[65]**
3	$\frac{\delta^2}{\delta\xi^2}(M\cdot\dot{x})=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[65]**
4	$\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta\dot{x}}{\delta\xi}\right)=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[65]**
5	$\dot{x}=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[7]**
6	$\frac{\delta^2\dot{x}}{\delta\xi^2}=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[55],[81]**
7	$\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta\dot{x}}{\delta\xi}\right)-\frac{2\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)\cdot\frac{\delta\dot{x}}{\delta\xi}}{\frac{\delta x}{\delta\xi}}=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left(M\cdot\frac{\delta x}{\delta\xi}\right)$	**[28]**

As equações consideradas não só forçam a equidistribuição da malha x(ξ,t) como impedem o cruzamento nodal. Assim, a resolução numérica é efectuada por intermédio de diferenças centrais na malha uniforme (ξ,t), e consequente aplicação do método das linhas.

Aplica-se, igualmente, uma função monitor alterada para a suavização da malha, em cada um dos nodos definida por**[64]**:

$$\tilde{M}_i=\sqrt{\frac{\sum_{k=i-p}^{i+p}\left(M_k\right)^2\cdot\left(\frac{\gamma}{1+\gamma}\right)^{|k-i|}}{\sum_{k=i-p}^{i+p}\left(\frac{\gamma}{1+\gamma}\right)^{|k-i|}}} \qquad (3.182)$$

onde, k - constante positiva (parâmetro de suavização);
p - inteiro não negativo (índice de suavização).

A suavização é obtida por substituição da função monitor M pela sua correspondente suavizada $\tilde{M}$.

Huang et al **[64]** utilizam a função monitor de comprimento de arco, discretizada por aplicação de diferenças finitas. O método de integração temporal escolhido consiste em fórmulas de discretização *backward*, enquanto que os sistemas de O.D.E.´s para a grelha móvel são resolvidos através do integrador-O.D.E. *stiff* DDASSL.

Por outro lado, *Budd et al* [21] aplicam algumas destas M.M.P.D.E.´s na resolução numérica de modelos caracterizados por *blow-up*, nos quais a definição de invariância escalar representa um papel muito importante da descrição da estrutura da solução. Nestes casos, a utilização de malhas não uniformes é essencial na reprodução do comportamento da solução na região de *blow-up*. Assim, os autores recorrem à utilização de métodos de malha móvel, em que a respectiva P.D.E. (correspondente ao modelo em estudo) e a M.M.P.D.E. escolhida (que descreve o comportamento da malha) são resolvidas simultaneamente.

De forma que a M.M.P.D.E. preserve a invariância escalar do problema global, são estudadas diferentes equações conjuntamente com diversas funções monitor M. Das várias equações propostas para a movimentação da malha (vd. Tabela 3.1), onde a transformação entre os sistemas coordenados físico e computacional é baseada no princípio de equidistribuição já referido (vd. equação (3.179)), aplicado a uma função monitor positiva M, consideram-se especificamente as equações denominadas por MMPDE4 e MMPDE6. Assim, são analisadas a aplicações de monitores apropriados, de forma a garantir a invariância da M.M.P.D.E. em relação à escala.

A computação numérica do problema é realizada em dois passos: inicialmente, procede-se à transformação do problema inicial para as coordenadas computacionais ξ; em seguida, realiza-se a discretização por diferenças finitas centradas, nesse domínio uniforme. Adicionalmente, aplica-se a estratégia de suavização das funções monitores testadas, através da transformação definida pela equação (3.182).

Huang e Russell [66] desenvolvem duas M.M.P.D.E.´s adicionais, que possuem propriedades de suavização espacial. A técnica de suavização usada é inspirada pelo método robusto de malha móvel proposto por *Dorfi e Drury* [28]. No caso de problemas que envolvam grandes variações, a evolução da função monitor não é muito suave no espaço. Então, é necessário considerar alguma estratégia de suavização de M(x,t) no E.P., de forma a suavizar a transformação. Deste modo, introduz-se no problema, uma P.D.E. em ξ e t, que envolva um termo adicional de difusão. Assim, para a suavização de M, os autores definem $\tilde{M}$, da forma:

$$\tilde{M} - \lambda^{-2}\Delta\tilde{M} = M \qquad \Rightarrow \qquad \tilde{M} = G^{-1}M \tag{3.183}$$

com as condições fronteira:

$$\frac{\delta \tilde{M}}{\delta \xi}(0,t) = \frac{\delta \tilde{M}}{\delta \xi}(1,t) = 0 \tag{3.184}$$

e $\Delta = \dfrac{\delta^2}{\delta \xi^2}$; com λ - parâmetro positivo.

Com a substituição de M, na equação (3.181), por $\tilde{M}$, obtém-se o princípio de equidistribuição suavizado:

$$\frac{\delta}{\delta \xi}\left\{\tilde{M}(\xi,t)\frac{\delta x}{\delta \xi}(\xi,t)\right\} = \frac{\delta}{\delta \xi}\left\{\left(G^{-1}M\right)\frac{\delta x}{\delta \xi}(\xi,t)\right\} = 0 \tag{3.185}$$

Aplicando o procedimento utilizado por *Huang et al* [65], deduz-se a partir de (3.185), a versão suavizada da equação MMPDE4, que apresenta a forma:

$$\frac{\delta}{\delta\xi}\left\{\frac{\delta\dot{x}}{\delta\xi}(\xi,t)G^{-1}M\right\}=-\frac{1}{\tau}\cdot\frac{\delta}{\delta\xi}\left\{\frac{\delta x}{\delta\xi}(\xi,t)G^{-1}M\right\} \tag{3.186}$$

Deste modo, de forma a se obter uma malha que seja simultaneamente suave no tempo e no espaço, pode-se resolver as equações (3.186) e (3.183), em conjunto, para x e $\tilde{M}$, respectivamente. No entanto, a equação adicional não envolve diferenciais temporais e, portanto, esta estratégia não é muito adequada para problemas dependentes do tempo. Por outro lado, na equação (3.183), G^{-1} é um operador integral, cuja discretização produz um sistema de matriz densa. Assim, para uma forma alternativa de integração em ξ, *Huang e Russell* **[66]** consideram a relação:

$$\left(G^{-1}M\right)\cdot\left(\frac{\delta\dot{x}}{\delta\xi}+\frac{1}{\tau}\cdot\frac{\delta x}{\delta\xi}\right)=c(t) \tag{3.187}$$

onde c(t) - constante integral; reescrevendo-a da forma:

$$\frac{\delta}{\delta\xi}\left\{\frac{G\left(\frac{1}{\frac{\delta\dot{x}}{\delta\xi}+\frac{1}{\tau}\cdot\frac{\delta x}{\delta\xi}}\right)}{M}\right\}=0 \tag{3.188}$$

para a qual se requer que:

$$\frac{\delta^2 x}{\delta\xi^2}(0,t)=\frac{\delta^2 x}{\delta\xi^2}(1,t)=0 \tag{3.189}$$

Portanto, a transformação x(ξ) é determinada pela resolução da M.M.P.D.E. suavizada:

$$\frac{\delta}{\delta\xi}\left\{\frac{\left(I-\lambda^{-2}\Delta\right)\left(\frac{1}{\frac{\delta\dot{x}}{\delta\xi}+\frac{1}{\tau}\cdot\frac{\delta x}{\delta\xi}}\right)}{M}\right\}=0 \tag{3.190}$$

juntamente com as condições fronteira (3.184) e (3.189).

Alternativamente, os autores derivam uma M.M.P.D.E. suavizada em x, em função da concentração de malha n, definida por:

$$n(\xi,t)=\frac{1}{\frac{\delta x}{\delta\xi}} \tag{3.191}$$

Assumindo que $\dot{n}$ é uniformemente delimitada em todo o domínio espacial ξ e temporal t, então, para τ reduzido:

$$\frac{1}{\frac{\delta \dot{x}}{\delta \xi}+\frac{1}{\tau}\cdot\frac{\delta x}{\delta \xi}}=\frac{\tau\cdot n}{1-\tau\cdot\frac{\dot{n}}{n}}=\tau\cdot n\cdot\left(1+\tau\cdot\frac{\dot{n}}{n}+\cdots\right)=\tau\cdot n+\tau^2\cdot\dot{n}+O\left(\tau^3\right) \qquad (3.192)$$

Substituindo (3.192) em (3.188), e truncando os termos de ordem elevada, obtém-se,

$$\frac{\delta}{\delta \xi}\left\{\frac{G\left(\dot{n}+\frac{1}{\tau}\cdot n\right)}{M}\right\}=0 \qquad (3.193)$$

na qual, a definição de G, obriga a que n satisfaça as condições fronteira:

$$\frac{\delta n}{\delta \xi}(0,t)=\frac{\delta n}{\delta \xi}(1,t)=0 \qquad (3.194)$$

Deste modo, a M.M.P.D.E. deduzida é,

$$\frac{\delta}{\delta \xi}\left\{\frac{\left(I-\lambda^{-2}\Delta\right)\left(\dot{n}+\frac{1}{\tau}\cdot n\right)}{M}\right\}=0 \qquad (3.195)$$

ou, de forma equivalente:

$$\frac{\delta}{\delta \xi}\left(\frac{\dot{\hat{n}}}{M}\right)=-\frac{1}{\tau}\cdot\frac{\delta}{\delta \xi}\left(\frac{\hat{n}}{M}\right) \qquad (3.196)$$

onde $\hat{n} = Gn$

Finalmente, *Huang e Russell* **[66]** demonstram que a solução das M.M.P.D.E.´s (3.190) e (3.196) e as discretizações correspondentes, por diferenças finitas, preservam as propriedades de não permissão de cruzamentos nodais e de quasi-uniformidade local da malha.

Huang e Sloan **[67]** procuram estudar os problemas da extensão da aplicação do princípio de equidistribuição a modelos bidimensionais, no contexto da geração de grelhas adaptativos.

Assim, introduzindo a transformação de coordenadas, para o caso mais geral, do domínio computacional D_C para o domínio físico D_P:

$$\underline{x}=\underline{x}\left(\underline{\xi}\right) \qquad (3.197)$$

onde $\underline{x}=\left[x_1,x_2,x_3\right]^T$

$$\underline{\xi} = \left[\xi_1, \xi_2, \xi_3\right]^T$$

considera-se que a variação do comprimento de arco de u ao longo do elemento de arco de $\underline{x}$ para $\underline{x}$ +dx pode ser expressa por:

$$ds = \left[\alpha^2 \cdot (du)^2 + d\underline{x}^T \cdot d\underline{x}\right]^{1/2} = \left[d\underline{x}^T \cdot M \cdot d\underline{x}\right]^{1/2} \tag{3.198}$$

onde

$$M = \alpha^2 \cdot \nabla u \cdot \nabla u^T + I \tag{3.199}$$

Aplicando a transformação (3.197) em (3.198) obtém-se a expressão para as coordenadas computacionais:

$$ds = \left[d\underline{\xi}^T \cdot J^T \cdot M \cdot J \cdot d\underline{\xi}\right]^{1/2} \tag{3.200}$$

onde $J = \left[\frac{\delta \underline{x}}{\delta \xi_1}, \frac{\delta \underline{x}}{\delta \xi_2}, \frac{\delta \underline{x}}{\delta \xi_3}\right]^T$ é o Jacobiano da transformação.

Assim, o princípio de equidistribuição requer que, se $u\left(\underline{x}\left(\underline{\xi}\right)\right)$ apresenta a mesma variação ds em qualquer elemento de arco, no domínio computacional de comprimento fixo $\left[d\underline{\xi}^T \cdot d\underline{\xi}\right]^{1/2}$, o termo da direita de (3.200) tem de ser independente de $\underline{\xi}$. Deste modo $\left[J^T \cdot M \cdot J\right]$ deverá ser independente de $\underline{\xi}$, ou seja:

$$\left[d\underline{\xi}^T \cdot J^T \cdot M \cdot J \cdot d\underline{\xi}\right]^{1/2} = \left[d\underline{\xi}^T \cdot \tilde{M} \cdot d\underline{\xi}\right]^{1/2} \tag{3.201}$$

onde $\tilde{M}$ é uma matriz 3×3 simétrica e positiva definida, independente de $\underline{\xi}$. Deste modo, se a transformação de variáveis (3.201) satisfizer a relação, u terá a mesma variação em qualquer ponto de D_P, ao longo de qualquer arco de comprimento:

$$\left[\left(\sum_{i=1}^{3} \frac{\delta \underline{x}}{\delta \xi_i} \cdot d\xi_i\right)^T \cdot \left(\sum_{j=1}^{3} \frac{\delta \underline{x}}{\delta \xi_j} \cdot d\xi_j\right)\right]^{1/2}$$

Assim, a condição (3.201) é designada pelo princípio de equidistribuição na forma contínua. Por exemplo, ao longo da linha coordenada l_1: $\xi_2, \xi_3 = c^{te}$, (3.201) toma a forma:

$$\left[\left(\frac{\delta \underline{x}}{\delta \xi_1}\right)^T \cdot M \cdot \frac{\delta \underline{x}}{\delta \xi_1}\right]^{1/2} = c_1 \tag{3.202}$$

sendo c_1, uma constante positiva, que através da equidistribuição pode ser definida por:

$$c_1 = \frac{\int \left[\left(\frac{\delta \underline{x}}{\delta \xi_1} \right)^T \cdot M \cdot \frac{\delta \underline{x}}{\delta \xi_1} \right]^{1/2} d\xi_1}{\int d\xi_1} \tag{3.203}$$

integrado ao longo de l_1 em D_C. Considerando todas as coordenadas, obtêm-se fórmulas análogas para cada direcção. Assim, tem-se que:

$$\begin{cases} \left[\left(\frac{\delta \underline{x}}{\delta \xi_1} \right)^T \cdot M \cdot \frac{\delta \underline{x}}{\delta \xi_1} \right]^{1/2} = c_1(\xi_2, \xi_3) \\ \left[\left(\frac{\delta \underline{x}}{\delta \xi_2} \right)^T \cdot M \cdot \frac{\delta \underline{x}}{\delta \xi_2} \right]^{1/2} = c_2(\xi_1, \xi_3) \\ \left[\left(\frac{\delta \underline{x}}{\delta \xi_3} \right)^T \cdot M \cdot \frac{\delta \underline{x}}{\delta \xi_3} \right]^{1/2} = c_3(\xi_1, \xi_2) \end{cases} \tag{3.204}$$

Desta forma, obtêm-se as formas bi- e unidimensionais de (3.204):

$$\begin{cases} \left[\begin{pmatrix} \frac{\delta x}{\delta \xi} \\ \frac{\delta y}{\delta \xi} \end{pmatrix}^T \cdot M \cdot \begin{pmatrix} \frac{\delta x}{\delta \xi} \\ \frac{\delta y}{\delta \xi} \end{pmatrix} \right]^{1/2} = c_1(\eta) \\ \left[\begin{pmatrix} \frac{\delta x}{\delta \eta} \\ \frac{\delta y}{\delta \eta} \end{pmatrix}^T \cdot M \cdot \begin{pmatrix} \frac{\delta x}{\delta \eta} \\ \frac{\delta y}{\delta \eta} \end{pmatrix} \right]^{1/2} = c_2(\xi) \end{cases} \tag{3.205}$$

com

$$M = \alpha^2 \cdot \begin{bmatrix} \left(\frac{\delta u}{\delta x} \right)^2 & \frac{\delta u}{\delta x} \cdot \frac{\delta u}{\delta y} \\ \frac{\delta u}{\delta x} \cdot \frac{\delta u}{\delta y} & \left(\frac{\delta u}{\delta y} \right)^2 \end{bmatrix} + \begin{bmatrix} 1 & 0' \\ 0 & 1 \end{bmatrix} \tag{3.206}$$

e

$$\left[\left(\frac{d x}{d \xi} \right)^2 \cdot M \right]^{1/2} = c \tag{3.207}$$

com

$$M = 1 + \alpha^2 \cdot \left(\frac{d\,u}{d\,x} \right)^2 \tag{3.208}$$

onde (3.208) representa o princípio de equidistribuição de comprimento de arco, unidimensional.

O princípio de equidistribuição não conduz a nenhum mecanismo de controlo da qualidade da malha, no que diz respeito à sua suavidade ou ortogonalidade ao longo das fronteiras. Assim, esses mecanismos devem ser introduzidos adicionalmente. Nos métodos variacionais, a relação entre a adaptação e a qualidade da malha é feita através da introdução de uma combinação linear de termos associados à adaptação, suavidade e ortogonalidade. Neste trabalho, *Huang e Sloan* **[67]** estendem a estratégia de suavização unidimensional de *Dorfi e Drury* **[28]** (de forma a impedir o desenvolvimento de *skewness* na malha), substituindo a matriz M discretizada pelas correspondentes suavizadas $\tilde{M}$ bidimensionais, definidas por:

$$\tilde{M}_{i+1/2,j} = \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} M_{k+1/2,l} \cdot \left(\frac{\gamma}{\gamma + 1} \right)^{|k-i|+|l-j|} \tag{3.209}$$

e

$$\tilde{M}_{i,j+1/2} = \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} M_{k,l+1/2} \cdot \left(\frac{\gamma}{\gamma + 1} \right)^{|k-i|+|l-j|} \tag{3.210}$$

Por outro lado, para casos bastante *stiff*, podem ocorrer espaçamentos muito pequenos na grelha, quando $|\nabla u|$ se torna bastante elevado. Este problema é evitado, substituindo a expressão (3.199) por:

$$M = \frac{\alpha^2 \cdot \nabla u \cdot \nabla u^T}{1 + \beta \cdot \nabla u^T \cdot \nabla u} + I \tag{3.211}$$

onde β é uma constante positiva, o que implica a verificação da relação:

$$\|M\|_2 \leq 1 + \frac{\alpha^2}{\beta} \tag{3.212}$$

impondo-se, assim, valores mínimos para o espaçamento em x e y, ao longo da cada coordenada computacional. Deste modo, introduzem-se, no método adaptativo, os parâmetros α, β e γ relacionados com a concentração, a definição da escala, e suavização da malha, respectivamente.

Daripa **[25]** apresenta uma nova teoria para a geração de grelhas adaptativas a uma dimensão espacial. Esta perspectiva baseia-se em aproximações da resolução e da razão de espaçamento da grelha, em pontos discretos, por variáveis contínuas. A ordem de exactidão destas aproximações em sistemas de coordenadas adequados, caracteriza os vários métodos de geração, sendo utilizadas aproximações de primeira e de segunda ordem.

A definição de resolução da grelha é realizada, considerando a necessidade das concentrações nodais serem elevadas em zonas de variação rápida da função do modelo f(x). Para isso, o autor define a resolução pela expressão:

$$s(x) = \frac{d\,\xi}{d\,x} \tag{3.213}$$

Como foi referido anteriormente, a estratégia consiste na aproximação numérica da resolução (s) e contínua da razão de espaçamento (r_i), de forma a se estabelecer uma relação entre as coordenadas discretas da grelha adaptativa e a transformação x_ξ. Todas as aproximações são efectuadas recorrendo ao desenvolvimento de séries de *Taylor*. Deste modo, obtêm-se as expressões seguintes:

Resolução:

Primeira ordem ⇨ $$n_i = \frac{h_\xi}{x_{i+1} - x_i} \tag{3.214}$$

Segunda ordem ⇨ $$n_i = \frac{2 \cdot h_\xi}{x_{i+1} - x_i} \tag{3.215}$$

onde n_i corresponde à aproximação numérica n_i a s, no ponto da grelha x_i.

Razão de espaçamento:

Primeira ordem ⇨ $$R_3(x_i) = \frac{x_\xi(\xi_i)}{x_\xi(\xi_{i-1})} \tag{3.216}$$

Segunda ordem ⇨ $$R_1(x) = 1 + \alpha(x) \text{ e } R_2(x) = \frac{2 + \alpha(x)}{2 - \alpha(x)} \tag{3.217}$$

onde: $$\alpha(x) = h_\xi \cdot \frac{x_{\xi\xi}}{x_\xi} \tag{3.218}$$

e $R_i(x)$ são aproximações contínuas a r_i, sujeitas à restrição:

$$0 < r_L \le r(x_i) \le r_U, \quad \forall x_i \in D_x$$

que impõe a seguinte restrição para α:

$$a \le \alpha(x) \le b, \quad \forall x \in D_x$$

com: $$a = \begin{cases} r_L - 1 & \rightarrow \quad R = R_1 \\ \dfrac{2 \cdot r_L - 1}{r_L + 1} & \rightarrow \quad R = R_2 \end{cases} \text{ e } b = \begin{cases} r_U - 1 & \rightarrow \quad R = R_1 \\ \dfrac{2 \cdot r_U - 1}{r_U + 1} & \rightarrow \quad R = R_2 \end{cases} \tag{3.219}$$

A geração da grelha é obtida por integração directa de

$$x_\xi = c(k) \cdot w(x;k), \qquad x_L \le x \le x_R \tag{3.220}$$

onde: c(k) - parâmetro de escala que controla a transformação de D_x para D_ξ;
w(x;k) - função escolhida de modo a apresentar a forma:

$$w(x;k) = \tilde{w}\left(x, f(x), f_{xx}; k\right) \tag{3.221}$$

O parâmetro k é introduzido para se poder ajustar a função, de forma a se obterem as fronteiras desejadas para a razão de espaçamento. Obviamente que o tipo de adaptação vai depender da função w escolhida. *Daripa* [25] utiliza vários tipos de w, dependentes de f, da sua primeira e/ou segunda derivada ou apenas da coordenada espacial. Cada uma das funções revela-se apropriada para diversos tipos de perfis acentuados na solução.

3.3.2.4 - Métodos Adaptativos Baseados em Aproximações de Volumes Finitos

A partir de estratégias variacionais, foram desenvolvidos igualmente, vários métodos adaptativos para problemas multidimensionais baseados na discretização de P.D.E.´s por métodos de volumes finitos. Um método de volumes finitos [59] é uma aproximação geometricamente conservativa da forma integral da P.D.E.. O domínio é subdividido em volumes de controlo e as equações de diferenças finitas são obtidas por conservação dos fluxos intervolumes. Assim, o sistema de equações mantém a conservação da massa, energia e momento, de elemento para elemento.

Carcaillet et al [22] apresentaram um método deste tipo, aplicado à perspectiva variacional da geração de grelhas adaptativas, baseando-se no controlo da soma, para todos os nodos, de medidas de suavidade e ortogonalidade da grelha e suavidade da solução. O método é uma extensão do trabalho de *Hayes et al* [60] e *Kennon e Dulikravich* [74] que apenas utilizaram as primeiras duas medidas.

Inicialmente é calculada a área das quatro células adjacentes ao nodo. Deste modo, é monitorizada a suavidade da grelha. A ortogonalidade da grelha é computada pela soma dos quadrados dos produtos dos vectores de posição relativa, em relação aos nodos adjacentes. Finalmente, a suavidade da solução é controlada pelo cálculo de uma medida de erro definida pelo produto das áreas das células adjacentes. O somatório das medidas em relação a todos os nodos do domínio define uma medida global da qualidade da grelha que é análoga ao integral de volume de *Brackbill e Saltzman*.

Os autores utilizam um procedimento simplex não linear de forma a variar as posições dos nodos até que a medida de erro seja minimizada.

Kennon e Dulikravich [74] procedem igualmente à extensão dos cálculos das propriedades da grelha para casos tridimensionais.

Gnoffo [52,53], a partir do trabalho de *Dwyer et al* [33] e de *Rai e Anderson* [96,97], utiliza uma medida de erro de forma a definir forças de tensão entre nodos. Assim, os nodos são movidos de modo a equidistribuir as forças ao longo de linhas de coordenadas computacionais constantes.

As forças entre nodos adjacentes são definidas por:

$$F = K \cdot \Delta S \tag{3.222}$$

onde: K - constante elástica;
ΔS - comprimento de arco entre os nodos.

Por sua vez, K é calculado por:

$$K = 1 + E \tag{3.223}$$

onde E é a medida do erro, definida pela soma pesada das magnitudes das derivadas espaciais de cada componente da solução.

A adaptação é realizada entre vários passos de tempo, movimentando os nodos ao longo de curvas de coordenada computacional constante, através de um método iterativo.

Nakahashi e Deiwert **[86]** aperfeiçoam o trabalho de *Gnoffo* **[52,53]**, adicionando forças de torsão na implementação do método anterior. Assim, os autores ajustam a distribuição nodal no domínio bidimensional de forma a impor a ortogonalidade da grelha, assim, como a sua adaptação. Esta é aplicada ao longo de uma direcção da coordenada computacional, em tempos discretos.

A localização dos nodos é definida por um balanço de forças a cada nodo, de forma a se obter um sistema tridiagonal de equações que pode ser resolvido para cada nodo ao longo de linhas de i constante. Assim, para o nodo (i,j) tem-se que,

$$K_{i,j} \cdot \left[S_{i,j+1} - S_{i,j}\right] - K_{i,j-1} \cdot \left[S_{i,j} - S_{i,j-1}\right] - T_{i-1,j} \cdot \left[S_{i,j} - \tilde{S}_{i,j}\right] = 0 \tag{3.224}$$

onde:

$S_{i,j}$ - comprimento de arco entre o nodo (i,1) e (i,j), ao longo da linha de i constante;

$K_{i,j}$ - constante elástica de tensão entre os nodos (i,j) e (i,j+1);

$\tilde{S}_{i,j}$ - projecção da linha entre os nodos (i,j) e (i-1,j) na linha de i constante;

$T_{i-1,j}$ - constante elástica de torsão entre os nodos (i-1,j) e (i,j).

Este método é igualmente generalizado para geometrias tridimensionais por *Nakahashi e Deiwert* **[85]**.

Eiseman **[39]** aplica um método adaptativo em grelhas bidimensionais com células rectangulares. Para tal, calcula-se uma medida do erro definida como a soma das áreas das células adjacentes e das áreas pesadas da células definidas por vectores de posição relativa em relação aos quatro pontos vizinhos. Estas quantidades contêm o gradiente da solução, a curvatura da linha, o desvio da ortogonalidade e da conformidade da grelha.

Os nodos são movimentados a partir da medida do erro, considerando os quatro vectores posicionais que são divididos em dois pares de vectores opostos R^+ e R^-, a que estão associados as medidas de erro E^+ e E^-. Se $E^+ \cdot \left\|R^+\right\| > E^- \cdot \left\|R^-\right\|$ gera-se um vector de deslocamento:

$$\frac{E^+ \cdot \left\|R^+\right\| - E^- \cdot \left\|R^-\right\|}{3 \cdot \left[E^+ + E^-\right] \cdot \left\|R^-\right\|} \cdot R^+$$

Caso contrário, o vector de deslocamento é definido por:

$$\frac{E^- \cdot \left\|R^-\right\| - E^+ \cdot \left\|R^+\right\|}{3 \cdot \left[E^+ + E^-\right] \cdot \left\|R^-\right\|} \cdot R^-$$

O outro par de vectores opostos é utilizado para estimar um segundo vector de deslocamento. Então, a nova posição do nodo em relação à sua posição anterior, é obtida por introdução da soma dos dois vectores calculados.

Depois da adaptação de toda a grelha, a distribuição dos nodos é suavizada por um método descrito por *Eiseman* [37] de modo a remover oscilações nas posições nodais.

Noutro trabalho, *Eiseman* [40] aplica procedimentos semelhantes em grelhas bidimensionais com células triangulares, discutindo, igualmente, estratégias de remoção e adição de nodos.

Por outro lado, *Eiseman* [37,41] aplica um método de adaptação unidimensional em linhas de coordenadas curvilíneas constantes, semelhante ao de *Nakahashi e Deiwert* [86], para grelhas bi- e tridimensionais. Para tal, os nodos são movimentados ao longo de linhas de coordenadas computacionais constantes de forma a equidistribuir a função do erro calculada entre pares de nodos.

✧✧✧✧✧

Paralelamente aos métodos gerais, têm sido igualmente apresentados métodos adaptativos cuja aplicação se restringe ao tipo muito limitado de modelos para os quais foram desenvolvidos. Esta classe de métodos de geração de grelhas adaptativas geralmente, pressupõe um conhecimento bastante profundo *à priori* do tipo de P.D.E.´s a integrar, assim como do comportamento tipo dos perfis da solução desenvolvidos. Deste modo, as estratégias propostas nesta área podem-se inserir em qualquer das classes gerais de métodos adaptativos referidos anteriormente.

Dentro de um número bastante vasto de trabalhos apresentados nesta área, escolheram-se alguns exemplos recentes, entre os quais se salienta o método adaptativo de redistribuição nodal estática de *Larrouturou* [78], aplicado a problemas unidimensionais de propagação de chama, no qual a velocidade da grelha é calculada em função do integral espacial da velocidade da reacção normalizada Ω e das derivadas espaciais da temperatura normalizada. De facto, todo o campo de problemas associados a combustão tem-se revelado fértil para o desenvolvimento de métodos adaptativos específicos e mesmo, gerais.

Trangenstein [111] estuda a aplicação de um algoritmo de refinamento de malha utilizado para problemas instáveis de dinâmica de gases, a modelos de propagação de ondas em sólidos não-lineares. Este algoritmo é associado a técnicas *standard* de identificação de choques.

Strain [106] desenvolve métodos adaptativos eficientes para a resolução de equações de calor no espaço livre, que podem ser aplicados com passos temporais elevados. O algoritmo baseia-se na combinação entre uma transformação rápida de *Gauss* e esquemas de refinamento adaptativo que representam a solução como uma aproximação polinomial contínua de ordem variável.

Adicionalmente, o problema de desenvolvimento de métodos adaptativos eficientes já se encontra suficientemente generalizado de modo a ser abordado em livros científicos com preocupações de carácter mais geral. Assim, *Finlayson* [44] refere e compara vários tipos de metodologias adaptativas simples, aplicadas a discretizações baseadas em **Diferenças e Elementos Finitos**, discutindo as vantagens e desvantagens de cada método aplicado a problemas caracterizados por ondas móveis. Paralelamente, *Schiesser* [103] apresenta estratégias adaptativas baseadas no **Método das Linhas**, ou seja, pertencentes ao grupo de

métodos de **Diferenças Finitas**. No entanto, os algoritmos referidos são pouco elaborados, constituindo apenas a ilustração de aplicações de estratégias de movimentação nodal a casos típicos simples.

✧✧✧✧✧

Na tabela das páginas seguintes são resumidos e enquadrados a maioria dos métodos apresentados neste trabalho, baseados em discretizações de diferenças finitas:

Tabela 3.2: Apresentação resumida das referências bibliográficas.

Método		*Estratégia*			*Dimensão*	*Referência*
Estático	B.V.P.	Equidistribuição de um monitor				*Pereyra e Sewell*[89], *Carey e Dinh*[23] e *Chen*[24]
						White[117]
						Russell e Christiansen[100]
						Ablow e Schechter[2]
		Minimização do integral do erro				*Gough et al*[54]
						Denny e Landis[27]
	Refinamento	Estimativas de erro de truncatura			Unidimensional	*Hyman e Naughton*[70]
						Smooke e Koszykowski[105]
						Nowak et al[87], *Oliveira et al*[88] e *Guiné*[58]
					Bidimensional	*Berger e Oliger*[15] e *Berger e Colella*[14]
						Gropp[56,57] e *Arney e Flaherty*[10,11]
						Trompert e Werver[112,113]
						Lee e Tsuei[79]
		Resolução de equações de *Poisson* (Variacional)				*Acharya e Monkalled*[3]
		Estimativa de erro de quadratura				*Altas e Stephenson*[5]
		Controlo de derivadas			Unidimensional	*Kulkarni e Dudukovic*[77]
		Identificação de ondas				*Fraga e Morris*[47]
		Estimativas de erro de truncatura			Tridimensional	*Bell et al*[12]
						Fung et al[48]
	Redistribuição	Minimização do integral do erro			Unidimensional	*Pierson e Kutler*[92]
						Klopfer e McRae[75]
		Equidistribuição de um monitor				*Sanz-Serna e Christie*[102] e *Revilla*[99]
						Verwer et al[116] e *Blom et al*[16]
		Variacional	Equidistribuição de um monitor		Bidimensional	*Dwyer et al*[30-36]
						Brackbill e Saltzman[18,19], *Brackbill*[17] e *Saltzman e Brackbill*[101]
					Unidimensional	*Bell e Shubin*[13]
			Volumes Finitos	Minimização do integral do erro	Bidimensional	*Carcaillet et al*[22], *Hayes et al*[60] e *Kennon e Dulikravich*[74]
				Pseudo-forças entre nodos		*Gnoffo*[52,53]
					Bi- e tridimensional	*Nakahashi e Deiwert*[85,86]
						Eiseman[37,39-41]

(continua na página seguinte)

(continuação da página anterior)

Dinâmico	Equidistribuição de um monitor		Unidimensional	*White*[118]
				Ren e Russel[98] e *Huang et al*[64,65], *Budd et al*[21] e *Huang e Russell*[66]
			Bidimensional	*Huang e Sloan*[67]
	Minimização do integral do erro		Unidimensional	*Verwer et al*[115]
				Kansa et al[73] e *Ghia et al*[51]
				Petzold[91] e *Hyman*[68]
	Equidistribuição de um monitor		Multidimensional	*Anyiwo*[9]
	Variacional	Minimização do integral do erro	Bidimensional	*Yanenko et al*[123-125]
			Uni- e bidimensional	*Hindman e Spencer*[63]
		Equidistribuição de um monitor	Uni- e Tridimensional	*Matsuno e Dwyer*[82]
	Pseudo-forças entre nodos		Uni- e bidimensional	*Rai e Anderson*[96,97] e *Anderson e Rai*[8]
			Unidimensional	*Greenberg*[55]
				Madsen[81]
				Winkler et al[119-121]
				Dorfi e Drury[28]
				Hyman e Larrouturou[69]
	Equidistribuição de um monitor			*Daripa*[25]

4. Algoritmos Numéricos Adaptativos

O objectivo do presente capítulo consiste na apresentação dos **Métodos Numéricos Adaptativos** aplicados e estudados no decorrer do trabalho efectuado. Pretende-se que os algoritmos descritos sejam o mais versátil e eficientes possível na resolução de sistemas de equações diferenciais ou algébrico-diferenciais parciais (P.D.E.´s ou P.D.A.E.´s) unidimensionais, cuja solução exiba um comportamento caracterizado por variações bruscas no espaço. Os algoritmos referidos baseiam-se essencialmente no **Método das Linhas**, que pode ser resumido da seguinte forma: efectua-se uma aproximação das derivadas espaciais através de relações algébricas obtendo-se um sistema de O.D.E.´s dependente da solução e das posições relativas na malha discretizada; este sistema é integrado na direcção temporal por intermédio de um integrador numérico. No entanto, no caso de algoritmos adaptativos, recorre-se à introdução, no procedimento referido, de uma estratégia que possibilite o deslocamento dos nodos para regiões do domínio onde a dificuldade de integração seja maior (zonas de maior actividade da solução) e onde serão mais necessários. Duma forma resumida e simplista é possível dividir as principais estratégias utilizadas para o efeito em dois grupos gerais:

❶ Refinamento ou alargamento da grelha - consiste na introdução e/ou eliminação de nodos nas zonas de maior e menor actividade da solução, respectivamente;

❷ Deslocamento dos nodos para as zonas do domínio de maior actividade da solução - subdividem-se em métodos estáticos e dinâmicos de movimentação nodal.

4.1 – Formulação Matemática Geral dos Modelos

Considere-se, então, o seguinte modelo geral para os sistemas de P.D.E.´s ou P.D.A.E.´s que se pretendem resolver através da utilização de algoritmos adaptativos:

$$F\left(u_t, u, u_z, u_{zz}\right) = 0 \qquad (4.1)$$

e

$$G\left(u\right) = 0 \qquad (4.2)$$

Condições fronteira:

$$u\left(z^L, t\right) = u^L \qquad (4.3)$$

$$u\left(z^R, t\right) = u^R \qquad (4.4)$$

Condição inicial:

$$u\left(z,0\right) = u^0\left(z\right) \quad ; \quad z \in \left[z^L, z^R\right] \qquad (4.5)$$

onde,

$$u\left(z,t\right) = \left[u^1\left(z,t\right), u^2\left(z,t\right), \quad \cdots \quad , u^N\left(z,t\right)\right]$$

Aplicando uma discretização espacial ao modelo anterior, obtém-se um sistema de O.D.E.´s a resolver ao longo do tempo: numa malha móvel (Estratégia ❷), através de sucessivos refinamentos de zonas do domínio seleccionadas (Estratégia ❶) ou da conjugação de ambos os procedimentos.

4.2 – Método de Refinamento de Malha

Para o desenvolvimento de uma estratégia de refinamento é necessário a definição de uma medida de erro espacial que de certo modo permita efectuar uma quantificação do erro associado à discretização do modelo para uma determinada grelha. Deste modo, é possível identificar as zonas do domínio, onde o refinamento se revela necessário.

O algoritmo aplicado neste trabalho baseia-se no método de refinamento enunciado em *Guiné*[58], sendo a estimativa do erro espacial efectuada pela comparação entre duas soluções obtidas a partir de duas grelhas diferentes: uma grelha mais fina constituída por 2×NP-1 pontos, definida por subdivisão de uma grelha larga de NP pontos. As duas malhas não têm que ser necessariamente uniformes, obtendo-se a malha fina por bissecção dos intervalos da malha grossa. Deste modo, sendo: (com n=2, 3, …. , N_{max})

Wh ⇨ Aproximação da solução obtida pela integração entre t_k e t_{k+1} na malha mais fina (malha de nível n);
W2h ⇨ Aproximação da solução obtida pela integração entre t_k e t_{k+1} na malha mais larga (malha de nível n-1),

a estimativa do erro é realizada pela expressão seguinte:

$$EU^{i}_{j,k+1} = Wh^{i}_{j,k+1} - W2h^{i}_{j,k+1} \qquad \begin{matrix} j = 1, \quad \cdots \quad , NP_{n-1} \\ i = 1, \quad \cdots \quad , NPDE \end{matrix} \tag{4.6}$$

onde: NP_{n-1} – número de nodos da malha de nível n-1;
NPDE – número de equações diferenciais parciais do modelo.

Se para o tempo k+1, no ponto j da grelha de nível n-1, se obtiverem valores do indicador de erro para a variável i, superiores a uma tolerância pré-definida:

$$\left|EU^{i}_{j,k+1}\right| > TOL_i, \qquad i = 1, \quad \cdots \quad , NPDE \tag{4.7}$$

então, torna-se necessário refinar os subdomínios $[z_{j-1},z_{j+1}]$, ou seja, introduzem-se pontos nas posições médias dos intervalos que definem os subdomínios, e integra-se de novo o modelo entre os instantes de tempo referidos, utilizando a nova malha de nível n+1.

Obviamente, que antes de se passar a uma grelha de nível superior, é necessário estimar o erro em todos os pontos do domínio total (para n=2) ou dos subdomínios obtidos anteriormente (para n>2) por comparação entre as soluções de nível n e n-1. De seguida, procede-se à junção de todos os pontos marcados (que não verificam a tolerância TOL_i definida para a variável i) em zonas de refinamento delimitadas pelos primeiros pontos da malha de grau n-1 que satisfaçam a tolerância requerida. Formadas estas zonas de refinamento, é possível fazer avançar a integração de cada uma, no novo nível de refinamento

n+1, entre t_k e t_{k+1}. Este procedimento é repetido sucessivamente, até que a solução em todos os pontos de cada uma das malhas formadas verifique as tolerâncias especificadas, ou o nível de refinamento máximo (N_{max}) seja ultrapassado. Nesse caso, todo o algoritmo é repetido desde o nível de refinamento mínimo 1, a partir de t_k, com um passo temporal reduzido a metade. Logo que se atinja o tempo t_{k+1}, a integração seguinte é efectuada com o passo temporal base Δt_k.

Em cada um dos refinamentos, os perfis da solução de arranque da integração são obtidos pela interpolação do perfil da solução de nível 2, nas posições intermédias.

Na Figura 4.1 apresenta-se, a título ilustrativo, um esquema resumido da metodologia aplicada e descrita anteriormente:

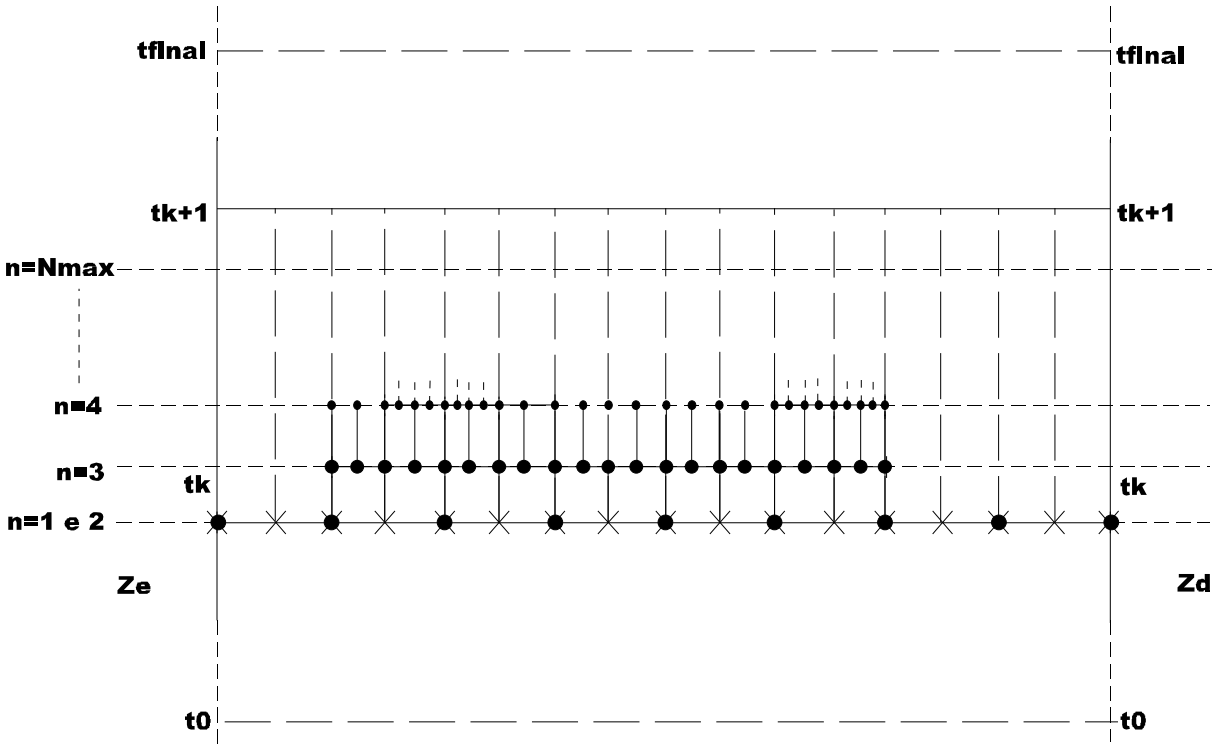

Figura 4.1: Esquema resumido da metodologia de refinamento.

A partir da solução nos pontos das malhas de nível 1 e 2, obtidas no passo anterior aplica-se o algoritmo para o cálculo do perfil da solução em t_{k+2}. Assim, o método de refinamento pode ser esquematizado da forma apresentada na Figura 4.2.

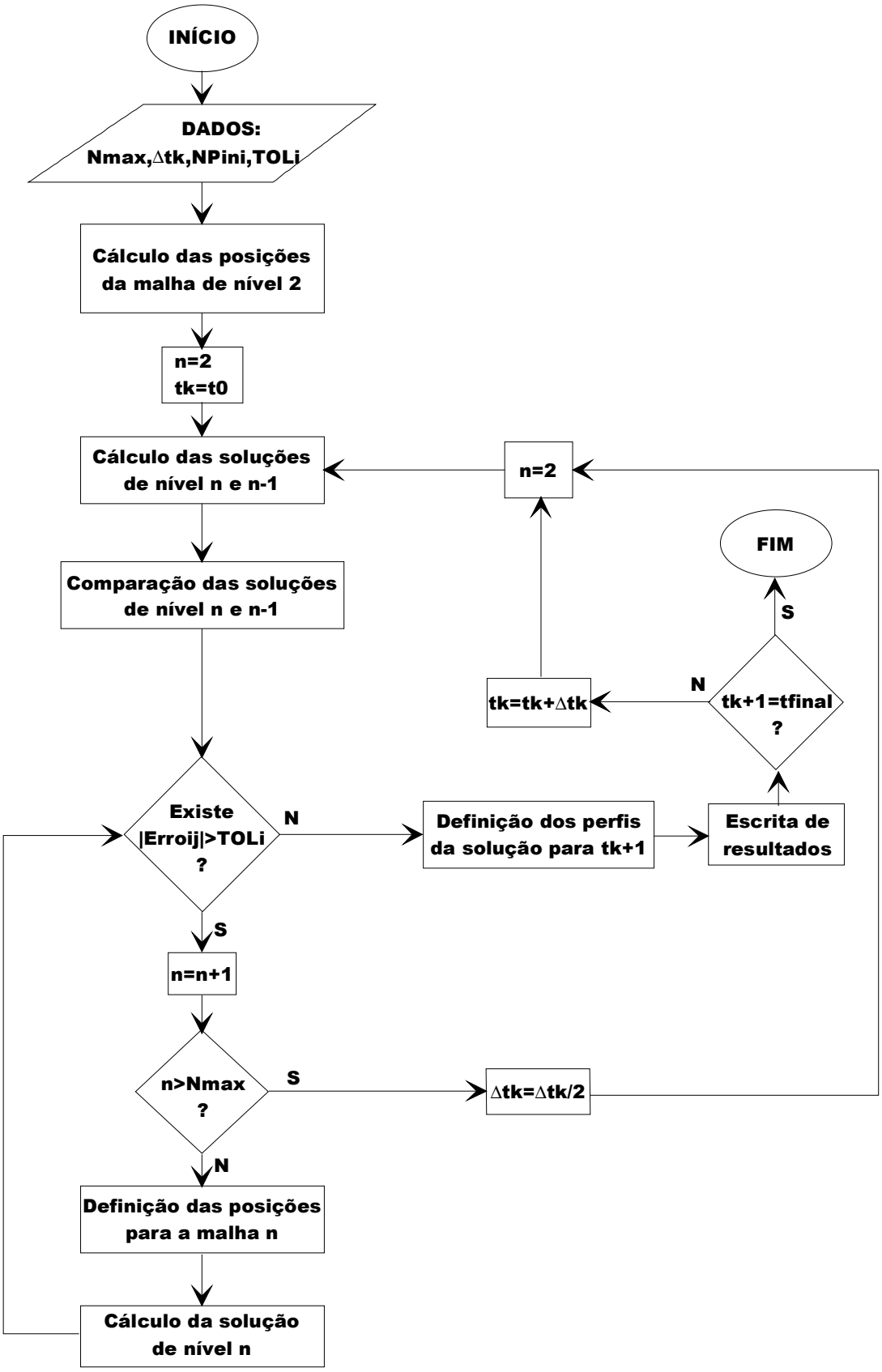

Figura 4.2: Fluxograma referente ao método de refinamento.

4.3 – Método de Malha Móvel Adaptativa

4.3.1 – Equação de Movimentação Nodal Implícita

O algoritmo descrito em seguida, utiliza um esquema semelhante ao método anterior, para a identificação das regiões mais problemáticas do domínio, do ponto de vista da maior dificuldade de integração temporal do modelo estudado, geralmente provocada pela introdução de difusão numérica na avaliação dos perfis.

No entanto, a diferença deste método consiste no facto de não recorrer a uma estratégia de refinamento para a integração dos subproblemas formados para cada subdomínio. De facto, esta estratégia é substituída pela introdução de uma equação de movimentação nodal dinâmica na integração dos subdomínios que, como anteriormente, são formados pela comparação entre as soluções obtidas uma malha de grau 1 (malha larga) e uma de grau 2 (malha fina), definidas da mesma forma como no método anterior.

Este algoritmo é baseado na aplicação de equações deduzidas por *Petzold*[91], que consistem numa generalização do esquema desenvolvido por *Hyman*[68] para modelos explícitos da forma: $u_t = f(u, u_z, u_{zz})$, para modelos que não dependam explícitamente de f, ou seja, equações diferenciais implícitas da forma:

$$F(u_t, u, u_z, u_{zz}) = 0 \tag{4.8}$$

Considerando $u_{\tau j}$ como a derivada da solução em relação ao tempo num nodo pertencente a uma malha fixa z_j, a transformação das variáveis para o sistema móvel é dada por:

$$\dot{u} = u_\tau + u_z \cdot \dot{z} \tag{4.9}$$

Efectuando-se a minimização da taxa de variação no tempo de u e z:

$$\min_{\dot{z}} \left[\|\dot{u}\|_2^2 + \alpha \cdot \|\dot{z}\|_2^2 \right] = \min_{\dot{z}} \left[\Sigma \dot{u}^2 + \alpha \cdot \dot{z}^2 \right] = \min_{\dot{z}} \left[(u_\tau + u_z \cdot \dot{z})^2 + \alpha \cdot \dot{z}^2 \right] \tag{4.10}$$

obtém-se, por resolução da equação quadrática em $\dot{z}$, o mínimo para cada ponto da malha, pela expressão:

$$(u_\tau + u_z \cdot \dot{z}) \bullet u_z + \alpha \cdot \dot{z} = 0 \tag{4.11}$$

que através de (4.9) pode ser escrita da forma:

$$\alpha \cdot \dot{z} + \dot{u} \bullet u_z = 0 \tag{4.12}$$

A equação (4.12) consiste na equação diferencial de movimentação nodal, na sua forma implícita.

4.3.2 – Interpretação Geométrica da Equação de Movimentação Nodal

Para uma melhor compreensão do significado da aplicação da equação (4.12), procede-se à análise da sua dedução geométrica para uma P.D.E escalar. Assim, considera-se a frente móvel ilustrada na Figura 4.3 para o tempo t_n.

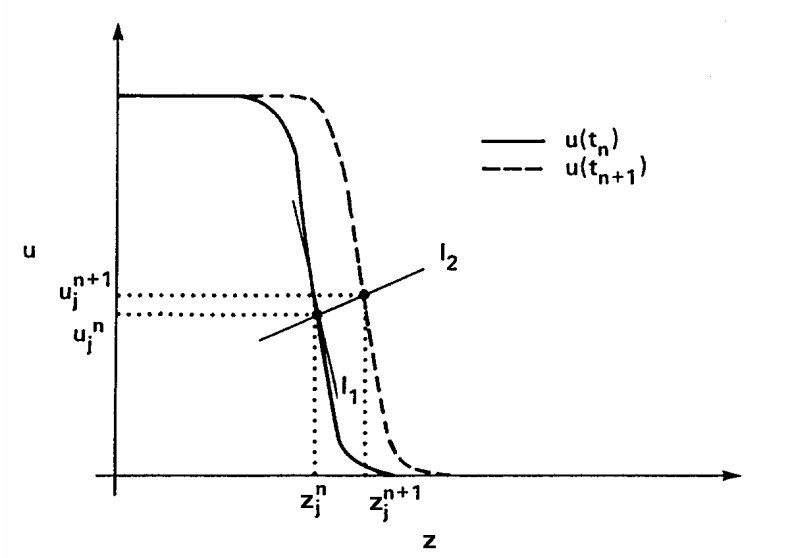

Figura 4.3: Interpretação geométrica das equações de movimentação nodal.[91]

A recta tangente ℓ_1 à curva da solução que passa no ponto (z_j^n,u_j^n), correspondente ao nodo da grelha z_j, no tempo t_n, é calculada por:

$$u - u_j^n = u_z \cdot \left(z - z_j^n\right) \tag{4.13}$$

Por outro lado, a linha ℓ_2 que passa em (z_j^n,u_j^n) e é ortogonal a ℓ_1 é dada por:

$$u - u_j^n = \left(z - z_j^n\right) \Big/ u_z \tag{4.14}$$

Supondo que (z_j^{n+1},u_j^{n+1}) é o ponto em ℓ_2 que intersecta a solução no tempo t_{n+1}, então, (z_j^{n+1},u_j^{n+1}) satisfaz a equação:

$$u_j^{n+1} - u_j^n = -\left(z_j^{n+1} - z_j^n\right) \Big/ u_z \tag{4.15}$$

No limite t_{n+1}- $t_n \rightarrow 0$, a equação anterior toma a forma:

$$\dot{u} = -\dot{z} \Big/ u_z \qquad \Leftrightarrow \qquad \dot{z} + u_z \bullet \dot{u} = 0 \tag{4.16}$$

que corresponde à equação (4.12) (para α=1) deduzida na secção anterior.

Deste modo, verifica-se que no limite t_{n+1}- $t_n \rightarrow 0$, e para uma única P.D.E., a equação (4.12) movimenta cada ponto da grelha ao longo do segmento de recta ortogonal à solução em (z_j^n,u_j^n), até que este intersecte a solução no tempo seguinte n+1.

Esta estratégia garante a minimização da variação de u e z, mas não conduz necessariamente ao maior passo temporal possível, já que não impede de forma alguma o cruzamento nodal.

Para tal, considere-se uma frente móvel extremamente abrupta do tipo da esquematizada na Figura 4.4. O ponto da malha em j move-se a uma velocidade aproximadamente igual à velocidade da frente, enquanto que o ponto da malha j+1 não se movimenta, visto que na sua posição a solução apresenta gradiente nulo. Assim, os passos temporais têm de ser bastante reduzidos para impedir a colisão entre os pontos da malha.

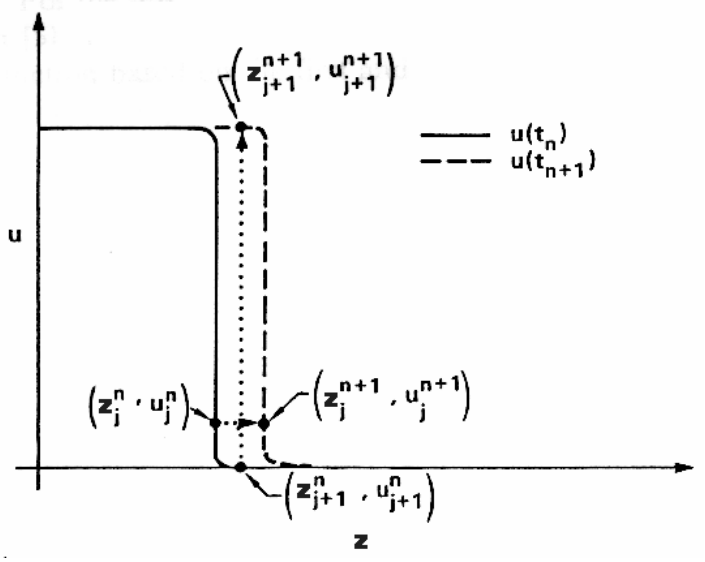

Figura 4.4: Ilustração do problema dos cruzamentos nodais.[91]

4.3.3 – Equações de Movimentação Nodal de Aplicação Mais Geral

Este problema pode ser suavizado pela adição de uma função de penalidade à expressão de minimização considerada anteriormente (vd. Equação (4.10)), que possibilita que nodos vizinhos se movimentem com velocidades semelhantes:

$$\min_{\dot{z}_j}\left[\left\|\dot{u}_j\right\|_2^2+\alpha\cdot\left\|\dot{z}_j\right\|_2^2+\lambda\cdot\left(\left\|\frac{\dot{z}_j-\dot{z}_{j-1}}{z_j-z_{j-1}}\right\|_2^2+\left\|\frac{\dot{z}_{j+1}-\dot{z}_j}{z_{j+1}-z_j}\right\|_2^2\right)\right] \tag{4.17}$$

Desta forma, deduz-se uma equação de movimentação nodal dinâmica, que pode ser apresentada da seguinte forma:

$$\alpha\cdot\dot{z}_j+\dot{u}_j\bullet u_{zj}+\lambda\cdot\left(\frac{\dot{z}_j-\dot{z}_{j-1}}{\left(z_j-z_{j-1}\right)^2}-\frac{\dot{z}_{j+1}-\dot{z}_j}{\left(z_{j+1}-z_j\right)^2}\right)=0 \tag{4.18}$$

O efeito da introdução do termo de penalidade assemelha-se à de um factor extra de difusão ou suavização das velocidades nodais na malha. No entanto, apesar da introdução do termo adicional se revelar bastante eficaz na eliminação da maioria dos cruzamentos de nodos, não garante a sua erradicação total.

De forma a ultrapassar problemas de escala entre variáveis e tornar a equação de movimentação nodal o mais invariante possível em relação às diferentes ordens de grandeza e

translações em z e u, procede-se à minimização de uma norma ℓ_2 pesada, para a dedução da malha móvel:

$$\left\|\bar{u}\right\|_2^2 = \sum_{i=1}^{N+1}\left(\bar{u}_i \Big/ w_i\right)^2 \tag{4.19}$$

onde: $\bar{u} = \left(u^T, z\right)^T$

N – número de P.D.E.´s do modelo original.

A equação de movimentação nodal em z_j, assim obtida, é a seguinte:

$$\sum_{i=1}^{N} \frac{\dot{u}_{i,j} \cdot u_{zi,j}}{w_i^2} + \alpha \cdot \frac{\dot{z}_j}{w_{N+1}^2} + \lambda \cdot \left(\frac{\dot{z}_j - \dot{z}_{j-1}}{\left(z_j - z_{j-1}\right)^2} - \frac{\dot{z}_{j+1} - \dot{z}_j}{\left(z_{j+1} - z_j\right)^2} \right) = 0 \tag{4.20}$$

sendo os pesos w_i escolhidos através da expressão:

$$w_i = \max\left(\left|\bar{u}_{i,max} - \bar{u}_{i,min}\right|, \text{floor}(i)\right), \qquad i = 1, 2, \quad \cdots \quad , N+1 \tag{4.21}$$

onde: $\bar{u}_{i,max}$ e $\bar{u}_{i,min}$ - correspondem aos componentes i da matriz $\bar{u}$, máximos e mínimos respectivamente, ao longo da totalidade da malha.

A escolha dos valores para os parâmetros floor(i) é bastante importante, especialmente para componentes da solução que desenvolvam gradientes elevados após um arranque suave.

4.3.4 – Descrição do Algoritmo

Deste modo, o algoritmo desenvolvido resume-se a uma conjugação entre o procedimento de estimativa do erro espacial, em cada ponto da malha de nível 1 (da mesma forma como no algoritmo de refinamento), com a introdução de uma equação adaptativa dinâmica na integração dos subproblemas correspondentes a cada subdomínio de nível 2 seleccionado. Desta forma, o método pode ser dividido em dois estágios:

ESTÁGIO I ⇨ Estimativa do erro espacial e identificação dos subdomínios.
ESTÁGIO II ⇨ Integração de cada subproblema através da introdução de uma malha móvel em cada subdomínio.

No estágio II, é possível aplicar qualquer uma das equações de movimentação nodal (equações (4.12), (4.18) ou (4.20)) apresentadas anteriormente.

Por outro lado, o passo temporal é ajustado de forma a impedir qualquer cruzamento nodal. Além disso, é efectuada uma redefinição da malha de nível 2 base, após cada integração, de forma a impedir a aproximação ou o afastamento excessivo de nodos adjacentes, o que, em alguns casos, pode provocar dificuldades acrescidas no avanço temporal da integração. Deste modo, procede-se a um afastamento dos nodos demasiado próximos ($\Delta z < \Delta z_{MIN}$) e a uma equidistribuição de dois nodos adicionais entre nodos demasiado afastados ($\Delta z > \Delta z_{MAX}$).

O algoritmo descrito anteriormente é resumido no fluxograma da Figura 4.5:

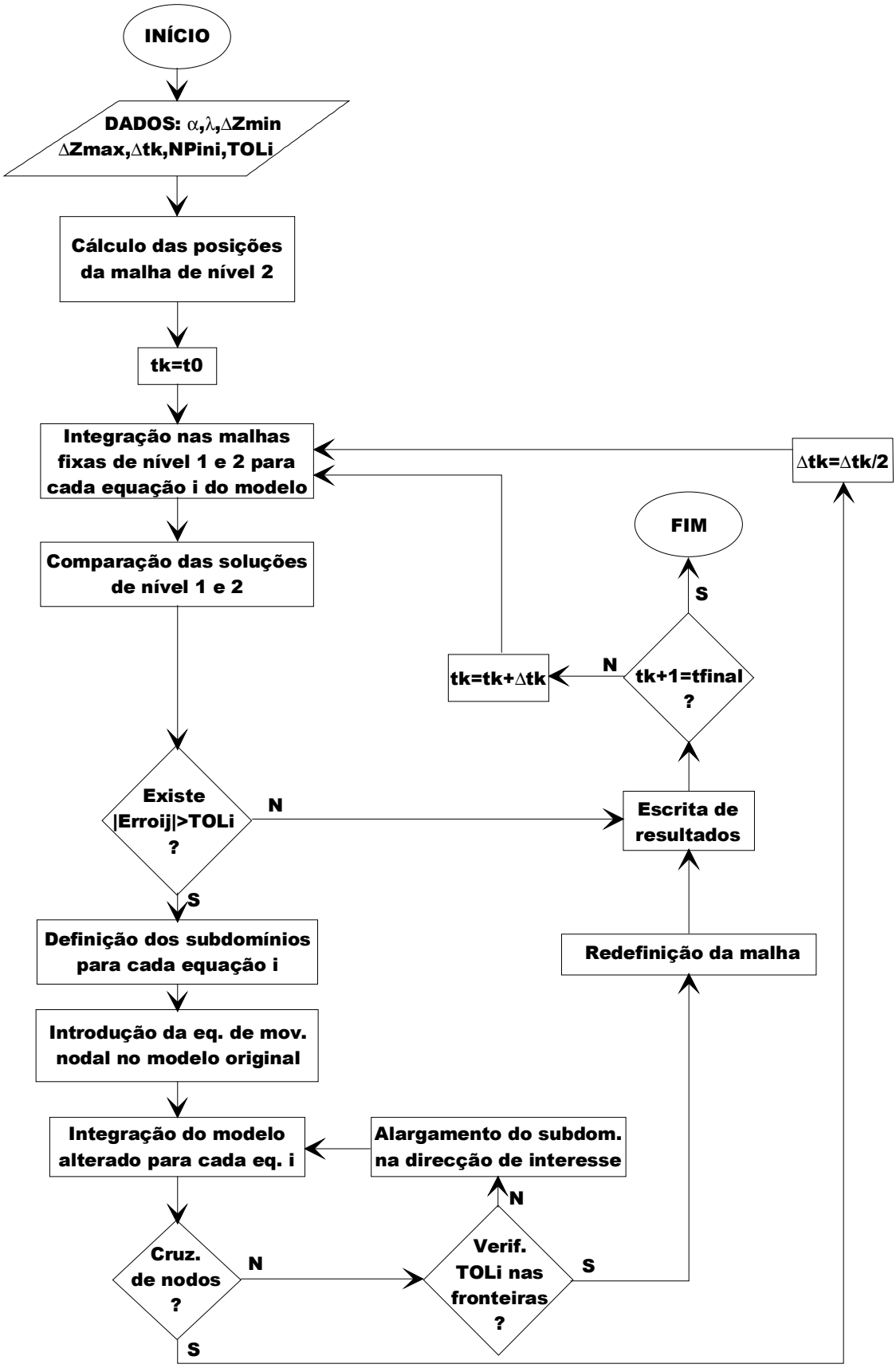

Figura 4.5: Fluxograma referente ao método de malha móvel adaptativa.

4.4 – Tratamento dado às Fronteiras nos Subproblemas Gerados

Das várias hipóteses que se poderiam considerar no que diz respeito ao tratamento a dar aos pontos fronteira interiores durante a integração de cada subproblema, seleccionaram-se dois tipos de estratégias:

① ***Subproblemas integrados com condições fronteira de Dirichlet:***

As derivadas espaciais nas fronteiras são avaliadas de modo semelhante ao problema geral no que diz respeito à selecção dos pontos que afectam as fórmulas de discretização. Considera-se que cada ponto fronteira situado no interior do domínio global seja afectado por condições fronteira de *Dirichlet*, ou seja, na integração da grelha de nível n+1, mantêm-se fixos os valores da solução calculados no tempo final correspondente a esse passo, através da utilização da malha de nível n. Há que ter ainda em conta, que as posições da fronteira coincidem com os primeiros pontos da grelha de nível n-1 que verificam a tolerância pré-definida.

Apesar deste procedimento garantir a inexistência de descontinuidades no perfil de solução final, originados pela junção de secções da solução calculados a partir de malhas de nível diferente, não assegura completamente a continuidade da sua derivada. De facto, é possível gerarem-se gradientes diferentes de cada lado dos pontos de junção entre zonas com níveis de refinamento distintos.

No entanto, este procedimento é aplicado na formulação do método de refinamento, não originando problemas de maior na definição correcta dos perfis da solução, para a maioria dos casos testados.

② ***Condições fronteira móveis – utilizam-se pontos exteriores aos subdomínios para a avaliação das derivadas espaciais:***

Quando possível, a estimativa das derivadas espaciais nos pontos fronteira é efectuada com a utilização dos pontos adjacentes e exteriores aos subdomínios. Estes pontos são utilizados no cálculo das derivadas espaciais, mas não estão incluídos na integração. O número de pontos extra utilizados de cada lado do subdomínio depende do tipo de diferenças finitas aplicado ao problema e do posicionamento relativo deste em relação ao domínio global. Como o integrador subdivide o intervalo de tempo inicial em vários passos intermédios, necessita do valor da solução nesses instantes. Assim, nestes pontos, a solução é calculada por interpolação linear entre o valor inicial para a abcissa pretendida em t_k, e a calculada anteriormente para t_{k+1} ou num número pré-definido de pontos temporais intermédios, igualmente espaçados. Desta forma, este procedimento implica a introdução de interpolações adicionais e, consequentemente, de mais imprecisões no avanço da integração, o que se constitui como a sua maior desvantagem.

É de referir, igualmente, que as soluções nos pontos fronteira dos subproblemas interiores têm obrigatoriamente de verificar a tolerância pré-estabelecida, em relação às soluções obtidas com a utilização da grelha fixa nesses pontos. Caso tal não aconteça, procede-se ao alargamento do subdomínio de interesse na direcção respectiva, repetindo-se o procedimento até que a tolerância seja verificada em ambas as extremidades. Deste modo, o processo de integração dos subdomínios transforma-se num esquema iterativo, mas

absolutamente indespensável, de forma a se evitar descontinuidades significativas no perfil da solução final. Deste modo, este procedimento assegura uma melhor continuidade da solução entre zonas calculadas de forma diferente.

Esta estratégia é aplicada conjuntamente com o algoritmo de malha móvel adaptativa. De facto, verifica-se que a estratégia ① provoca problemas de performance do integrador a quando da introdução da malha móvel no modelo original. Este comportamento seria de esperar, dado que este procedimento origina variações muito bruscas nas soluções fronteira que podem acrescentar problemas ao desempenho do integrador. Por outro lado, verifica-se que a aplicação de uma estratégia gradual como a ②, não provoca dificuldades de maior, obtendo-se resultados satisfatórios. No entanto, estas dificuldades acrescidas deveriam ser sentidas, já que a introdução das equações diferenciais da malha aumenta significativamente a não-linearidade dos modelos.

5. Comparação do Desempenho dos Métodos Numéricos

Neste capítulo apresenta-se a comparação entre o desempenho dos métodos numéricos descritos no capítulo anterior e o **Método de Diferenças Finitas** fixas, quando aplicados a uma equação não-linear, cuja solução desenvolva choques e/ou frentes abruptas móveis. Deste modo, pretende-se não só comparar a qualidade das soluções obtidas através da aplicação dos métodos referidos a um modelo base, como, igualmente, analisar o esforço computacional exigido por cada algoritmo em relação ao **Método de Elementos Finitos Móveis** (M.E.F.M.), cujo código foi desenvolvido e aplicado por *Duarte*[29]. Para exemplo base foi escolhida a equação víscida de *Burgers* que é enunciada por:

Exemplo 1: Equação Víscida de Burgers

$$\frac{\delta u}{\delta t} = -u \cdot \frac{\delta u}{\delta z} + De \cdot \frac{\delta^2 u}{\delta z^2} \qquad (5.1)$$

com as condições fronteira:

$$u(0,t) = 0 \qquad (5.2)$$

$$u(1,t) = 0 \qquad (5.3)$$

e a condição inicial:

$$u(z,0) = \tfrac{1}{2} \cdot \sin(\pi \cdot z) + \sin(2 \cdot \pi \cdot z) \qquad (5.4)$$

Este modelo descreve a difusão de uma substância, num meio isotrópico, sujeita a fenómenos convectivos/difusivos, sendo:

u – concentração da substância difusora;
De – coeficiente de difusão.

O cálculo da solução deste modelo é problemático e difícil de obter através da utilização de um método de não-adaptativo, devendo-se tal facto, essencialmente ao tipo de condição inicial considerado.

O comportamento da solução pode ser resumido da seguinte forma: partindo da onda sinusoidal inicial, cada uma das secções desta deslocam-se em sentidos opostos, originando uma frente abrupta em $z \cong 0.6$ para $t \cong 0.2$, cuja espessura é proporcional ao valor do coeficiente de difusão De; a partir deste instante, a frente desloca-se para a direita (na direcção positiva de z), ao mesmo tempo que a sua amplitude se reduz; por volta de $t \cong 1.4$, e já depois da secção negativa da frente ter desaparecido, esta embate na fronteira direita (z=1), mantendo-se nessa posição até desaparecer. Deste modo, a maior dificuldade na integração deste modelo consiste na definição correcta da frente formada pela solução, e do seu movimento. Neste caso, esta terá uma espessura da ordem de $De=1\times10^{-3}$, exigindo desse modo, que o espaçamento entre os nodos nas regiões onde a frente se formará e se deslocará, não seja muito superior a esse valor. Assim, se se pretender integrar este modelo numa malha fixa, esta terá de ser necessariamente bastante fina, provocando um esforço computacional excessivo na integração do modelo.

5.1 – Método das Diferenças Finitas

Considera-se a P.D.E. geral do tipo,

$$u_t = L(u, u_z, u_{zz}, z, t) \tag{5.5}$$

sujeita às condições fronteira:

$$u(z^L, t) = u^L \tag{5.6}$$

$$u(z^R, t) = u^R \tag{5.7}$$

e à condição inicial:

$$u(z,0) = u^0(z) \quad ; \quad z \in [z^L, z^R] \tag{5.8}$$

Os métodos de integração descritos anteriormente baseiam-se no **Método das Linhas**, ou seja, procede-se a uma discretização da direcção espacial, sendo as respectivas derivadas estimadas por aproximações algébricas (vd. Apêndice C). Desta forma, obtém-se um sistema de O.D.E.´s que é resolvido por intermédio de um integrador numérico (neste caso, o integrador implícito DASSL).

Na resolução deste caso específico, optou-se por recorrer a aproximações de diferenças finitas de quarta ordem centradas, utilizando-se o método recursivo desenvolvido por *Fornberg*[45,46] (vd. Apêndice C) para a avaliação do coeficientes correspondentes a uma grelha uniformemente espaçada. Deste modo, a matriz de diferenciação para derivadas de primeira ordem tem a forma:

$$\underline{u}_z = \frac{1}{24 \cdot \Delta z} \cdot \begin{bmatrix} -50 & 96 & -72 & 32 & -6 \\ -6 & -20 & 36 & -12 & 2 \\ 2 & -16 & 0 & 16 & -2 \\ -2 & 12 & -36 & 20 & 6 \\ 6 & -32 & 72 & -96 & 50 \end{bmatrix} \cdot \underline{u} \tag{5.9}$$

As aproximações para a segunda derivada podem ser calculadas a partir dos valores da primeira derivada (por aplicação da equação (5.9)), ou directamente através dos valores da solução u. Neste caso, existem diversas formas de definição da matriz de diferenciação, dependentes do tipo de condições fronteira considerado. No entanto, para o caso de condições fronteira de *Dirichlet*, deduz-se a seguinte matriz de diferenciação geral para diferenças finitas centradas de quarta ordem:

$$\underline{u}_{zz} = \frac{1}{12 \cdot \Delta z^2} \cdot \begin{bmatrix} 35 & -104 & 114 & -56 & 11 \\ 11 & -20 & 6 & 4 & -1 \\ -1 & 16 & -30 & 16 & -1 \\ -1 & 4 & 6 & -20 & 11 \\ 11 & -56 & 114 & -104 & 35 \end{bmatrix} \cdot \underline{u} \tag{5.10}$$

Nas expressões (5.9) e (5.10), as primeira, segunda, quarta e quinta linhas referem-se a coeficientes utilizados na estimativa das derivadas nos pontos “especiais”, situados junto às

fronteiras, onde a expressão de discretização central com cinco pontos não pode ser utilizada sem se recorrer as fronteiras fictícias. Por outro lado, a linha central refere-se aos pontos genéricos situados no interior do domínio onde a fórmula de discretização escolhida pode ser aplicada.

É possível desenvolver fórmulas de aproximação semelhantes de ordem diferente (maior ou menor) ou dando mais peso a pontos situados a montante (diferenças *Upwind*) ou a jusante (diferenças *Downwind*) daquele onde se pretende estimar a derivada (vd. Apêndice C).

A discretização espacial da equação de *Burgers*, usando diferenças finitas de quarta ordem centradas numa grelha fixa uniforme de N pontos, conduz ao sistema de O.D.E.´s seguinte:

Ponto i genérico:

$$\dot{u}_i = -u_i \cdot \frac{2\cdot u_{i-2} - 16\cdot u_{i-1} + 16\cdot u_{i+1} - 2\cdot u_{i+2}}{24\cdot \Delta z} + De \cdot \frac{-u_{i-2} + 16\cdot u_{i-1} - 30\cdot u_i + 16\cdot u_{i+1} - u_{i+2}}{12\cdot \Delta z^2}$$

$$i = 3, \ldots., N-2 \qquad (5.11)$$

Ponto fronteira:

$$\text{Condições de } \textit{Dirichlet} \quad \Rightarrow \quad \dot{u}_1 = \dot{u}_N = 0 \qquad (5.12)$$

Pontos "especiais":

$$\dot{u}_2 = -u_2 \cdot \frac{-6\cdot u_1 - 20\cdot u_2 + 36\cdot u_3 - 12\cdot u_4 + 2\cdot u_5}{24\cdot \Delta z} + De \cdot \frac{11\cdot u_1 - 20\cdot u_2 + 6\cdot u_3 + 4\cdot u_4 - u_5}{12\cdot \Delta z^2} \qquad (5.13)$$

$$\dot{u}_{N-1} = -u_{N-1} \cdot \frac{-2\cdot u_{N-4} + 12\cdot u_{N-3} - 36\cdot u_{N-2} + 20\cdot u_{N-1} + 6\cdot u_N}{24\cdot \Delta z} + De \cdot \frac{-u_{N-4} + 4\cdot u_{N-3} + 6\cdot u_{N-2} - 20\cdot u_{N-1} + 11\cdot u_N}{12\cdot \Delta z^2} \qquad (5.14)$$

Este sistema pode ser resumido na sua forma matricial:

$$\underline{\underline{A}} \cdot \dot{\underline{y}} = \underline{g}\left(\underline{y}\right) \qquad (5.15)$$

em que $\underline{y} = [u_1, \cdots, u_N]$ e $\underline{\underline{A}}$ é uma matriz diagonal unitária.

No Gráfico 5.1 são apresentados os perfis da solução obtidos com uma grelha de 41 nodos (Δz=0.025). Verifica-se que os resultados são afectados por forte dispersão numérica na região onde se deveria formar a frente. Desta forma, torna-se impossível prosseguir a integração do modelo a partir de t=0.2, devido à instabilidade do sistema de O.D.E.´s (5.15).

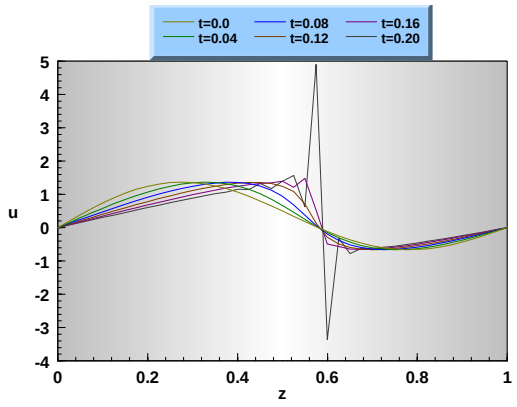

Gráfico 5.1: Resultados obtidos com Diferenças Finitas Centradas de 4ª ordem (tol INT=1×10^{-5}, Δz=0.025).

Os resultados obtidos através de uma discretização de quarta ordem *Biased Upwind* (utilização de três pontos à esquerda e um ponto à direita do nodo de interesse) numa grelha fixa e uniforme de 41 nodos são apresentados no Gráfico 5.2. Neste caso, observa-se novamente instabilidade na solução a partir do tempo t=0.15, perfeitamente verificada pela ocorrência de fortes oscilações nos perfis. As oscilações (cuja amplitude é superior às observadas no caso anterior) são predominantemente negativas e ocorrem na secção do domínio situada após a frente (ao contrário do que sucedia no caso anterior). Por outro lado, verifica-se que, enquanto foi possível efectuar o avanço temporal da integração (até t=0.19) a formação da zona superior da frente ocorre de uma forma relativamente normal. Assim, é possível concluir-se que este esquema de diferenças se revela mais adequado que o anterior para a zona do domínio anterior à frente, já que aí a velocidade da onda é positiva. Do mesmo modo, verifica-se que este esquema é absolutamente inadequado para a região negativa dos perfis, onde uma discretização de diferenças finitas adiantadas seria mais indicada.

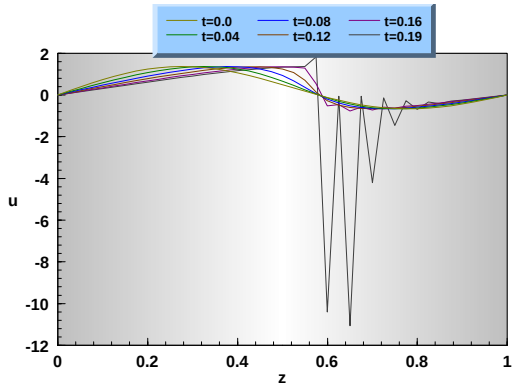

Gráfico 5.2: Resultados obtidos com Diferenças Finitas *Biased Upwind* de 4ª ordem (tol INT=1×10^{-5}, Δz=0.025).

Por análise dos resultados obtidos, é possível concluir que o **Método de Diferenças Finitas** fixas não conduz a resultados aceitáveis, principalmente quando aplicado a modelos caracterizados por termos advectivos não-lineares, associados a termos difusivos de influência reduzida. Por outro lado, se se partir de soluções iniciais caracterizadas por advecção, este tipo de equações desenvolve frentes abruptas de espessura múltipla do factor de difusão (De) que somente seriam resolvidas satisfatoriamente por este tipo de estratégia se se utilizassem grelhas de espaçamento muito reduzido, o que implicaria esforços computacionais incomportáveis. O aparecimento de instabilidade e consequente dificuldade na integração é

resultado dos valores muito elevados da segunda derivada originados pelo aparecimento da frente.

De forma a ultrapassar estas dificuldades e assegurar que a grelha apresente maiores concentrações nas regiões do domínio mais críticas, onde ocorrem maiores problemas na integração, e vice-versa (o que consiste no objectivo principal dos métodos adaptativos de integração), podem ser aplicadas duas estratégias básicas:

❶ Refinamento da grelha nas zonas do domínio de maior actividade, correspondente a gradientes da solução mais elevados;

❷ Deslocamento dos nodos para as zonas do domínio de maior actividade.

5.2 – Método de Refinamento de Grelha

O presente subcapítulo consiste na apresentação e análise dos resultados obtidos pela aplicação do **Método de Refinamento de Grelha** descrito no Capítulo ao exemplo 1 (Equação Víscida de *Burgers*). Assim, pretende-se analisar a performance deste método através da precisão dos resultados obtidos, e do esforço computacional exigido em relação ao M.E.F.M..

Logicamente que o desempenho do referido método está dependente dos valores assumidos pelo parâmetros de entrada deste que são:

➔ Tolerância (Absoluta ou Relativa);

➔ Número de pontos da grelha base de nível 1.

Por outro lado, é necessário igualmente definir: o tipo de interpolação a utilizar para estimar os valores intermédios da solução para as grelhas de nível superior a dois, que constituem parte dos valores de arranque para a integração de cada subdomínio; o tipo de grelha inicial - uniforme ou não-uniforme, e consequentemente as posições relativas dos nodos; o tipo de diferenças finitas para a discretização espacial.

No caso do modelo estudado, optou-se por utilizar diferenças finitas do tipo centrado que se verificou constituírem um melhor compromisso para uma descrição mais exacta do movimento antagónico das duas ondas. Devido as características do comportamento inicial da solução não se torna necessário concentrar nodos em qualquer zona específica do domínio, tendo-se, portanto, utilizado grelhas base uniformes. Deste modo, apenas se considerou importante variar a concentração nodal da grelha base (NP) e o grau de precisão exigida (TOL).

Para tal, desenvolveu-se um código em linguagem FORTRAN 77 (vd. Apêndice D) que executa o algoritmo apresentado, usando o integrador implícito DASSL para o avanço temporal da solução. Deste modo, o código foi testado para diversas situações, tendo-se analisado a sua performance em cada caso. As condições de cada *run* efectuado são resumidas no Tabela seguinte:

Tabela 5.1: Condições fixadas para a execução de cada *run* (M. Ref.).

Run	***Tol***	***Tipo de Interpolação***	***NP***	***tcpu (s) (Tfinal = 1)***
1	1×10^{-3}	Linear	41	5496.4
2	5×10^{-3}	Linear	41	2096.2
3	1×10^{-2}	Linear	41	1752.4
4	5×10^{-3}	Splines 5 P.	41	3599.0
5	5×10^{-3}	Linear	21	852.3

Todos os *runs* foram executados com grelhas equidistribuídas de NP pontos e discretizações espaciais centradas de cinco pontos (4ª. Ordem), sendo a tolerância do integrador fixada em 1×10^{-5}.

No Gráfico 5.3 apresenta-se a comparação entre os perfis obtidos em todos os *runs* para t=0.2, que corresponde a um período crítico onde a frente já se encontra perfeitamente formada. Em nenhum dos casos se observa oscilação significativa, verificando-se em qualquer deles a formação correcta da frente. No entanto, para o *run5* nota-se uma ligeira discrepância na curva correspondente a um decaimento excessivo da onda positiva.

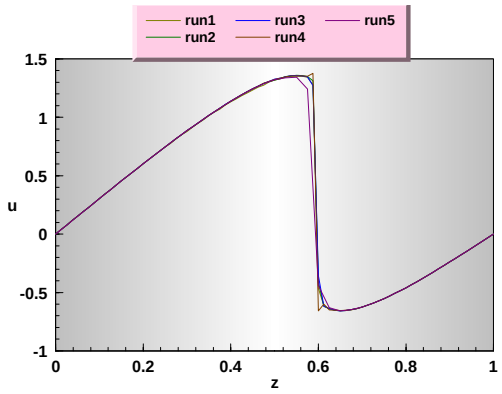

Gráfico 5.3: Resultados obtidos pelo método de refinamento em todos os *runs* para t=0.2.

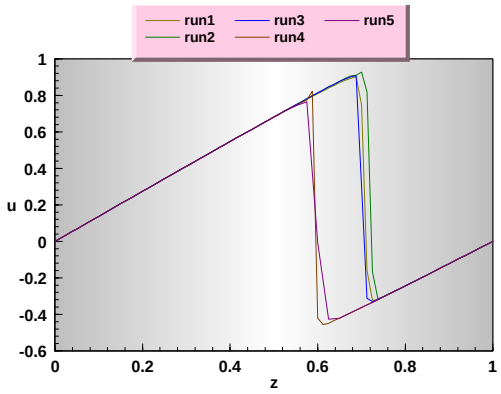

Gráfico 5.4: Resultados obtidos pelo método de refinamento em todos os *runs* para t=0.4.

No caso do *run5*, verifica-se um decaimento da amplitude da onda positiva consideravelmente mais acentuado no tempo t=0.4 (vd. Gráfico 5.4), que também ocorre no perfil referente ao *run4*. Este decaimento acelerado é acompanhado por uma estagnação das frentes na posição inicial.

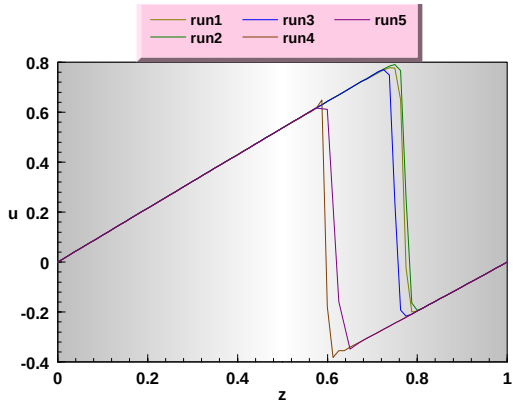

Gráfico 5.5: Resultados obtidos pelo método de refinamento em todos os *runs* para t=0.8.

Em t=0.8 (vd. Gráfico 5.5), a separação entre os perfis obtidos pelos *runs* referidos e os restantes é ainda mais notória, continuando as frentes paralisadas em z ≅ 0.6 e o decaimento demasiado acelerado da onda positiva e excessivamente lento da onda negativa. Curiosamente este comportamento é precisamente o necessário de forma a que a amplitude das ondas evolua de forma equivalente à esperada, ou seja, acompanhando o movimento da frente, mas mantendo-se praticamente fixa na posição inicial. De facto, se se partir dos perfis bastante aceitáveis obtidos nos restantes *runs* e se fixar uma frente imaginária na abcissa referida, ao retirar-se a porção positiva da onda, prolongando-se a secção negativa obtém-se um perfil muito semelhante aos obtidos para os *run4* e *run5*. De qualquer modo, verifica-se que nestas condições, o método não conduz a resultados satisfatórios.

Portanto, pode-se concluir que a utilização de interpolações por intermédio de *Splines* cúbicas (*run4)* não é adequado às características das curvas da solução. Estas desenvolvem declives bastante abruptos entre zonas de gradientes muito suaves, sendo portanto, melhor descritas por interpolações lineares. Assim, verifica-se que a interpolação por *Splines* introduz demasiados erros na aproximação dos valores intermédios, principalmente na zona da frente, conduzindo às deficiências observadas pela solução do *run4*. Verifica-se, igualmente ligeiras oscilações no perfil correspondente a t=0.2 (vd. Gráfico 5.3) que, no entanto são posteriormente esbatidas.

Por outro lado, é possível concluir que a grelha inicial utilizada em *run5* é demasiado larga, sendo constituída por NP=21. Neste caso, a acumulação de erros, principalmente de interpolação, revela-se mais importante, afectando negativamente o comportamento do método e levando a resultados insatisfatórios.

Verifica-se um ligeiro atraso da frente correspondente ao *run3* em relação às restantes. Este comportamento seria de esperar já que este *run* corresponde a uma tolerância (TOL) superior, e consequentemente a um grau de precisão mais baixo em relação aos *run1* e *run2*. No entanto, pode-se observar igualmente que não existe melhoria significativa na qualidade dos resultados com a redução de TOL entre *run2* e *run1*. De facto, os resultados obtidos para cada um desses *runs* até t=10, são bastante semelhantes (vd. Gráficos 5.6 a 5.9), mas o tempo de cpu sobe significativamente para o caso do *run1*: $tcpu_{run1}$ = 9954.2 s e $tcpu_{run2}$ = 6157.5 s.

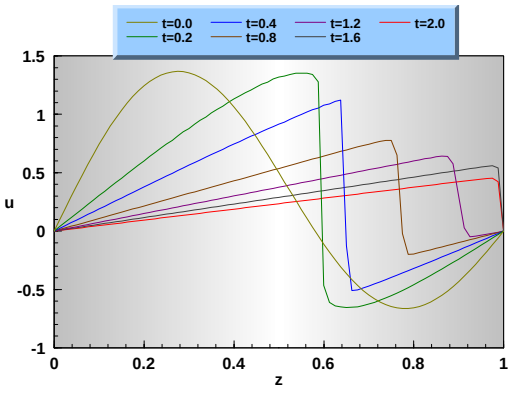

Gráfico 5.6: Resultados obtidos pelo método de refinamento para o *run1* (até t=2).

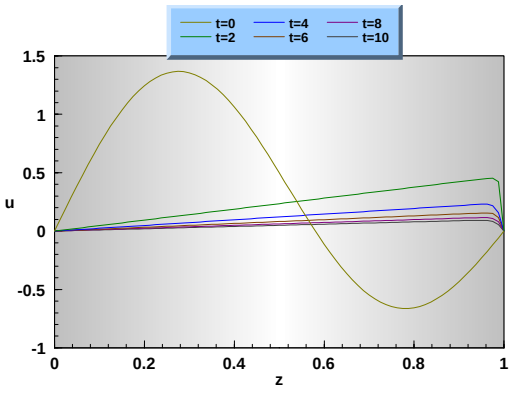

Gráfico 5.7: Resultados obtidos pelo método de refinamento para o *run1* (até t=10).

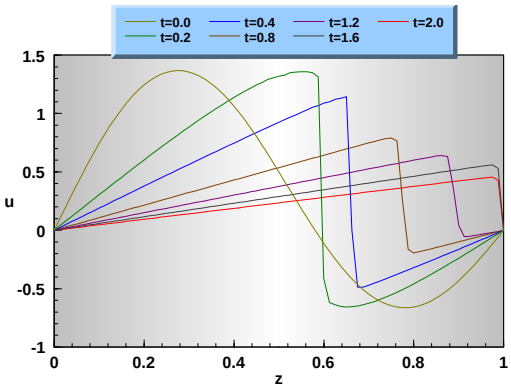

Gráfico 5.8: Resultados obtidos pelo método de refinamento para o *run2* (até t=2).

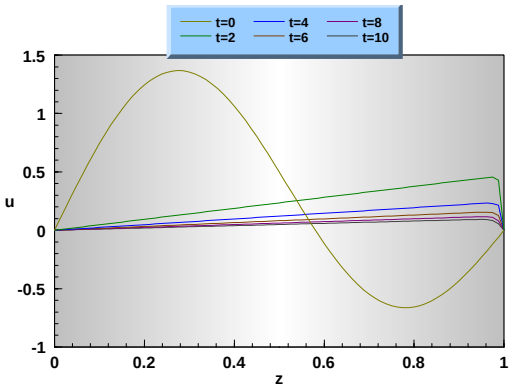

Gráfico 5.9: Resultados obtidos pelo método de refinamento para o *run2* (até t=10).

Deste modo, conclui-se que o esforço computacional exigido pelo caso 2 é inferior ao do caso 1, ou seja, não necessita de recorrer a graus de refinamento tão elevados em zona significativas do domínio para a obtenção de perfis semelhantes (vd. Gráficos 5.10 a 5.13). Assim, o aumento do grau de precisão do *run1* não compensa o trabalho computacional adicional exigido, já que não corresponde a uma melhoria significativa da qualidade dos resultados, já de si bastante aceitáveis, obtidos pelo *run2*, consistindo, por isso, num esforço computacional desnecessário.

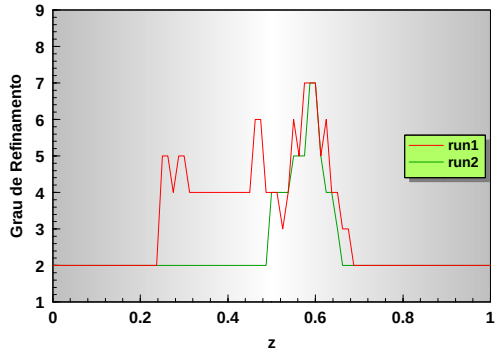

Gráfico 5.10: Grau de refinamento exigido pelo método nas condições do *run1* e *run2* para t=0.2.

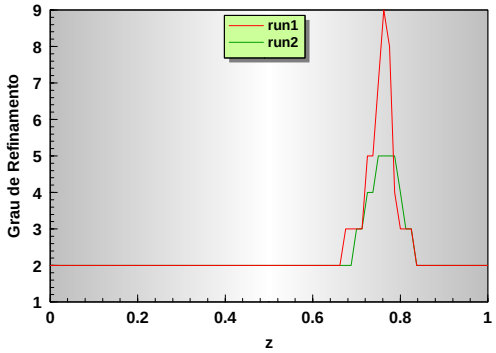

Gráfico 5.11: Grau de refinamento exigido pelo método nas condições do *run1* e *run2* para t=0.8.

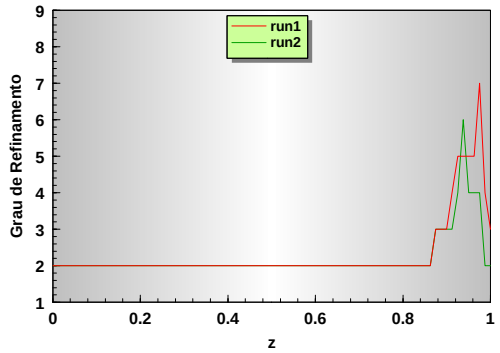

Gráfico 5.12: Grau de refinamento exigido pelo método nas condições do *run1* e *run2* para t=1.4.

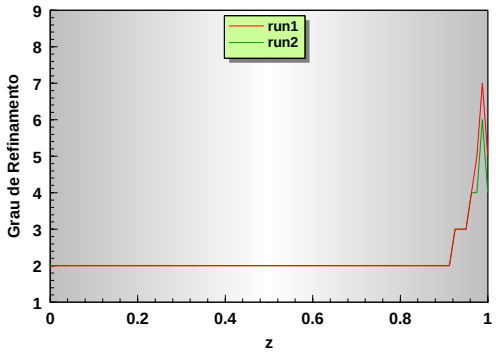

Gráfico 5.13: Grau de refinamento exigido pelo método nas condições do *run1* e *run2* para t=2.0.

Conclui-se, assim, que os melhores resultados obtidos tanto do ponto de vista da qualidade dos perfis calculados, como do esforço computacional exigido, são os referentes ao *run2*. Por outro lado, verifica-se que os maiores problemas que afectam o desempenho do método têm a ver com a introdução de erros de interpolação excessivos que conduzem a perfis

de solução pouco precisos a partir da formação da frente inicial (em t=0.2) e que se degradam progressivamente no tempo.

O tempo de cpu obtido nas condições do *run2*: tcpu= 6157.5 s é consideravelmente mais elevado que o obtido por *Duarte*[29] através da aplicação do M.E.F.M.: t_{MEFM}= 1176.5 s. No entanto, há que considerar que o **Método de Refinamento** é formalmente bastante mais simples do que o M.E.F.M., sendo esta discrepância perfeitamente natural. De qualquer modo, a diferença de performance não é de todo excessiva, sendo o tempo computacional obtido neste caso perfeitamente razoável.

5.3 – Método Dinâmico de Movimentação Nodal

No que diz respeito ao **Método Dinâmico de Movimentação Nodal** foi igualmente desenvolvido um código em FORTRAN 77 (vd. Apêndice D) para o algoritmo proposto no capítulo anterior que também utiliza a subrotina DASSL para a integração temporal.

Neste caso, os parâmetros que é necessário seleccionar para se estudar a eficácia do método na resolução do exemplo considerado são:

- ➔ Tolerância (Absoluta ou Relativa);
- ➔ Número de pontos e o tipo da grelha base inicial;
- ➔ Distâncias máxima e mínima admissíveis entre nodos consecutivos.

A forma como o algoritmo é concebido não só não permite o cruzamento nodal, por ajuste do passo temporal, já que este fenómeno introduz instabilidade na solução, como evita aproximações excessivas entre nodos consecutivos, por definição de um espaçamento mínimo Δz_{MIN}. Tal procedimento é imprescindível porque pontos demasiado próximos podem originar dificuldades de integração. Assim, os nodos em questão são afastados e a solução é avaliada por interpolação. Por outro lado, nodos demasiado afastados podem igualmente provocar problemas de integração em variados casos. Então, quando o intervalo entre dois nodos adjacentes fôr superior a um valor fornecido pelo utilizador Δz_{MAX}, são introduzidos dois pontos nesse intervalo, equidistantes entre si e aos extremos do intervalo, sendo os valores da solução igualmente obtidos por interpolação do perfil original.

Para além disto, é também necessário definir o tipo de discretização espacial a aplicar ao modelo e a equação de movimentação nodal a adicionar ao modelo alterado. No capítulo anterior foram apresentadas três equações distintas. De facto, a primeira não é mais que um caso particular da segunda, considerando um coeficiente de viscosidade (λ) nulo, e a terceira consiste num aperfeiçoamento idealizado para evitar problemas de escala, ou seja, obstar às dificuldades numéricas provocadas por sistemas cujas variáveis tenham valores relativos muito díspares entre si e em relação às correspondentes abcissas. No entanto, neste caso não se põem este tipo de problemas já que os valores de u e de z são aproximadamente da mesma ordem de grandeza. Assim, os parâmetros associados à equação de movimentação nodal são: α e λ (coeficiente de viscosidade nodal).

As condições correspondentes a cada *run* efectuado são apresentadas na Tabela seguinte:

Tabela 5.2: Condições fixadas para a execução de cada *run* (M. M. Móv.).

Run	Δz_{MIN}	***NP***	λ (α=1)	***tcpu (s)*** ***(Tfinal = 10)***
1	5×10^{-3}	12	0	1638.9
2	5×10^{-3}	41	0	5014.9
3	4×10^{-3}	26	0	3867.8
4	4×10^{-3}	21	0	2886.9
5	4×10^{-3}	26	0.1	—

Em cada um dos *runs* foram mantidas as seguintes condições:

⇨ Tipo de discretização - diferenças finitas centradas de 4ª ordem;
⇨ Tolerância absoluta = 1×10^{-3};
⇨ Tolerância do integrador = 1×10^{-5};
⇨ Δz_{MAX} = 0.5 - o que possibilita que em nenhum dos casos seja necessária a introdução de pontos adicionais para a correcta evolução dos perfis;
⇨ Interpolação linear - que se revelou mais adequada para o tipo de perfis desenvolvidos.

No Gráfico 5.14 efectua-se a comparação dos perfis obtidos em todos os *runs* para t=0.2. Verifica-se uma deficiente formação da zona superior da frente para o *run4*. No entanto, as curvas são bastante semelhantes, apesar de se notarem oscilações na formação da zona inferior da frente correspondente ao *run5*, conjuntamente com uma ligeira perturbação na zona superior da respectiva frente.

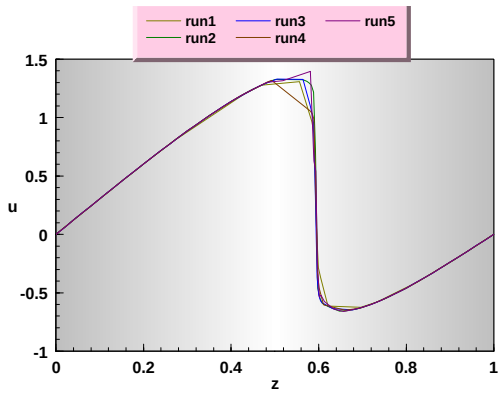

Gráfico 5.14: Resultados obtidos pelo método dinâmico de malha móvel em todos os *runs* para t=0.2.

Analisando a evolução dos perfis obtidos para as condições do *run5* (vd. Gráfico 5.15), verifica-se um agravamento da instabilidade para tempos superiores. A frente estagna inicialmente em z ≅ 0.6 e de seguida torna-se cada vez mais larga, observando-se um afastamento anormal dos pontos que a formam, devido ao recuo da posição da extremidade superior, além de uma aceleração notória do decaimento da onda positiva. Devido à má qualidade dos resultados, a execução do *run5* foi realizada apenas até t=1 (tcpu=1644.3 s). Deste modo, conclui-se que a introdução de um factor de viscosidade (ainda que baixo) na movimentação nodal afecta negativamente a evolução da solução, obtendo-se resultados muito diferentes dos esperados a partir da formação da frente (t=0.2).

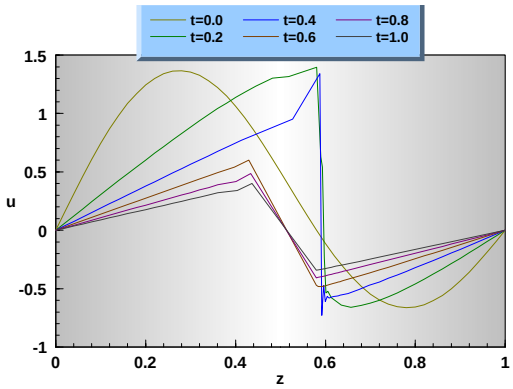

Gráfico 5.15: Resultados obtidos pelo método dinâmico de malha móvel para o *run5* (até t=1).

No Gráfico 5.16, observam-se os perfis obtidos em cada *run* para t=0.8. Para além da discrepância visível da curva correspondente ao *run5*, é notória a influência que os restantes parâmetros seleccionados exercem sobre a movimentação de cada frente e consequentemente, sobre as velocidades de decaimento de cada onda. Assim, verifica-se que um aumento do Δz_{MIN} provoca uma desaceleração sucessiva do movimento da frente (*run1* e *run2*). Conclui-se, então, que o valor de Δz_{MIN} = 0.005 é demasiado elevado, provocando um número excessivo de redefinições da grelha e atrasando a movimentação da frente. É de notar que este efeito é agravado no caso de grelhas mais finas, ou seja, com um maior número de nodos (*run2*), observando-se, igualmente, para este caso um agravamento do esforço computacional. Este efeito seria de esperar já que é natural que a integração se realize mais rapidamente em grelhas menos concentradas, como é o caso do *run1*.

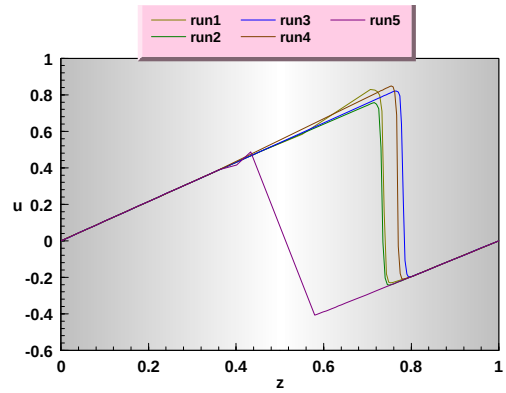

Gráfico 5.16: Resultados obtidos pelo método dinâmico de malha móvel em todos os *runs* para t=0.8.

No caso de Δz_{MIN} = 0.004 (*run3* e *run4*) não se verificam dificuldades acrescidas no avanço temporal do algoritmo (que seriam possíveis no caso dos pontos se aproximarem demasiado de forma a dificultarem a avanço da integração), sendo o ritmo de avanço da frente mais próximo do esperado. No entanto, observa-se para a frente do *run4* um ligeiro atraso em relação à correspondente ao *run5*, ao que se junta a incapacidade de formação da extremidade superior da frente para t=0.2, já referida anteriormente. Estes problemas são resultantes do facto da grelha utilizada ser constituída por menos pontos (NP=21) que a referente ao *run3* (NP=26), o que provoca visíveis dificuldades na verificação de todas as características da solução. Por outro lado, a utilização de uma grelha mais espaçada possibilita tempos de computação mais reduzidos, mas que, neste caso, não compensam de forma alguma a pior qualidade dos resultados.

Nos Gráficos 5.17 a 5.24 apresentam-se os resultados obtidos para os casos 1 a 4. É notório o atraso relativo na evolução temporal das respectivas frentes para o *run1* e *run2* (vd. Gráficos 5.17 e 5.19 respectivamente) em relação às restantes (vd. Gráficos 5.21 e 5.23), além da deficiência no perfil para t=0.2 no *run4* (vd. Gráfico 5.23).

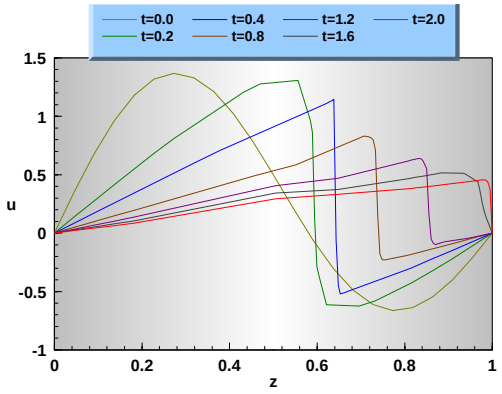

Gráfico 5.17: Resultados obtidos pelo método dinâmico de malha móvel para o *run1* (até t=2).

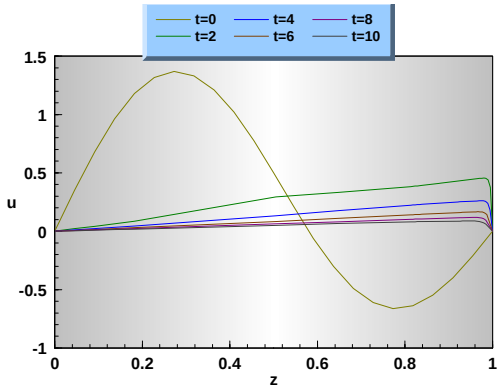

Gráfico 5.18: Resultados obtidos pelo método dinâmico de malha móvel para o *run1* (até t=10).

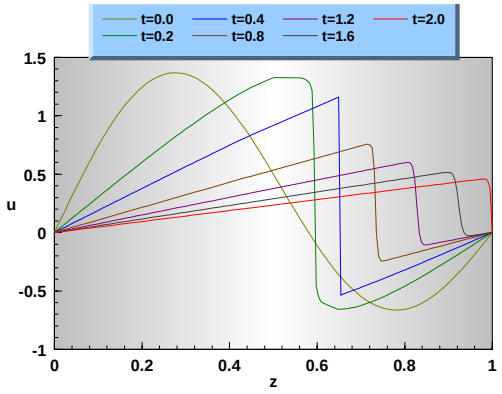

Gráfico 5.19: Resultados obtidos pelo método dinâmico de malha móvel para o *run2* (até t=2).

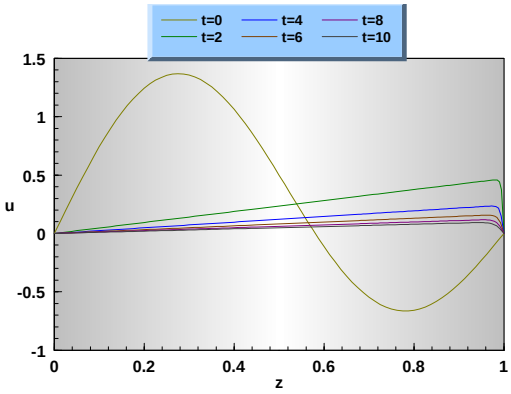

Gráfico 5.20: Resultados obtidos pelo método dinâmico de malha móvel para o *run2* (até t=10).

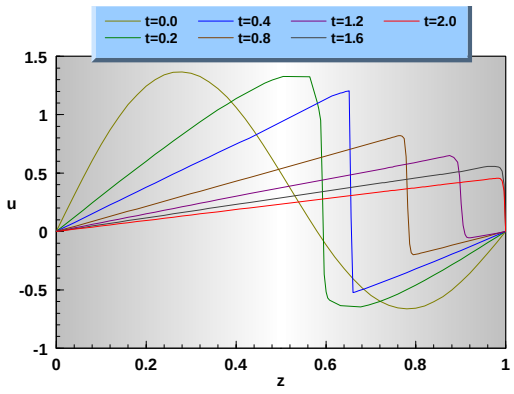

Gráfico 5.21: Resultados obtidos pelo método dinâmico de malha móvel para o *run3* (até t=2).

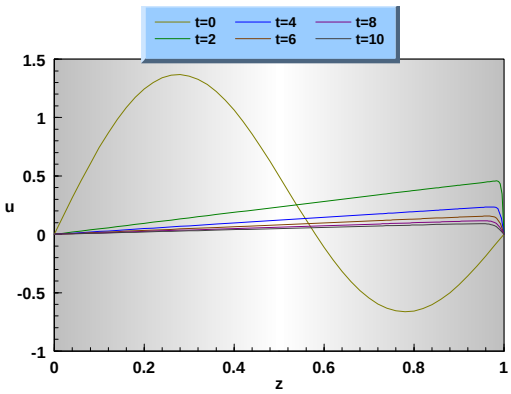

Gráfico 5.22: Resultados obtidos pelo método dinâmico de malha móvel para o *run3* (até t=10).

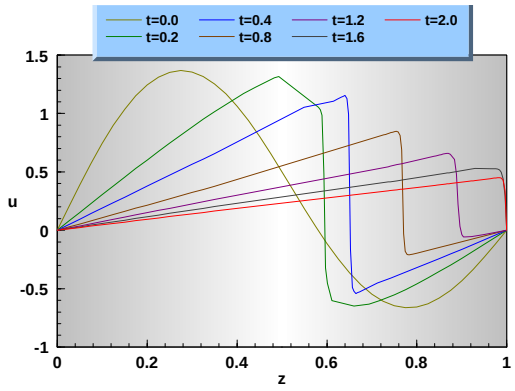

Gráfico 5.23: Resultados obtidos pelo método dinâmico de malha móvel para o *run4* (até t=2).

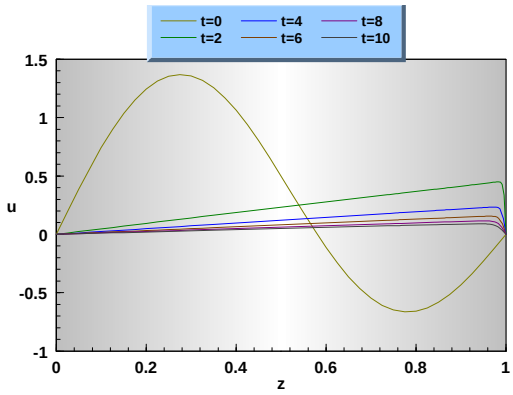

Gráfico 5.24: Resultados obtidos pelo método dinâmico de malha móvel para o *run4* (até t=10).

A partir do choque das frentes na fronteira direita (z=1), todos os *runs* exibem comportamentos semelhantes (vd. Gráficos 5.18, 5.20, 5.22 e 5.24). Por outro lado, o comportamento dos nodos é relativamente parecido em todos os casos (vd. Gráficos 5.25 a 5.28). As posições nodais situadas na região do domínio anterior à frente permanecem inalteradas, notando-se perfeitamente a migração dos nodos que formam a frente junto a z=0.6 para t=0.2. A partir desse momento, verifica-se, como seria de esperar, o deslocamento desses nodos na direcção positiva de z, acompanhando a movimentação da frente até à sua colisão com a fronteira direita.

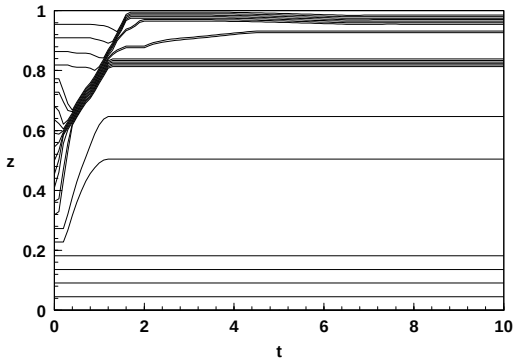

Gráfico 5.25: Resultados para a movimentação nodal da malha nas condições do *run1*.

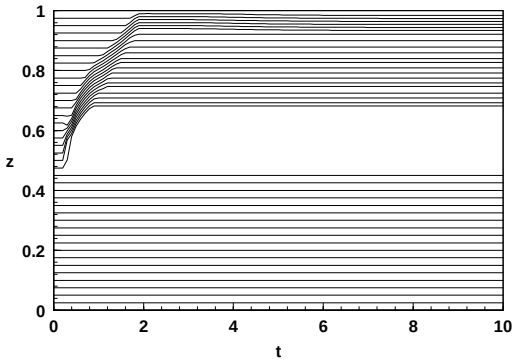

Gráfico 5.26: Resultados para a movimentação nodal de metade dos pontos da malha nas condições do *run2*.

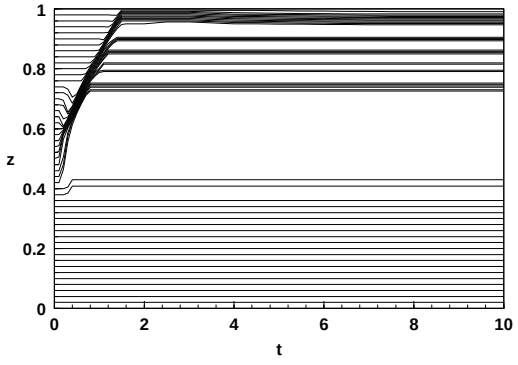

Gráfico 5.27: Resultados para a movimentação nodal da malha nas condições do *run3*.

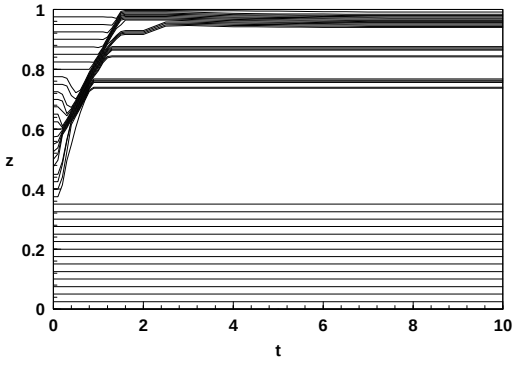

Gráfico 5.28: Resultados para a movimentação nodal da malha nas condições do *run4*.

O tempo de computação para o caso onde se obtiveram melhores resultados (*run3*) é: 3867.8 s, sendo significativamente inferior ao verificado para o melhor *run* do **Método de**

Refinamento (tcpu = 6157.5 s). Deste modo, pode-se concluir que, para este exemplo, o **Método Dinâmico de Movimentação Nodal** é mais eficiente, obtendo-se resultados de qualidade semelhante, com esforços computacionais mais reduzidos. No entanto, este resultado continua a ser ainda relativamente elevado, em relação ao conseguido pelo M.E.F.M (t_{MEFM}=1176.5 s)[29]. De qualquer modo, este método é também, do ponto de vista formal, consideravelmente mais simples do que a formulação de elementos finitos móveis (vd. Apêndice B).

6. Apresentação e Comentário dos Resultados

De forma a testar a performance dos métodos numéricos descritos anteriormente e a eficácia dos códigos desenvolvidos para a sua execução, estes foram testados por intermédio da aplicação a problemas típicos e já sobejamente conhecidos e estudados. Estes modelos padrão caracterizam-se por apresentarem características bastante variadas, permitindo identificar as circunstâncias em que os algoritmos se revelam mais ou menos eficientes. Deste modo, torna-se possível avaliar as potencialidades e limitações dos métodos, e portanto, em que condições e para que tipo de modelos estes serão mais apropriados.

A performance de cada código é analisada através de dois parâmetros essenciais e distintos: a qualidade, ou seja, a precisão dos resultados; o esforço computacional exigido para a obtenção desses resultados. Para cada exemplo, utiliza-se como referência, os resultados apresentados por *Duarte*[29], através da aplicação de um código que executa um algoritmo baseado no **Método dos Elementos Finitos Móveis** (M.E.F.M.).

6.1 – Posto de Trabalho

As execuções de cada exemplo numérico foram executadas numa *workstation* SUN *Sparcstation* de arquitectura RISC com 16 Mb de memória RAM. Como é óbvio, esta máquina trata-se do mesmo computador utilizado por *Duarte*[29], porque só nesse caso, faria qualquer sentido comparar os tempos de cpu referentes aos dois trabalhos.

É importante realçar, que, ao efectuar este tipo de analogia, não só se comparam os desempenhos entre as estratégias adaptativas e de discretização espacial, que constituem o núcleo de cada algoritmo, como também se compara a eficiência relativa entre dois integradores distintos.

Neste trabalho, como foi referido anteriormente, ambos os algoritmos utilizam o integrador implícito DASSL, para o avanço temporal da solução. No entanto, *Duarte*[29] aplica um integrador mais recente denominado por DASOLV, desenvolvido por *Jarvis e Pantelides*[71]. Este integrador baseia-se em métodos de **Diferenças Finitas** atrasadas de ordem e passo variáveis. Todas as manipulações algébricas são efectuadas em matrizes esparsas, ao contrário do que acontece com a DASSL, o que transforma a DASOLV num integrador rápido, destinado preferencialmente a sistemas algébrico-diferenciais de índice inferior a dois. Deste modo, é possível que a utilização da DASOLV possa, só por si, constituir uma vantagem na integração de certos modelos, devido à utilização de operações algébricas em matrizes esparsas, o que resulta numa maior rapidez do código desenvolvido por *Duarte*[29], em casos onde os algoritmos adaptativos tenham comportamentos semelhantes. No entanto, dada a maior complexidade do M.E.F.M., é possível que esta vantagem não se revele importante em determinados casos.

6.2 – Aplicação dos Métodos Numéricos a P.D.E.´s

6.2.1 - Exemplo 2: Adsorção num leito fixo

Este modelo[29] descreve a adsorção de um único componente num leito fixo que opera isotermicamente. O equilíbrio entre o soluto e a fase sólida é considerado instantâneo e admite-se que o escoamento é do tipo pistão.

$$\frac{\delta u}{\delta t} = \frac{1}{Pe \cdot [1 + \xi \cdot g'(u)]} \cdot \frac{\delta^2 u}{\delta z^2} - \frac{1}{1 + \xi \cdot g'(u)} \cdot \frac{\delta u}{\delta z} \qquad (6.1)$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = Pe \cdot (u - 1) \qquad (6.2)$$

$$\frac{\delta u(1,t)}{\delta z} = 0 \qquad (6.3)$$

Condição Inicial

$$u(z,0) = 0 \qquad (6.4)$$

Domínio

$$0 \le z \le 1 \qquad (6.5)$$

$$0 \le t \le 3 \qquad (6.6)$$

Parâmetros do Modelo

$u = \frac{C}{C_0}$ - concentração adimensional do componente adsorvido em relação à concentração de entrada.

C - concentração do componente adsorvido no seio do fluido.

C_0 - concentração do componente adsorvido na corrente de alimentação.

$t = \frac{\theta}{\tau}$ - tempo adimensional, normalizado em relação ao tempo de passagem.

θ - variável temporal.

$\tau = \frac{l}{V_0}$ - tempo de passagem de um elemento de fluido no leito.

l - comprimento do leito.

V_0 - velocidade do fluido no leito.

z - variável espacial.

$\xi = \frac{1-\varepsilon}{\varepsilon} \cdot \frac{q_0}{C_0} = 1$ - factor de capacidade do leito.

ε - porosidade do leito.

q_0 - concentração do componente adsorvido no sólido, em equilíbrio com C_0.

g(u) - isotérmica de equilíbrio.

$Pe = \frac{V_0 \cdot l}{D_{eff}} = 10^4$ - número de *Peclet* mássico.

D_{eff} - difusividade efectiva do componente adsorvido no leito.

Como isotérmica de equilíbrio, considerou-se a lei de acção de massas, ou seja:

$$g(u) = \frac{k \cdot u}{1 + (k-1) \cdot u} \tag{6.7}$$

em que: k – constante de equilíbrio.

Caso A: $\boxed{k = 0.1}$

Neste caso, a lei de acção de massas é desfavorável. Deste modo, a onda espalha-se ao longo do domínio, originando perfis bastante suaves. O tempo necessário para a obtenção do estado estacionário é mais elevado, já que a velocidade de propagação da onda é baixa.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 5×10^{-4}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 41 Nodos
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 0.01
- **Tempo Final** – 3.00
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 9293.3 s

Por análise dos resultados apresentados no Gráfico 6.1, verifica-se a obtenção de bons resultados, apesar da utilização de interpolações lineares, que poderiam não ser muito adequadas devido ao carácter curvilíneo dos perfis.

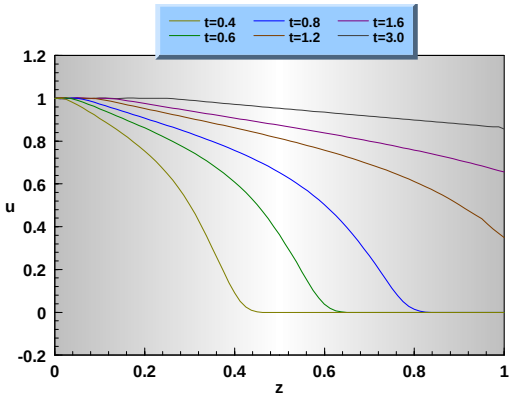

Gráfico 6.1: Resultados obtidos pelo método de refinamento para o Caso **A** do exemplo 2.

Ocorrem, no entanto, algumas dificuldades para a satisfação da condição fronteira de *Neumann*, na fronteira direita do domínio (z=1), após o choque da onda que ocorre por volta de t=1. Estas dificuldades, que provocam um significativo aumento do esforço computacional a partir desse instante de tempo, podem ser provocadas por:

❶ Alterações nas características da curva provocadas por condições de *Dirichlet* artificiais introduzidas no extremo interior a quando da integração dos subproblemas de refinamento que incluem a fronteira direita do domínio global;

❷ Inadequação das diferenças finitas centradas à resolução destes tipo de subproblemas; neste caso, talvez a utilização de diferenças descentradas *upwind* se revele mais apropriada.

Este tipo de problemas, ocorre na maioria dos modelos com estas características, verificando-se dificuldades inesperadas para o avanço do algoritmo, após o choque das ondas (abruptas ou difusionais) nas respectivas fronteiras de *Neumann*.

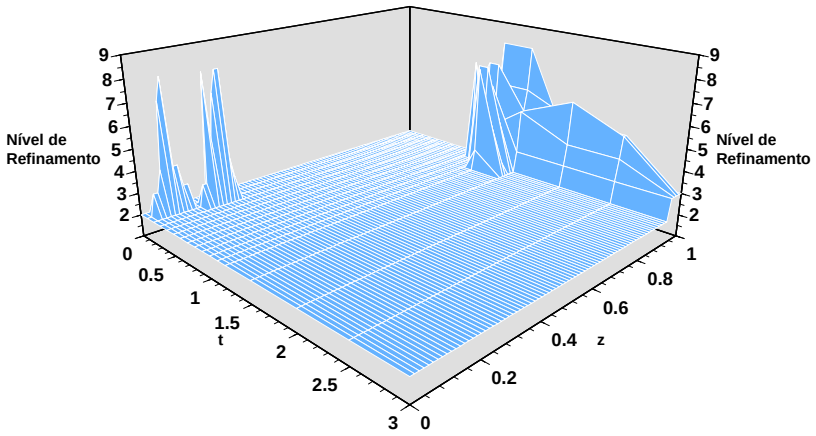

Gráfico 6.2: Resultados do nível de refinamento para o Caso **A** do exemplo 2.

As dificuldades referidas anteriormente, podem ser comprovadas por análise do Gráfico 6.2. Assim, observa-se que as zonas de refinamento acompanham o movimento da frente ao longo do domínio. A partir de t=1, a onda embate na fronteira da direita e, apesar dos perfis continuarem suaves, verifica-se uma grande actividade de refinamento nas zonas vizinhas a z=1. Esta actividade anormal provoca um aumento significativo no esforço computacional exigido pela integração.

Caso B: k = 1.0

Neste caso, a influência da difusão é mais importante. Assim, a solução desenvolve perfis significativamente mais abruptos.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 1×10^{-3}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 41 Nodos
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 0.01
- **Tempo Final** – 1.90
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 8053.2 s

Dadas as características do método de refinamento, que, por definição, executa a redução do passo espacial nas zonas onde esta se revela necessária, não existe vantagem na utilização de malhas base não uniformes, principalmente em casos onde uma onda atravessa todo domínio.

Neste caso, são utilizadas interpolações lineares porque estas se revelam bastante adequadas para modelos cujas soluções desenvolvam variações muito bruscas de gradiente entre secções do domínio próximas. A utilização de apenas dois pontos para interpolação é mais eficiente de forma a se lidar com problemas que desenvolvam frentes abruptas ou choques.

Os resultados conseguidos com os parâmetros apresentados são aceitáveis, como se pode verificar pela análise do Gráfico 6.3. No entanto, é necessária a utilização de uma malha base concentrada (malha de nível 2 com 81 nodos equidistribuidos), de forma a se obterem frentes relativamente abruptas, apesar da sua espessura ser ainda demasiado elevada. Verifica-se, igualmente, a ocorrência de oscilações ligeiras na zona superior das frentes.

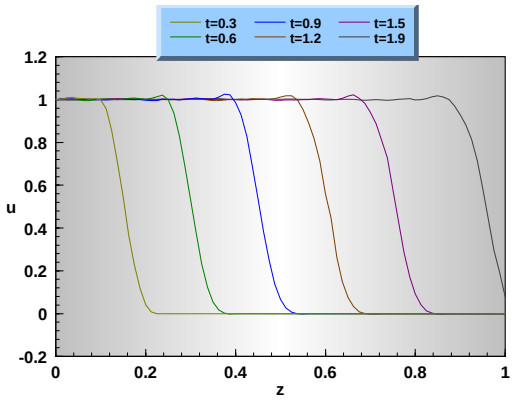

Gráfico 6.3: Resultados obtidos pelo método de refinamento para o Caso **B** do exemplo 2.

Através da análise do Gráfico 6.4, é possível visualizar as zonas do domínio onde o refinamento se revela necessário. Deste modo, verifica-se, como seria de esperar, que as zonas de refinamento acompanham o movimento da frente.

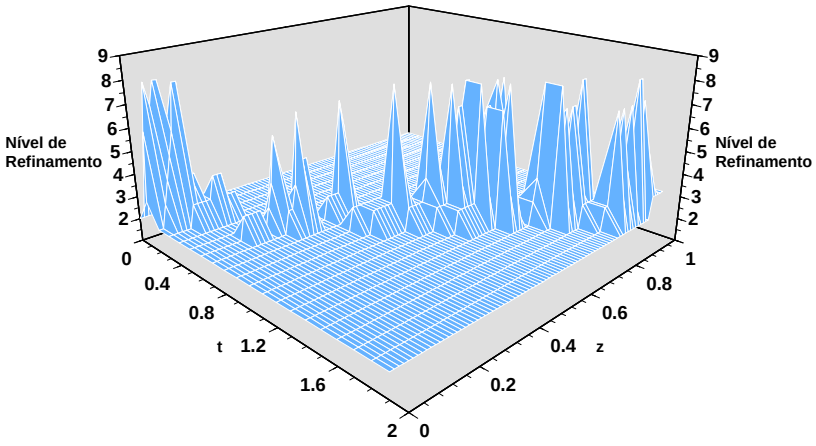

Gráfico 6.4: Resultados do nível de refinamento para o Caso **B** do exemplo 2.

Devido à maior complexidade dos perfis de solução deste caso, pela ocorrência de frentes abruptas, este foi igualmente executado através da aplicação do algoritmo de malha móvel adaptativa. Nessa execução, foram utilizados os parâmetros globais seguintes:

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 1×10^{-4}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Não Uniforme, 16 Nodos; z=[0.0, 2×10^{-4}, 5×10^{-4}, 1×10^{-3}, 1.5×10^{-3}, 2×10^{-3}, 5×10^{-3}, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0]
- **Tipo de Interpolação** – Linear
- Δz_{MIN} – 1×10^{-5}
- Δz_{MAX} – 7×10^{-3}
- α – 1
- λ – 0.75
- **Passo de Tempo Base** – 0.02
- **Tempo Final** – 1.90
- **Tempo de computação (T_{cpu})** – 8552.9 s

O método de malha móvel baseia-se numa estratégia de deslocamento nodal para zonas de maior actividade da solução. Deste modo, torna-se vantajoso utilizar uma grelha não uniforme, colocando os nodos iniciais em posições onde sejam esperadas maiores dificuldades de integração. Inicialmente, é introduzida uma perturbação na fronteira esquerda do domínio (z=0), que posteriormente se propaga através de todo o domínio sob a forma de uma frente abrupta móvel. Por isso, optou-se por definir uma grelha inicial concentrada na vizinhança da

referida fronteira. Posteriormente, o algoritmo deslocará os nodos das suas posições iniciais de forma a acompanharem o movimento da frente ao longo do domínio.

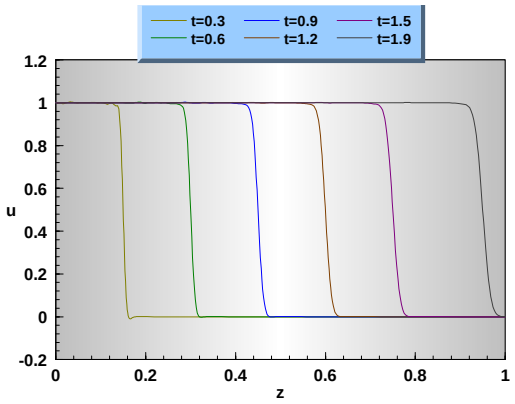

Gráfico 6.5: Resultados obtidos pelo método de malha móvel para o Caso **B** do exemplo 2.

Os perfis de solução resultantes são apresentados no Gráfico 6.5. Deste modo, verifica-se a ocorrência de uma perturbação muito ligeira na curvatura superior da frente e um *overshoot* negativo reduzido, em instantes de tempo mais baixos. Estas imperfeições são diluídas com o decorrer da integração, desaparecendo completamente, para tempos posteriores. A frente é consideravelmente mais abrupta do que a obtida pelo método de refinamento. Deste modo, é possível concluir, que do ponto de vista da precisão dos resultados o método de malha móvel adaptativa se revela mais eficiente do que a estratégia de refinamento.

No entanto, é de salientar a necessidade de fixar um Δz_{MAX} muito reduzido de forma a manter perfil da zona superior da frente estável. Assim, é introduzido um número bastante elevado de pontos adicionais intermédios durante a integração, à medida que os nodos base se afastam, de forma a manter a estabilidade dos perfis. No entanto, este facto provoca um aumento considerável do esforço computacional ao longo da integração, já que, apesar da grelha inicial de nível 2 apenas ser constituída por 32 nodos, a contínua introdução de nodos adicionais conduz a uma grelha extremamente pesada de 303 nodos. Assim, o tempo de computação de cada passo temporal aumenta drasticamente, à medida que a integração se aproxima do tempo final e a onda percorre todo o domínio. Deste modo, a integração das malhas fixas e a avaliação do erro espacial associado a cada nodo, torna-se no passo limitante do algoritmo a partir de determinado instante, devido à elevada concentração das malhas globais. Assim, é a operação de selecção de subdomínios e não a sua integração, que contribui mais significativamente para o aumento do esforço computacional ao longo da integração.

Por outro lado, verifica-se, igualmente, a ocorrência de frequentes dificuldades de convergência dos valores fronteira em cada subdomínio, o que leva a um alargamento significativo dos subdomínios em relação aos inicialmente calculados por estimativa do erro espacial. Este facto também contribui para o aumento do tempo de computação.

No Gráfico 6.6 apresenta-se a evolução da malha base inicial de 32 nodos, onde se torna perfeitamente visível o deslocamento dos nodos de forma a acompanharem o movimento da frente.

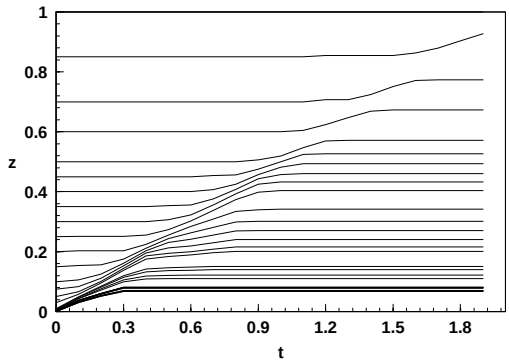

Gráfico 6.6: Evolução da malha inicial de nível 2 para o Caso **B** do exemplo 2.

De forma a analisar o efeito da variação de alguns parâmetros no desempenho do algoritmo, foram executados diversos *runs* do presente modelo até ao tempo final de t=0.3. As condições de cada *run* são resumidas na Tabela 6.1. Todos os parâmetros não referidos na tabela mantiveram-se inalterados. É de notar que as condições apresentadas anteriormente correspondem às do *run4*.

Tabela 6.1: Condições fixadas para a execução de cada *run* do modelo de adsorção em leito fixo.

Run	Δz_{MAX}	***Diferenças Finitas***	λ	***tcpu (s) (Tfinal = 0.3)***
1	7×10^{-3}	Centradas	0.5	1809.3
2	7×10^{-3}	Centradas	1.0	245.3
3	2×10^{-2}	Centradas	0.75	122.3
4	7×10^{-3}	Centradas	0.75	232.6
5	7×10^{-3}	*Biased Upwind*	0.75	516.0

Os resultados obtidos em cada *run* para os instantes t=0.1 e t=0.3 são apresentados nos Gráficos 6.7 e 6.8 respectivamente.

Assim, observa-se que, para o *run1*, ocorre grande instabilidade na zona superior da frente que praticamente não se forma, apesar de não se verificar oscilação visível na zona inferior desta. Os resultados são nitidamente insatisfatórios para as condições do *run1*.

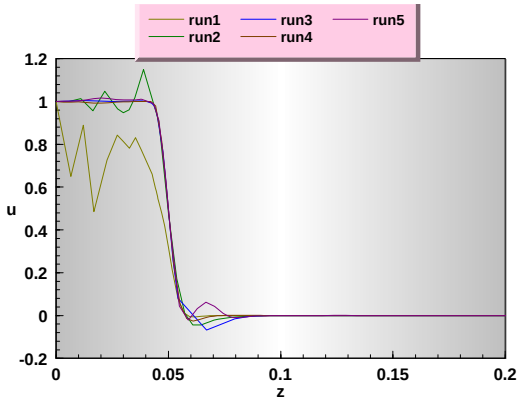

Gráfico 6.7: Resultados obtidos pelo método de malha móvel para cada *run* (t=0.1).

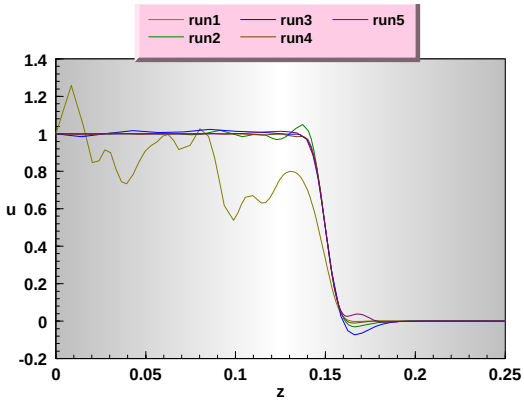

Gráfico 6.8: Resultados obtidos pelo método de malha móvel para cada *run* (t=0.3).

No caso do *run2*, verifica-se a ocorrência de oscilações moderadas na secção superior da frente e um ligeiro *overshoot* na formação da curvatura inferior desta. Deste modo, os resultados podem ser considerados como pouco satisfatórios. Para as condições do *run3*, observam-se algumas dificuldades na manutenção da secção a montante da frente em u=1. Verificam-se algumas oscilações nessa zona e um *overshoot* significativo na formação da zona inferior da frente. As condições do *run4* correspondem às apresentadas inicialmente, tendo sido já convenientemente discutidas. Finalmente, no caso do *run5*, nota-se uma formação correcta da zona anterior do perfil. No entanto, verifica-se, igualmente o desenvolvimento de uma bossa positiva anómala na curvatura da secção inferior da onda. No geral, todas as anomalias referidas atrás tendem a suavizar-se no tempo, e, eventualmente, desaparecerão com o decorrer da integração.

Deste modo, a análise dos resultados permite concluir que existem dois parâmetros críticos para a performance do método quando aplicado a este exemplo. Estes parâmetros são: factor de viscosidade internodal (λ) e a distância máxima entre nodos (Δz_{MAX}).

O ajuste correcto de λ é extremamente importante porque possibilita a fixação da velocidade nodal adequada à movimentação da frente e, deste modo, permite que o posicionamento da malha em cada instante seja o correcto de forma a manter a estabilidade da secção anterior da frente. Assim, verifica-se que se obtêm bons perfis, consistentemente, para um valor de λ a rondar 0.75. Em casos em que $\lambda < 0.75$ (*run1*) observa-se uma total incapacidade do algoritmo formar correctamente a curvatura superior da frente. Nestes casos, os nodos, apesar de partirem de posições muito próximas do eixo z=0, movimentam-se com uma velocidade demasiado elevada, não permitindo a formação da secção superior da onda. Nem mesmo a fixação de um Δz_{MAX} reduzido, conduz à manutenção da estabilidade pelo nodos adicionais. Essa instabilidade provoca, igualmente, um acréscimo nas dificuldades de avanço da integração, traduzido por um aumento drástico do tempo computacional em relação aos casos restantes. Por outro lado, para $\lambda > 0.75$ (*run2*), a movimentação nodal é demasiado lenta, provocando instabilidade visível na secção anterior da onda. No entanto, estas oscilações são consideravelmente menos graves do que as observadas para o *run1*. Assim, esta instabilidade não leva a um aumento visível do esforço computacional, já que o t_{cpu} para este caso é semelhante ao verificado no caso padrão (*run4*).

A influência do Δz_{MAX} na performance do algoritmo é discutida através da análise dos resultados obtidos nas condições do *run3*. Assim, verifica-se que um aumento no espaçamento máximo internodal provoca um acréscimo na instabilidade da secção superior e um agravamento significativo do *overshoot* negativo na curvatura inferior da frente. Por outro lado, nota-se um abaixamento importante do esforço computacional devido à menor necessidade de introdução de nodos adicionais intermédios, o que leva a definição de malhas fixas consideravelmente menos densas. No entanto, a poupança computacional conseguida, não compensa o abaixamento na qualidade dos resultados numéricos obtidos.

A utilização de diferenças finitas *biased upwind* (*run5*) não conduz a uma melhoria considerável dos resultados. Pelo contrário, apesar de não ocorrerem dificuldades na formação da curva superior da frente, verifica-se o desenvolvimento de uma bossa positiva na secção inferior desta. Assim, a utilização de diferenças finitas atrasadas conduz a piores resultados na definição do perfil imediatamente posterior à frente. Para além disso, verifica-se um aumento significativo do esforço computacional, para este tipo de diferenças finitas.

Os desempenhos computacionais associados a cada um dos métodos comparados neste trabalho, encontram-se resumidos na tabela seguinte:

Tabela 6.2: Desempenhos computacionais para o modelo de adsorção em leito fixo.

Caso	*Algoritmo*	*Tempo de Computação (s)*
A	**Refinamento**	9293.2
	M.E.F.M.[29]	29428.3
B	**Refinamento**	8053.2
	Malha Móvel	8552.9
	M.E.F.M.[29]	6722.6

No caso **A**, que à partida, não representa grandes problemas de integração devido à suavidade dos perfis desenvolvidos, verifica-se um melhor desempenho computacional do método de refinamento em relação ao M.E.F.M.. Os resultados obtidos por cada método são muito semelhantes. Assim, conclui-se que o refinamento se revela mais eficiente que a estratégia de elementos finitos móveis para este caso.

No que diz respeito ao caso **B**, verifica-se um melhor comportamento numérico do M.E.F.M., em relação aos restantes. Os desempenhos computacionais de cada um desses métodos são muito semelhantes. No entanto, verifica-se que o algoritmo de malha móvel consegue obter melhores resultados, do ponto de vista do desenvolvimento correcto das frentes abruptas.

6.2.2 - Exemplo 3: Reactor Pistão Difusional

Este exemplo[29] descreve um reactor tubular isotérmico, com o qual é aplicado o modelo pistão difusional. De forma a se estudar o efeito conjugado de termos difusivos, convectivos e reaccionais, introduz-se um termo de difusão axial no modelo, ao mesmo tempo que se admite a ocorrência de uma reacção homogénea de primeira ordem $A \rightarrow B$, no interior do reactor.

$$\frac{\delta u}{\delta t} = \frac{1}{Pe} \cdot \frac{\delta^2 u}{\delta z^2} - \frac{\delta u}{\delta z} - Da \cdot u \tag{6.8}$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = Pe \cdot (u - 1) \tag{6.9}$$

$$\frac{\delta u(1,t)}{\delta z} = 0 \tag{6.10}$$

Condição Inicial

$$u(z,0) = 0 \tag{6.11}$$

Domínio

$$0 \le z \le 1 \tag{6.12}$$

$$0 \le t \le 1.25 \tag{6.13}$$

Parâmetros do Modelo

u - concentração da espécie A.
z - variável espacial.
$t = \frac{\theta}{\tau}$ - tempo adimensional, normalizado em relação ao tempo de passagem.
θ - variável temporal.
$\tau = \frac{l}{V_0}$ - tempo de passagem de um elemento de fluido no leito.
l - comprimento do leito.
V_0 - velocidade do fluido no leito.
$Da = \frac{k \cdot l}{V_0} = 1$ - número de *Damkhöler*.
k - constante de velocidade de reacção.
$Pe = \frac{V_0 \cdot l}{D_{ax}}$ - número de *Peclet* axial.
D_{ax} - difusividade axial da espécie A.

Caso A: $\boxed{Pe = 10^2}$

Trata-se de um modelo de carácter difusional e, por isso, ocorre um afastamento da situação de pistão puro. Deste modo, a onda espalha-se ao longo do reactor e a sua velocidade de propagação diminui.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 5×10^{-4}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 21 Nodos
- **Tipo de Interpolação** – *Splines* Cúbicas 5 Pontos
- **Passo de Tempo Base** – 0.005
- **Tempo Final** – 1.250
- **Nº Máximo de Iterações** – 10
- **Tempo de computação (T_{cpu})** – 707.4 s

Por análise dos Gráficos 6.9 e 6.10, verifica-se a obtenção de bons resultados na integração do exemplo enunciado, com os parâmetros apresentados anteriormente. A utilização de interpolações com *splines* cúbicas ajusta-se perfeitamente às características curvilíneas da solução. Por outro lado, observa-se que não é necessário aplicar uma malha muito apertada para a obtenção de resultados aceitáveis. A malha base utilizada é uniforme,

pelas razões já expostas no exemplo anterior. Deste modo, ocorre uma redução drástica do tempo de cpu em relação a casos semelhantes.

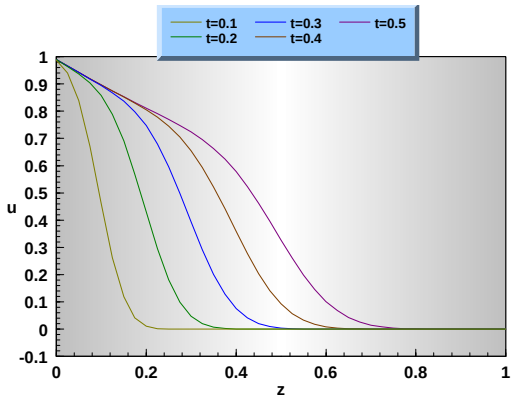

Gráfico 6.9: Resultados obtidos pelo método de refinamento no Caso **A** do exemplo 3 (tfinal=0.5).

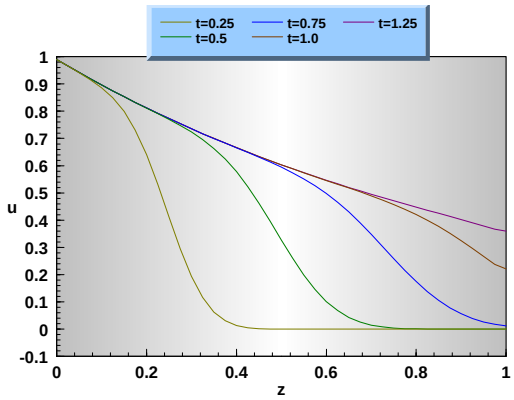

Gráfico 6.10: Resultados obtidos pelo método de refinamento no Caso **A** do exemplo 3 (tfinal=1.25).

Obtêm-se resultados muito aceitáveis, com um esforço computacional reduzido, já que a necessidade de refinamento é bastante moderada e apenas ocorre em dois períodos breves: nos instantes iniciais, de forma a lidar correctamente com a descontinuidade introduzida em z=0; nos instantes finais da integração, após o embate da onda com a fronteira z=1 (vd. Gráfico 6.11).

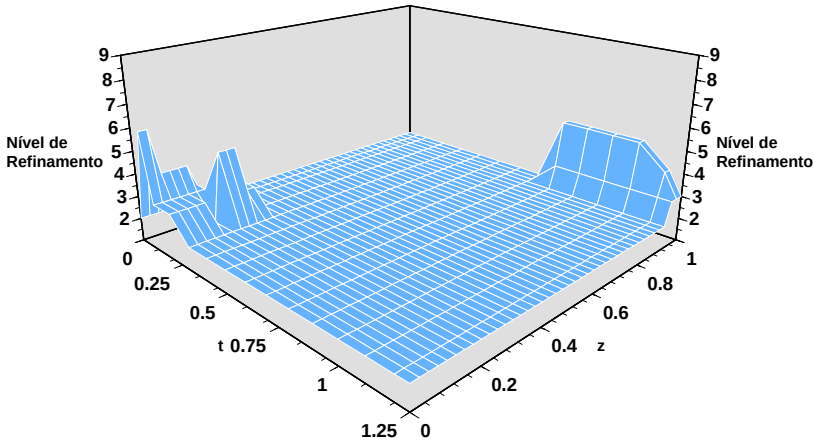

Gráfico 6.11: Resultados do nível de refinamento para o Caso **A** do exemplo 3.

Neste caso, a solução apenas desenvolve perfis difusionais bastante suaves. Deste modo, verifica-se uma menor dificuldade de integração e, consequentemente uma melhor performance do método de refinamento.

Caso B: $\boxed{Pe = 10^4}$

Neste exemplo, a difusão assume reduzida importância, o que provoca a ocorrência de frentes abruptas na solução. O comportamento do reactor aproxima-se da situação de pistão puro, em que o declive da frente tende para infinito. Neste caso particular, a onda sofre uma ligeira distorção, devido à presença do termo difusivo, mantendo-se, no entanto, com uma forma abrupta.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 1×10^{-3}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 41 Nodos
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 0.01
- **Tempo Final** – 1.00
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 7368.6 s

Através da análise dos Gráficos 6.12 e 6.13, verifica-se a evolução da frente ao longo do reactor. A partir de t=1, a onda embate na fronteira direita do domínio (z=1), atingindo-se o estado estacionário. No entanto, nota-se que a espessura da frente desenvolvida é ligeiramente maior que o esperado, apesar de se utilizar uma malha base apertada. Por outro lado, observam-se oscilações muito ligeiras na zona superior da frente, que se esbatem ao longo do tempo.

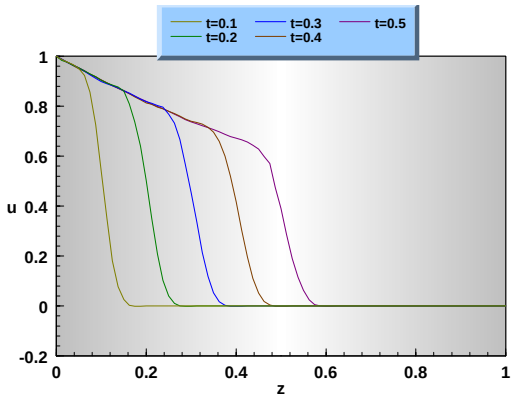

Gráfico 6.12: Resultados obtidos pelo método de refinamento no Caso **B** do exemplo 3 (tfinal=0.5).

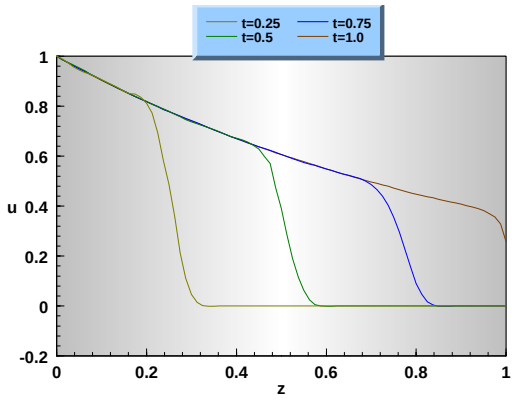

Gráfico 6.13: Resultados obtidos pelo método de refinamento no Caso **B** do exemplo 3 (tfinal=1.0).

A integração deste caso revela-se mais difícil que no caso anterior, devido ao carácter pistão da solução, que origina gradientes mais elevados, exigindo um esforço computacional mais elevado. A interpolação linear é a mais adequada para o caso em estudo, devido às características abruptas da frente.

No Gráfico 6.14, apresenta-se o grau de refinamento associado a cada nodo do domínio de nível 2, notando-se perfeitamente a evolução das zonas de refinamento de forma a acompanharem o deslocamento da frente.

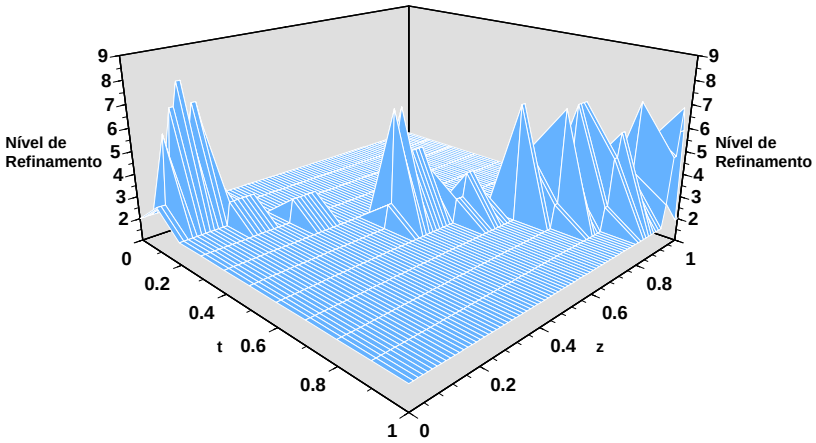

Gráfico 6.14: Resultados do nível de refinamento para o Caso **B** do exemplo 3.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 1×10^{-4}
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Não Uniforme, 16 Nodos; z=[0.0, 2×10^{-4}, 5×10^{-4}, 1×10^{-3}, 1.5×10^{-3}, 2×10^{-3}, 5×10^{-3}, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0]
- **Tipo de Interpolação** – Linear
- **Δz_{MIN}** – 1×10^{-5}
- **Δz_{MAX}** – 7×10^{-3}
- **α** – 1
- **λ** – 0.75
- **Passo de Tempo Base** – 0.02
- **Tempo Final** – 1.00
- **Tempo de computação (T_{cpu})** – 7000.7 s

Nos Gráficos 6.15 e 6.16 apresentam-se os perfis correspondentes à aplicação do método de malha adaptativa ao caso **B** do presente exemplo. Os parâmetros fixados nesta execução estão listados acima e correspondem aos utilizados no *run4* do exemplo de adsorção (Exemplo 2).

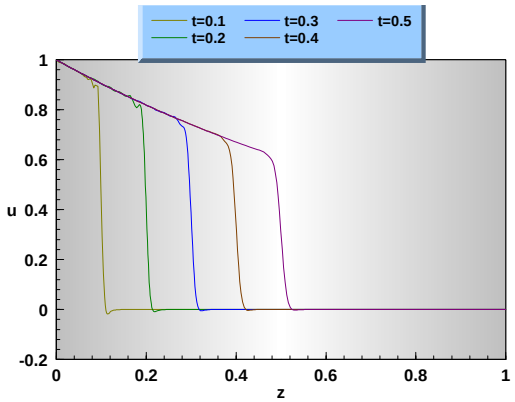

Gráfico 6.15: Resultados obtidos pelo método de malha móvel no Caso **B** do exemplo 3 (tfinal=0.5).

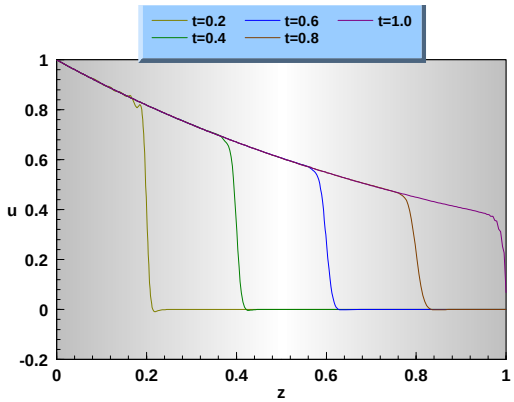

Gráfico 6.16: Resultados obtidos pelo método de malha móvel no Caso **B** do exemplo 3 (tfinal=1.0).

Por análise dos resultados, observa-se a ocorrência de oscilações ligeiras nas zonas imediatamente anteriores à frente, para instantes de tempo iniciais. Nota-se, igualmente, um pequeno *overshoot* negativo na extremidade inferior da frente para t=0.1. Estas anomalias diluem-se com o avanço da integração, desaparecendo completamente para t > 0.3.

As frentes desenvolvidas são bastante abruptas, apresentando uma espessura consideravelmente inferior à das frentes obtidas com o método de refinamento. Assim, verifica-se uma reprodução correcta das variações bruscas dos gradientes da solução e da movimentação da respectiva onda. Deste modo, pode-se considerar que os resultados obtidos são, no geral, bastante aceitáveis.

A evolução dos nodos iniciais da malha de nível 2, é apresentada no Gráfico 6.17. Optou-se por não se representar o comportamento dos nodos adicionais por uma questão de maior visibilidade da movimentação da malha. No entanto, através da análise da malha base inicial, é possível verificar que os nodos acompanham satisfatoriamente o deslocamento da onda.

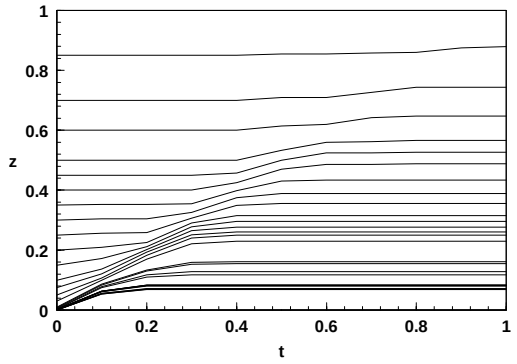

Gráfico 6.17: Evolução da malha inicial de nível 2 para o Caso **B** do exemplo 3.

A definição de um Δz_{MAX} muito reduzido, de forma a manter a estabilidade da secção superior da frente, conduz à introdução frequente de nodos adicionais, à medida que a onda percorre o reactor. Desse modo, apesar da malha inicial de nível 2 ser constituída por apenas 32 pontos (a maioria dos quais concentrada na vizinhança da fronteira esquerda do domínio), verifica-se que a malha final correspondente apresenta 287 pontos. Este facto provoca um aumento significativo do esforço computacional exigido pela integração ao longo do tempo, provocado pela crescente dificuldade na integração das malhas fixas que se tornam extremamente concentradas e extensas. Este problema pode ser ultrapassado por um aumento de Δz_{MAX}, permitindo um maior afastamento entre nodos adjacentes.

Nos Gráficos 6.18 e 6.19 são apresentados os resultados obtidos por aplicação de $\Delta z_{MAX}= 1.5\times10^{-2}$, aproximadamente o dobro do utilizado na execução anterior. Os restantes parâmetros permanecem inalterados.

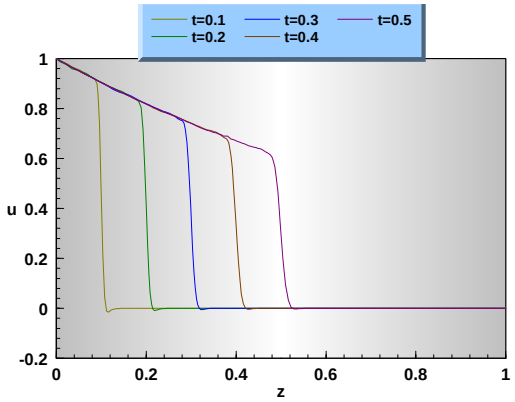

Gráfico 6.18: Perfis de resultados obtidos pelo método de malha móvel adaptativa no *run* adicional para o Caso **B** do exemplo 3 (tfinal=0.5).

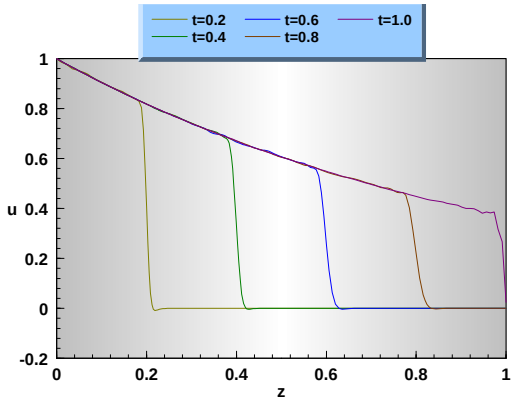

Gráfico 6.19: Perfis de resultados obtidos pelo método de malha móvel adaptativa no *run* adicional para o Caso **B** do exemplo 3 (tfinal=1.0).

Ao contrário do que acontecia no exemplo anterior, o aumento do valor de Δz_{MAX} não provoca um aumento visível da instabilidade dos perfis. Observa-se ainda um muito ligeiro *overshoot* negativo na curvatura inferior da frente que desaparece com o tempo.

Verifica-se uma estabilidade razoável dos perfis iniciais, que, em oposição ao que acontecia na execução anterior, piora ao longo do tempo. Assim, observam-se oscilações mais importantes para tempos mais elevados.

Por outro lado, o aumento de Δz_{MAX}, provoca uma diminuição na colocação de nodos adicionais (a malha final de nível 2 é constituída por apenas 137 nodos) e um consequente decréscimo no esforço computacional. Deste modo, o tempo computacional associado a esta execução é consideravelmente inferior ao anterior: T_{cpu} = 2998.8 s.

A análise do Gráfico 6.20 permite concluir que uma maior liberdade no afastamento nodal conduz a uma movimentação mais vigorosa dos pontos da malha base inicial de nível 2.

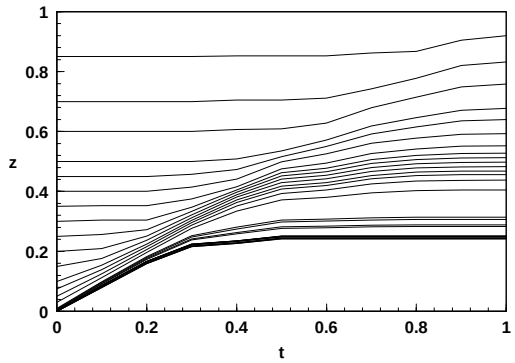

Gráfico 6.20: Evolução da malha inicial de nível 2 para o *run* adicional do Caso **B** do exemplo 3.

De qualquer modo, considera-se que, no geral, os resultados obtidos pela execução original conduzem a melhores resultados globais, apesar de exigirem um esforço computacional superior.

Na Tabela 6.3 apresentam-se os desempenhos computacionais associados a cada um dos diferentes métodos para o exemplo do reactor pistão difusional (Exemplo 3).

Tabela 6.3: Desempenhos computacionais para o modelo do reactor pistão difusional.

Caso	*Algoritmo*	*Tempo de Computação (s)*
A	**Refinamento**	707.4
	M.E.F.M.[29]	5927.5
B	**Refinamento**	7368.6
	Malha Móvel	7000.7
	M.E.F.M.[29]	10436.2

O caso **A** não representa grandes problemas de integração devido à suavidade dos perfis desenvolvidos. Assim, observa-se um considerável melhor desempenho computacional do algoritmo de refinamento em relação ao M.E.F.M.. Como os resultados obtidos por cada método são análogos, conclui-se que o refinamento se revela muito mais eficaz neste caso.

Para o caso **B**, verifica-se um pior desempenho numérico do M.E.F.M., em relação aos restantes, apesar de proporcionar a obtenção de resultados ligeiramente melhores do ponto de vista da qualidade dos perfis. As performances computacionais de cada um dos restantes métodos são bastante semelhantes, mas, verifica-se que o algoritmo de malha móvel obtém perfis mais correctos, relativamente à reprodução dos fortes gradientes da solução. Assim, conclui-se que a estratégia mais adequada para este exemplo é a da malha móvel adaptativa.

6.2.3 - Exemplo 4: Combustão

Este modelo[4,29,116] simula a dinâmica de uma chama. Deste modo, a temperatura na posição do ponto quente (z=0) aumenta gradualmente, até atingir o valor normalizado de $1+\alpha$. A partir desse instante, forma-se uma frente de carácter abrupto que se propaga com a velocidade $e^{\alpha\cdot\delta}\big/\left(2\cdot(1+\alpha)\right)$, até colidir com a fronteira direita (z=1). A velocidade da frente é aproximadamente exponencial, porque normalmente, o valor de δ é elevado e o correspondente a α situa-se próximo da unidade.

$$\frac{\delta u}{\delta t} = \frac{\delta^2 u}{\delta z^2} + Da\cdot(1+\alpha-u)\cdot e^{-\frac{\delta}{u}} \tag{6.14}$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = 0 \tag{6.15}$$

$$u(1,t) = 1 \tag{6.16}$$

Condição Inicial

$$u(z,0) = 1 \tag{6.17}$$

Domínio

$$0 \le z \le 1 \tag{6.18}$$

$$0 \le t \le 0.29 \tag{6.19}$$

Parâmetros do Modelo

$u = \frac{T}{T_0}$ - temperatura normalizada da chama.

T - temperatura da chama.

T_0 - temperatura inicial da chama.

t - variável temporal.

z - variável espacial.

$Da = \frac{R \cdot e^{\delta}}{\alpha \cdot \delta}$

$\delta = \frac{E_a}{R} = 20$

E_a - energia de activação.

$\alpha = 1$

$R = 5$

Parâmetros da Execução –

Tipo de Diferenças Finitas	–	5 Pontos Centradas
Tolerância do Método	–	5×10^{-3}
Tolerância do Integrador	–	1×10^{-6}
Grelha Inicial	–	Uniforme; 41 Nodos
Tipo de Interpolação	–	*Splines* Cúbicas 5 Pontos
Passo de Tempo Base	–	0.01
Tempo Final	–	0.29
Nº Máximo de Iterações	–	9
Tempo de computação (T_{cpu})	–	319.6 s

A aplicação de uma malha uniforme, apesar de relativamente pesada (81 nodos para a malha de nível 2) origina a obtenção de perfis correctos (vd. Gráfico 6.21), comparativamente aos que foram obtidos por *Duarte*[29].

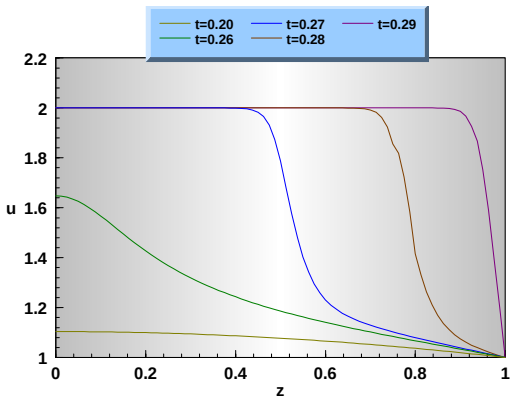

Gráfico 6.21: Resultados obtidos pelo método de refinamento no exemplo 4.

O refinamento apenas se revela necessário a partir de t=0.27, após a formação da onda abrupta que percorre o domínio com uma velocidade extremamente elevada. Observa-se uma ligeira distorção no perfil t=0.28 (Gráfico 6.21), cuja origem pode ser explicada pelas limitações da estratégia escolhida para o tratamento das condições fronteira interiores nos subdomínios de refinamento (vd. Capítulo 4). A opção de introduzir condições de *Dirichlet* nesses pontos fronteira pode provocar anomalias como a detectada neste caso, apesar desta não ser muito visível. Como seria de esperar, as zonas de refinamento acompanham o movimento da frente ao longo do domínio nos instantes finais da integração (vd. Gráfico 6.22).

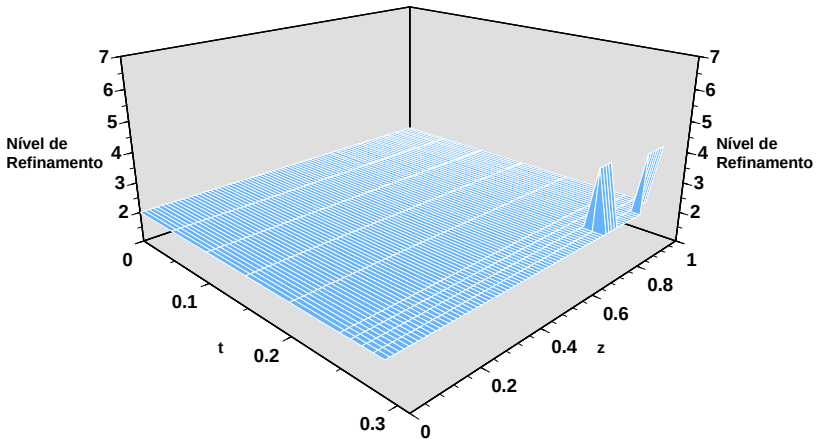

Gráfico 6.22: Resultados do nível de refinamento para o exemplo 4.

De modo a possibilitar o desenvolvimento correcto da solução, é necessário recorrer a uma grelha inicial concentrada (41 nodos). Verifica-se que a utilização de uma malha uniforme é adequada, já que a frente atravessa todo o domínio. Dado o carácter curvilíneo da solução, as interpolações são efectuadas através de *splines* cúbicas.

Assim, conclui-se que o método de refinamento conduz a resultados aceitáveis com tempos de computação reduzidos, devido à relativa facilidade de avanço da integração. Portanto, este método revela-se bastante adequado e eficaz para o exemplo em estudo.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas
- **Tolerância do Método** – 5×10^{-4}
- **Tolerância do Integrador** – 1×10^{-6}
- **Grelha Inicial** – Uniforme; 41 Nodos
- **Tipo de Interpolação** – *Splines* Cúbicas 5 Pontos
- Δz_{MIN} – 3×10^{-3}
- Δz_{MAX} – 1.0
- α – 1
- λ – 0.4
- **Passo de Tempo Base** – 0.01
- **Tempo Final** – 0.29
- **Tempo de computação (T_{cpu})** – 774.2 s

A aplicação do método de malha móvel adaptativa a este exemplo conduz aos resultados apresentados no Gráfico 6.23. Os elevados gradientes da onda são bem reproduzidos pelo algoritmo, obtendo-se perfis correctos com um esforço computacional moderado. A malha inicial uniforme é suficientemente fina para dispensar a introdução de nodos adicionais com o decorrer da integração.

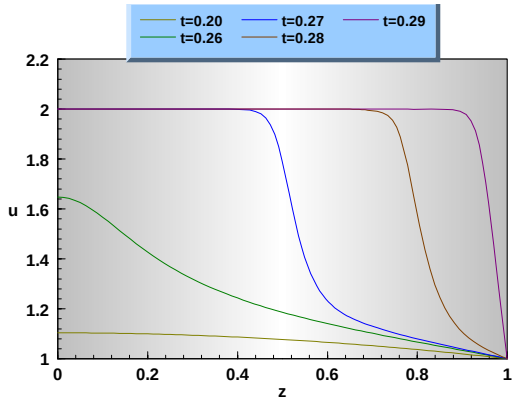

Gráfico 6.23: Resultados obtidos pelo método de malha móvel no exemplo 4.

A execução deste exemplo conduz a dificuldades de obtenção de convergência nas condições fronteira dos subproblemas gerados (principalmente no passo entre t=0.26 e 0.27). Neste passo, verifica-se a necessidade de introdução da mobilidade nodal num subdomínio bastante alargado em relação ao inicialmente seleccionado pela operação de estimativa dos erros espaciais. No entanto, este problema na aplicação do algoritmo, tem-se revelado característico de modelos caracterizados por frentes abruptas que atravessam a totalidade do domínio com velocidades relativamente elevadas.

No Gráfico 6.24 é possível visualizar o comportamento da malha de nível 2 a partir de t=0.20. A evolução dos perfis anterior a esse instante é demasiado lenta para necessitar da introdução de submalhas móveis. De facto, verifica-se que a movimentação nodal apenas é necessária a partir de t=0.26. Após esse instante, os nodos acompanham o deslocamento da onda de uma forma satisfatória. É de salientar a grande extensão relativa do subdomínio integrado no intervalo de tempo [0.26,0.27].

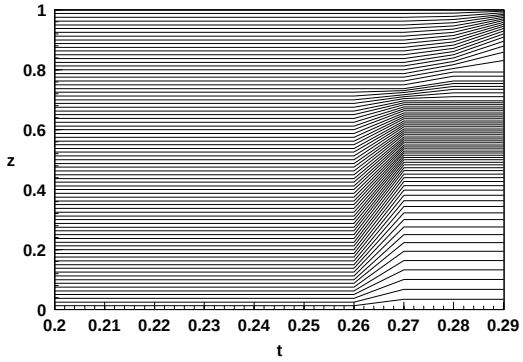

Gráfico 6.24: Evolução da malha de nível 2 para o exemplo 4.

Os resultados para o desempenho numérico de cada algoritmo estão resumidos na Tabela 6.4.

Tabela 6.4: Desempenhos computacionais para o modelo de combustão.

Algoritmo	*Tempo de Computação (s)*
Refinamento	319.6
Malha Móvel	774.2
M.E.F.M.[29]	6756.8

Assim, verifica-se que ambos os métodos apresentados neste trabalho obtêm melhores desempenhos computacionais que o M.E.F.M. Os perfis calculados são semelhantes, apesar de se observarem ligeiras perturbações nos resultados obtidos com o método de refinamento. Deste modo, conclui-se que, do ponto de vista do esforço computacional exigido, os métodos de refinamento e de malha móvel se revelam mais eficientes. No que diz respeito à precisão dos resultados, o método de refinamento é ligeiramente menos adequado que os restantes. Se se considerar a conjugação destes dois factores, conclui-se que o método mais eficaz para a integração deste modelo é o método de malha móvel adaptativa.

6.2.4 - Exemplo 5: Difusão, Convecção e Reacção numa Partícula Plana

O modelo[29] apresentado nesta secção, descreve o perfil de concentrações numa partícula plana em condições isotérmicas. Para tal, admite-se um catalisador poroso, onde ocorre uma reacção catalítica $A \rightarrow P$, além dos fenómenos de transporte de massa por difusão e convecção. Como $\lambda_m \neq 0$, os perfis de solução não são simétricos, tendo-se que considerar um domínio espacial [0,2]. O arranque do processo é efectuado através da introdução de perturbações em degrau em ambas as fronteiras do domínio, correspondentes (em termos físicos) ao início da alimentação de reagente.

$$\frac{\delta u}{\delta t} = \frac{\delta^2 u}{\delta z^2} - \lambda_m \cdot \frac{\delta u}{\delta z} - \phi^2 \cdot u \qquad (6.20)$$

Condições Fronteira

$$u(0,t) = 1 \qquad (6.21)$$
$$u(1,t) = 1 \qquad (6.22)$$

Condição Inicial

$$u(z,0) = 0 \qquad (6.23)$$

Domínio

$$0 \leq z \leq 2 \qquad (6.24)$$
$$0 \leq t \leq 0.5 \qquad (6.25)$$

Parâmetros do Modelo

$u = \frac{C}{C_0}$ - concentração adimensional de A referida à concentração no seio do fluido.

C - concentração de A no seio do fluido.

C_0 - concentração de A na corrente de alimentação.

$t = \frac{\theta}{\tau_D}$ - variável de tempo adimensional.

θ - variável temporal.

$\tau_D = \frac{\varepsilon_P \cdot l^2}{D_{eff}}$ - constante de tempo de difusão.

ε_P - porosidade da partícula.

l - semi-espessura da partícula.

D_{eff} - difusividade efectiva do reagente no leito.

z - variável espacial.

$\lambda_m = \frac{V_0 \cdot l}{D_{eff}} = 10$ - número de *Peclet* mássico intraparticular.

V_0 - velocidade do fluido no interior da partícula em relação à área de secção normal do escoamento.

$\phi = l \cdot \sqrt{\frac{k}{D_{eff}}} = 2$ - módulo de *Thiele*.

k - constante de reacção.

Parâmetros da Execução –

↳ **Tipo de Diferenças Finitas**	–	5 Pontos Centradas
↳ **Tolerância do Método**	–	1×10^{-2}
↳ **Tolerância do Integrador**	–	1×10^{-6}
↳ **Grelha Inicial**	–	Uniforme; 21 Nodos
↳ **Tipo de Interpolação**	–	Splines Cúbicas 5 Pontos
↳ **Passo de Tempo Base**	–	0.01
↳ **Tempo Final**	–	1.50
↳ **Nº Máximo de Iterações**	–	10
↳ **Tempo de computação (T_{cpu})**	–	167.3 s

Através da utilização de uma malha base uniforme, relativamente esparsa (apenas 21 pontos) e de um grau de precisão baixo (TOL=1×10^{-2}), obtêm-se bons resultados para a evolução da solução (vd. Gráfico 6.25 e 6.26) comparativamente aos obtidos por *Duarte*[29]. Apenas se verifica a necessidade do refinamento da malha, no início da integração, junto à fronteira z=0 (vd. Gráfico 6.27), devido à perturbação inicial em degrau aí exercida. A evolução dos perfis efectua-se sem qualquer dificuldade e o refinamento, quando se revela necessário, é bastante ligeiro.

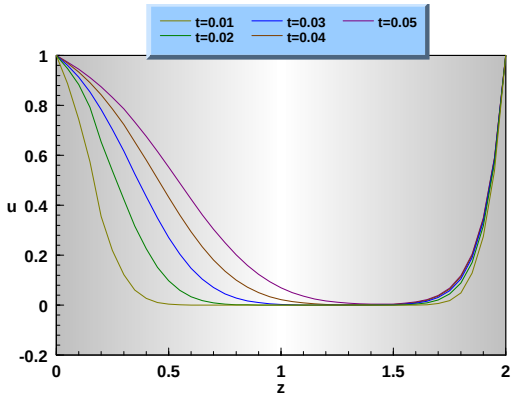

Gráfico 6.25: Resultados obtidos pelo método de refinamento no exemplo 5 (tfinal=0.05).

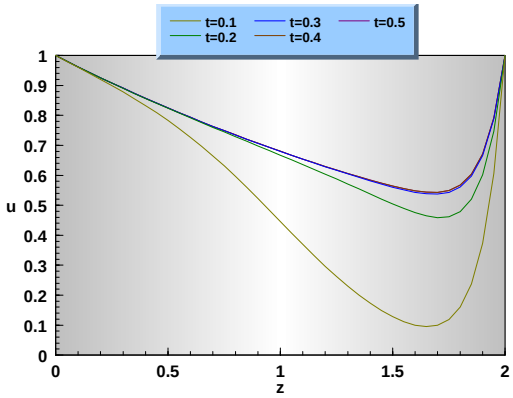

Gráfico 6.26: Resultados obtidos pelo método de refinamento no exemplo 5 (tfinal=0.5).

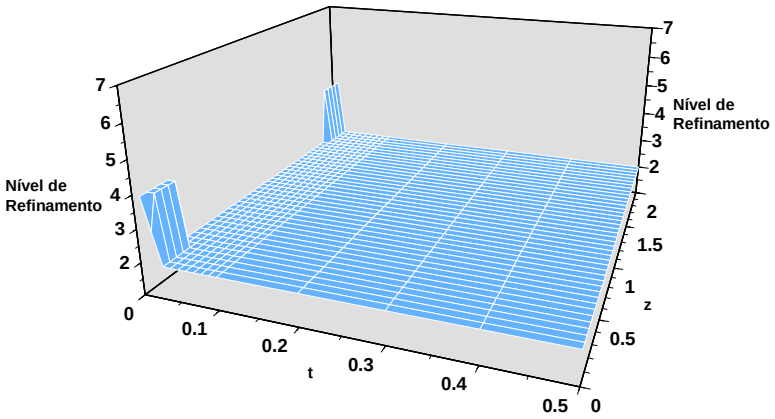

Gráfico 6.27: Resultados do nível de refinamento para o exemplo 5.

Por outro lado, este exemplo pode também ser utilizado para exemplificar a possibilidade da ocorrência de sobre-refinamento da malha, no caso da tolerância exigida se revelar demasiado baixa em relação à complexidade dos perfis da solução. Para tal, são apresentados nos Gráficos 6.28 e 6.29 os resultados obtidos através da utilização de uma tolerância 10× inferior (TOL=1×10^{-3}) à do caso anterior, mantendo todos os restantes parâmetros inalterados.

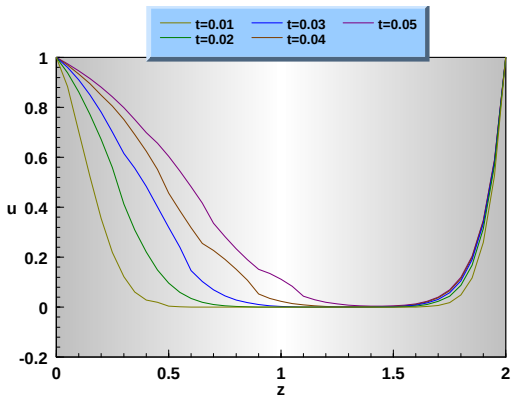

Gráfico 6.28: Resultados obtidos através de refinamento no *run* adicional do exemplo 5 (tf=0.05).

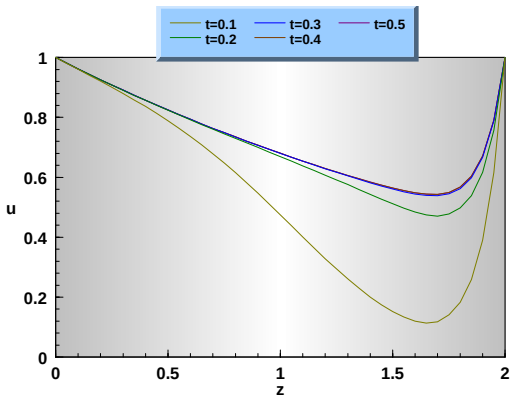

Gráfico 6.29: Resultados obtidos através de refinamento no *run* adicional do exemplo 5 (tf=0.5).

Por análise do Gráfico 6.28, verifica-se a ocorrência de distorções visíveis, nos perfis iniciais da solução, em zonas de gradientes suaves da solução, que, à partida não deveriam colocar dificuldades no avanço temporal da solução, e, portanto não se esperaria que fossem sujeitas a refinamento. Estas distorções situam-se entre zonas com níveis de refinamento diferentes, sendo diluídas à medida que a integração prossegue. Assim, é possível concluir-se que, neste exemplo, o tratamento dado às fronteiras interiores (através da fixação de condições de *Dirichlet* artificiais) pode originar problemas e levar à obtenção de resultados incorrectos. Como a exigência de um grau de precisão maior origina um aumento significativo no nível de refinamento de algumas zonas do domínio (vd. Gráfico 6.30), tal efeito provoca igualmente, um aumento drástico no esforço computacional exigido (T_{cpu}=271.0 s), sem daí se retirarem quaisquer benefícios do ponto de vista de um aumento na qualidade dos resultados, muito pelo contrário.

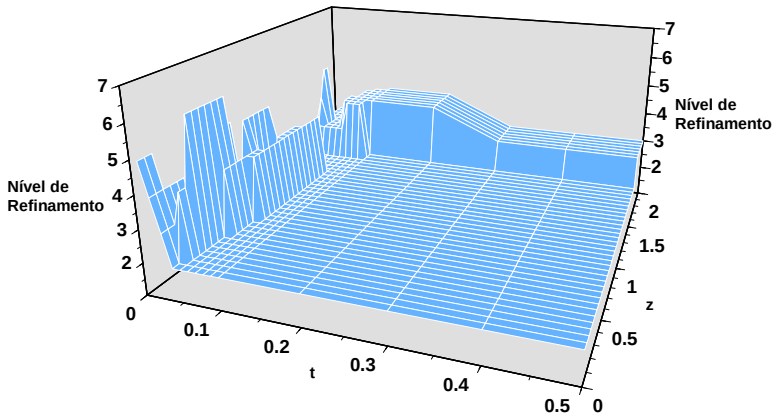

Gráfico 6.30: Resultados do nível de refinamento para o *run* adicional do exemplo 5.

O desempenho computacional entre o método de refinamento considerado neste trabalho e o algoritmo M.E.F.M. utilizado por *Duarte*[29] é resumido na tabela seguinte.

Tabela 6.5: Desempenhos computacionais para o modelo de difusão, convecção e reacção numa partícula plana.

Algoritmo	***Tempo de Computação (s)***
Refinamento	167.3
M.E.F.M.[29]	5765.0

Por análise da Tabela 6.5, verifica-se que, para a obtenção de resultados aceitáveis, o algoritmo de refinamento apenas necessita de recorrer a um esforço computacional reduzido, comparativamente ao exigido pelo M.E.F.M. ($T_{cpu} << T_{MEFM}$). Deste modo, pode-se concluir que para o caso do presente exemplo, cujos resultados não exibem gradientes elevados, um método relativamente simples como o algoritmo de refinamento é suficientemente poderoso para a obtenção de perfis aceitáveis sem o recurso a esforço computacional excessivo. Assim, este algoritmo revela-se muito eficaz, com uma tolerância relativamente alta, para este tipo de exemplos. Por outro lado, verifica-se que a utilização do M.E.F.M. gera resultados semelhantes, mas através do recurso a um esforço computacional extremamente elevado e, portanto, desnecessário. Esta discrepância deve-se, principalmente, à maior complexidade formal do algoritmo.

6.3 – Aplicação dos Métodos Numéricos a Sistemas de P.D.E.´s

6.3.1 - Exemplo 6: Propagação de uma Chama

O presente modelo[29,91,116] simula a dinâmica de propagação de uma chama, considerando a sua temperatura e a massa do respectivo combustível. O problema caracteriza-se pela definição, nos tempos iniciais, de uma condição fronteira à direita, dependente do tempo e em forma de rampa. Esta condição pretende descrever a evolução da temperatura desde a ignição até atingir o valor máximo. A solução resultante apresenta a forma de duas ondas que se propagam muito rapidamente no mesmo sentido (neste caso, na direcção negativa de z), devido à relação directa que é estabelecida entre o consumo de combustível e o aumento de temperatura, originado pela libertação de calor que daí advém.

$$\frac{\delta u}{\delta t} = \frac{\delta^2 u}{\delta z^2} - 3.52\times10^6 \cdot u \cdot e^{-\frac{4}{v}} \tag{6.26}$$

$$\frac{\delta v}{\delta t} = \frac{\delta^2 v}{\delta z^2} + 3.52\times10^6 \cdot u \cdot e^{-\frac{4}{v}} \tag{6.27}$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = 0 \tag{6.28}$$

$$\frac{\delta v(0,t)}{\delta z} = 0 \tag{6.29}$$

$$\frac{\delta u(1,t)}{\delta z} = 0 \tag{6.30}$$

$$v(1,t) = 0.2 + \frac{t}{0.0002} \quad \text{se} \quad t \le 0.0002 \tag{6.31}$$

$$v(1,t) = 1.2 \quad \text{se} \quad t > 0.0002 \tag{6.32}$$

Condições Iniciais

$$u(z,0) = 1 \tag{6.33}$$

$$v(z,0) = 0.2 \tag{6.34}$$

Domínio

$$0 \le z \le 1 \tag{6.35}$$

$$0 \le t \le 0.006 \tag{6.36}$$

Parâmetros do Modelo

u - massa de combustível.
v - temperatura da chama.

z - variável espacial.
t - variável temporal.

Parâmetros da Execução –

Tipo de Diferenças Finitas	–	5 Pontos Centradas	***(Para as duas variáveis)***
Tolerância do Método	–	1×10^{-2}	***(Para as duas variáveis)***
Tolerância do Integrador	–	1×10^{-6}	
Grelha Inicial	–	Uniforme	
Tipo de Interpolação	–	Linear	
Passo de Tempo Base	–	1.2×10^{-3}	
Tempo Final	–	6.0×10^{-3}	

Neste exemplo, opta-se por utilizar uma malha uniforme e interpolações lineares, já que, como se concluiu para outros casos, estas são as condições que permitem a obtenção de melhores resultados em modelos cuja solução é caracterizada pela rápida movimentação de frentes de carácter abrupto, que percorrem a totalidade do domínio espacial.

Tabela 6.6: Condições fixadas para a execução dos *runs* do modelo de propagação de uma chama (M. Ref.).

Run	***Nº Pontos da Grelha Base (Nível 1)***	***tcpu (s) (Tfinal = 0.0060)***
1	21	526.7
2	41	911.2

Nos Gráficos 6.31 e 6.32 apresentam-se os resultados obtidos através da execução *run1* caracterizada pelo uso de uma malha base de nível 1 uniforme com 21 nodos.

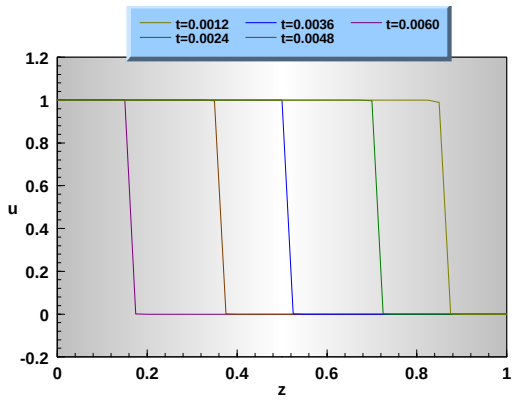

Gráfico 6.31: Resultados obtidos pelo método de refinamento para o *run1* do exemplo 6 (variável u).

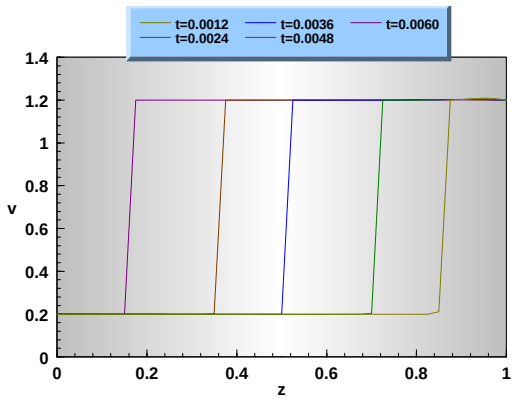

Gráfico 6.32: Resultados obtidos pelo método de refinamento para o *run1* do exemplo 6 (variável v).

A análise dos gráficos anteriores permite, desde já, identificar duas imprecisões graves:

⇨ A velocidade das ondas depende do passo temporal base fixado. Nos passos temporais [0.0024,0.0036] e [0.0048,0.0060] as ondas percorrem um espaço maior do que o correspondente aos restantes. Tal acontece porque o algoritmo é obrigado a dividir ao meio o passo de tempo base, devido a limitações numéricas do integrador.

⇨ A espessura das frentes é praticamente nula, sendo apenas limitada pelo passo espacial na malha de nível 2.

Na globalidade, a posição final das ondas encontra-se demasiada atrasada em relação ao esperado. Portanto, neste caso, verifica-se a competição entre dois efeitos que introduzem erros significativos nos resultados:

❶ Atraso das ondas provocado por erros introduzidos devido à excessiva esparsidade da malha de nível 1;

❷ Avanço artificial da onda originado pelo facto de nas operações de refinamento se simular o efeito da perturbação provocada pelo movimento das ondas, através da consideração de condições fronteira de *Dirichlet*. Deste modo, antecipa-se o efeito da passagem das ondas na resolução de cada subproblema de refinamento.

É possível visualizar o efeito ❷ se se considerar os gráficos de refinamento correspondentes. Assim, verifica-se que para níveis de refinamento crescentes, ocorre um nítido deslocamento para a esquerda no subdomínio de nível 3 inicialmente seleccionado (vd. Gráfico 6.33). Deste modo, obtêm-se ondas incorrectas de declive praticamente infinito que são empurradas artificialmente no sentido do seu movimento, em cada passo de refinamento. Deste modo, se se efectuasse a execução com um passo temporal base reduzido por um factor dois ($\Delta t=0.0006$), as ondas atingiam a fronteira esquerda, exactamente no tempo final t=0.006, o que constitui um comportamento incorrecto (resultados não apresentados).

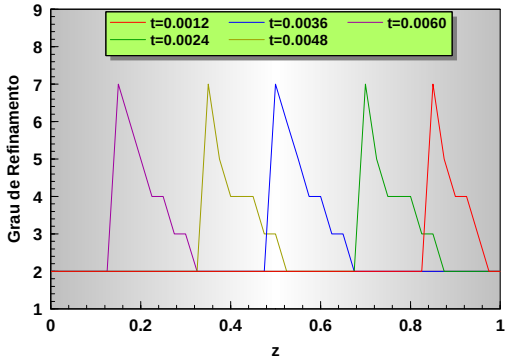

Gráfico 6.33: Resultados do nível de refinamento para o *run1* do exemplo 6.

Nas condições do *run2*, utiliza-se uma malha base mais apertada, verificando-se que as condicionantes baseadas em limitações numéricas se tornam menos importantes. Neste caso, a evolução das ondas é semelhante à esperada, sendo a velocidade de propagação das ondas constante e a sua posição final correcta (vd. Gráficos 6.34 e 6.35). Deste modo, a

ocorrência de erros de tipo ❶ referidos anteriormente é bastante menos importante. No entanto, o deslocamento forçado das ondas para a esquerda continua a verificar-se, já que os perfis de refinamento mantêm-se deslocados nessa direcção (vd. Gráfico 6.36). De facto, verifica-se igualmente neste caso, que, para passos temporais mais reduzidos, a velocidade das frentes se torna demasiado elevada. Este é um problema intrínseco da própria formulação do algoritmo, e apenas pode ser resolvido através da aplicação de estratégias alternativas à aproximação de *Dirichlet* considerada.

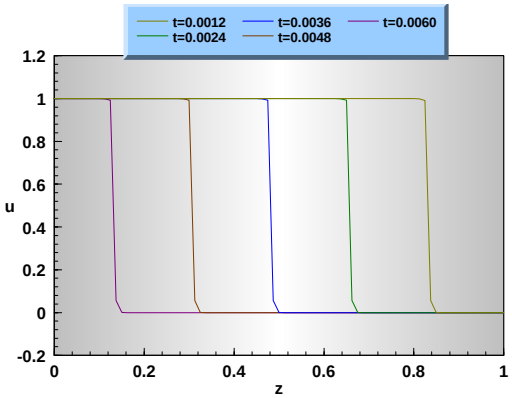

Gráfico 6.34: Resultados obtidos pelo método de refinamento para o *run2* do exemplo 6 (variável u).

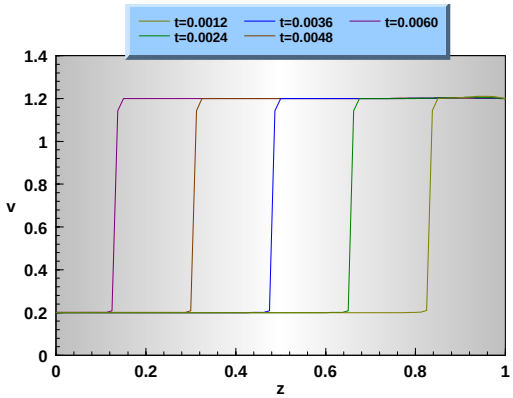

Gráfico 6.35: Resultados obtidos pelo método de refinamento para o *run2* do exemplo 6 (variável v).

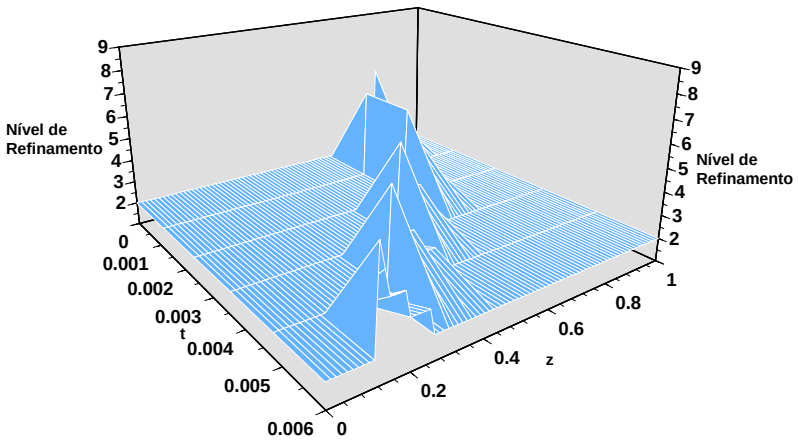

Gráfico 6.36: Resultados do nível de refinamento para o *run2* do exemplo 6.

No entanto, os resultados obtidos com passos temporais da ordem de $\Delta t=0.0012$ são semelhantes aos apresentados por *Duarte*[29], mas verifica-se que a velocidade de propagação das ondas continua a depender do respectivo passo. Por outro lado, o declive das ondas formadas mantem-se demasiado elevado em relação ao esperado.

Este exemplo demonstra que a estratégia escolhida para o tratamento das condições fronteira dos subproblemas de refinamento (que, de facto não é mais do que uma simples aproximação) não é a mais adequada, podendo originar problemas de precisão dos resultados e de performance do algoritmo em casos específicos.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas ***(Para as duas variáveis)***
- **Tolerância do Método** – 5×10^{-3} ***(Para as duas variáveis)***
- **Tolerância do Integrador** – 1×10^{-6}
- **Grelha Inicial** – Não Uniforme, 20 Nodos; z=[0.0, 0.2, 0.3, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.825, 0.8, 0.85, 0.875, 0.9, 0.925, 0.95, 0.975, 1.0]
- **Tipo de Interpolação** – Linear
- Δz_{MIN} – 1×10^{-4}
- Δz_{MAX} – 1×10^{-2}
- α – 1
- λ – 0.6
- **Passo de Tempo Base** – 1×10^{-4}
- **Tempo Final** – 6×10^{-3}
- **Tempo de computação (T_{cpu})** – 12788.6 s

A aplicação do método de malha móvel adaptativa ao presente exemplo conduziu à obtenção dos resultados apresentados nos Gráficos 6.37 e 6.38. Pode-se observar uma reprodução correcta das ondas e do seu movimento, sendo os resultados similares aos obtidos por *Duarte*[29]. Nota-se a ocorrência de um ligeiro *overshoot* nos perfis de v para tempos iniciais devido a problemas numéricos (vd. Gráfico 6.38), característica que se verifica igualmente nos resultados apresentados por *Petzold*[91]. O método demonstra-se robusto com a utilização de condições fronteira dependentes linearmente no tempo, que se revelam muito mais precisas do que as constantes utilizadas no método de refinamento. No entanto, este aumento de precisão provoca um acréscimo considerável nos tempos de cpu associados a cada método.

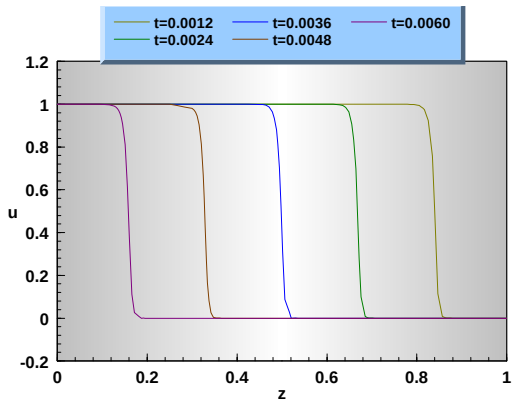

Gráfico 6.37: Resultados obtidos pelo método de malha móvel para o *run1* do exemplo 6 (variável u).

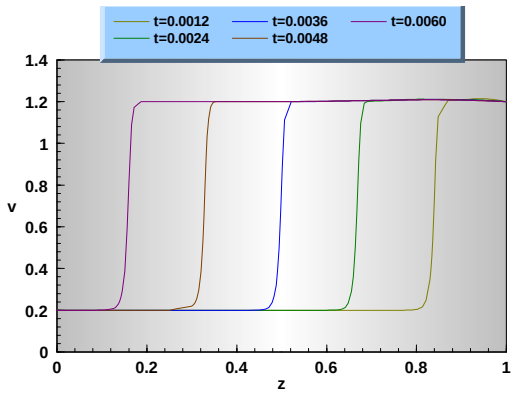

Gráfico 6.38: Resultados obtidos pelo método de malha móvel para o *run1* do exemplo 6 (variável v).

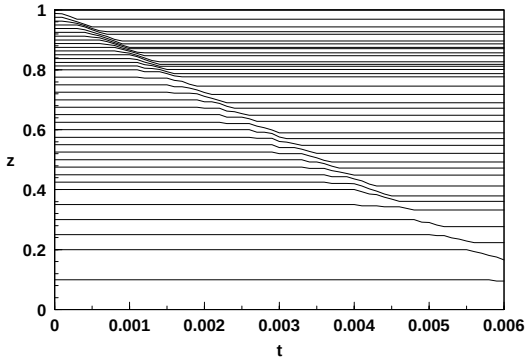

Gráfico 6.39: Evolução da malha inicial de nível 2 para o *run1* do exemplo 6 (variável u).

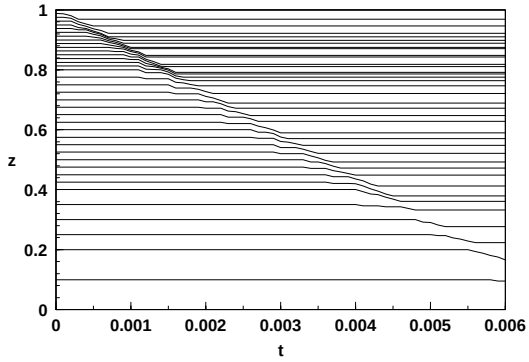

Gráfico 6.40: Evolução da malha inicial de nível 2 para o *run1* do exemplo 6 (variável v).

Através da análise da evolução dos nodos das malhas base de nível 2 referentes a cada variável (vd. Gráficos 6.39 e 6.40), verifica-se que estas acompanham de forma eficaz o deslocamento das respectivas ondas. Como este movimento é semelhante, então, cada uma das malhas sofre uma evolução análoga, apesar de não serem coincidentes, o que contribui fortemente para um aumento drástico do esforço computacional, já que, em cada passo, o algoritmo necessita de executar duas integrações distintas para cada uma das malhas geradas. Por outro lado e de forma a manter a estabilidade das secções constantes dos perfis, torna-se necessário introduzir um número considerável de nodos intermédios adicionais, à medida que as ondas percorrem o domínio. Assim, o peso da operação de avaliação do erro espacial torna-se cada vez mais determinante no esforço computacional global do algoritmo, devido à crescente complexidade das malhas fixas definidas.

De forma a se analisar a influência dos parâmetros: λ e Δz_{MAX} (que se revelaram críticos na execução de exemplos anteriores), na performance do método quando aplicado ao presente modelo, foram considerados vários *runs*, cujas condições se encontram resumidas na Tabela 6.7. Os resultados apresentados acima dizem respeito ao *run1*, mantendo-se todos os parâmetros restantes inalterados.

Tabela 6.7: Condições fixadas para a execução dos *runs* do modelo de propagação de uma chama (M. M. Móv.).

Run	Δz_{MAX}	λ	***tcpu (s) (Tfinal = 0.0060)***
1	0.01	0.6	12788.6
2	0.01	0.75	13079.1
3	0.02	0.6	6180.4
4	0.01	0.5	14165.7

Os perfis obtidos em cada um dos *runs*, para os tempos t=0.0012 e t=0.0060 são apresentados nos Gráficos 6.41 a 6.44. Verifica-se que em relação à variável u, não ocorrem problemas na formação das ondas. No entanto, para Δz_{MAX}=0.02 (*run3*), nota-se uma certa dificuldade na formação da curvatura da secção inferior da frente (vd. Gráficos 6.41 e 6.43), já que os pontos adjacentes que a deveriam reproduzir se afastam demasiado. No que diz respeito à variável v, verifica-se a formação de um ligeiro *overshoot* para tempos iniciais (vd. Gráfico 6.42). Observa-se, igualmente, problemas na definição da curvatura a montante da frente para o *run3*. Por outro lado, é visível em todos os gráficos, um atraso razoável das ondas referentes a esse *run*, relativamente às correspondentes aos restantes casos, que se mantém aproximadamente constante ao longo do tempo. As frentes associadas a cada um dos

restantes *runs* situam-se mais ou menos em posições similares. No caso de λ=0.5 (*run4*), observa-se uma ligeira dificuldade em manter a secção superior da onda exactamente em v=1.2, enquanto que para λ=0.75 (*run2*), se verifica um aumento da amplitude do *overshoot* inicial, que posteriormente desaparece com o decorrer da integração (vd. Gráficos 6.42 e 6.44).

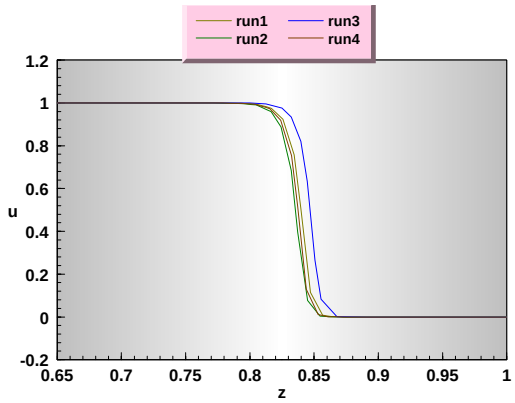

Gráfico 6.41: Resultados obtidos pelo método da malha móvel para cada *run* (t=0.0012, var. u).

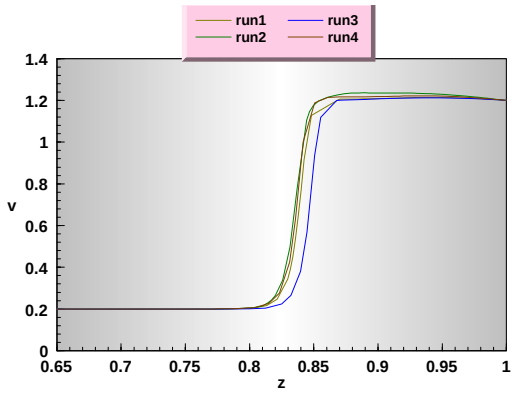

Gráfico 6.42: Resultados obtidos pelo método da malha móvel para cada *run* (t=0.0012, var. v).

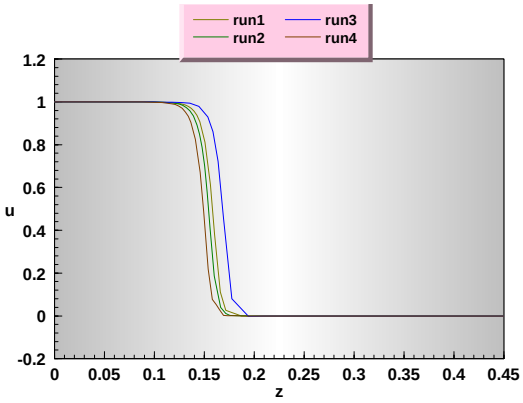

Gráfico 6.43: Resultados obtidos pelo método da malha móvel para cada *run* (t=0.0060, var. u).

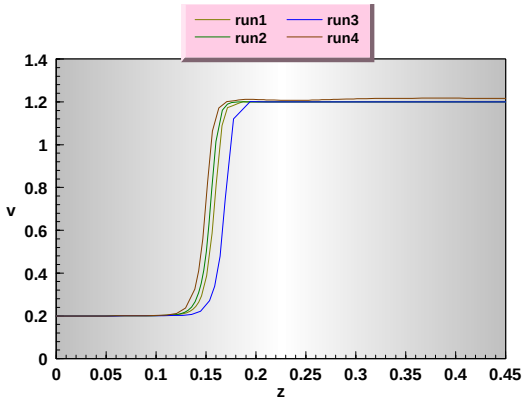

Gráfico 6.44: Resultados obtidos pelo método da malha móvel para cada *run* (t=0.0060, var. v).

Desta forma conclui-se que o factor de viscosidade nodal ideal para este modelo é aproximadamente 0.6. Para além disso, constata-se que a utilização de um espaçamento máximo internodal mais elevado que 0.01, introduz dificuldades na definição da forma das frentes abruptas e do seu movimento. No entanto, o aumento deste factor contribui de forma importante para a diminuição do esforço computacional relativo à execução do algoritmo, devido à redução na frequência de introdução de nodos adicionais. Assim, o tempo de cpu referente ao *run3* é consideravelmente mais baixo do que os restantes. No que diz respeito a esses *runs*, os tempos de computação são semelhantes, sendo o mais baixo referente à utilização de λ=0.6 (*run1*). Assim, verifica-se que, para além deste *run* possibilitar resultados gerais de melhor qualidade do ponto de vista da sua precisão, também exige o menor trabalho computacional.

Nos Gráficos 6.45 a 6.48 apresentam-se os perfis calculados nas condições do *run3*. Constata-se que o comportamento da solução é semelhante ao verificado para o *run1*. No entanto, as dificuldades na formação da curvatura a montante das frentes são visíveis (vd. Gráficos 6.45 e 6.46).

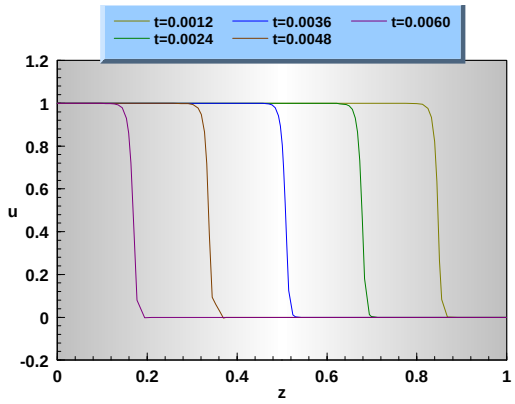

Gráfico 6.45: Resultados obtidos pelo método de malha móvel para o *run3* do exemplo 6 (variável u).

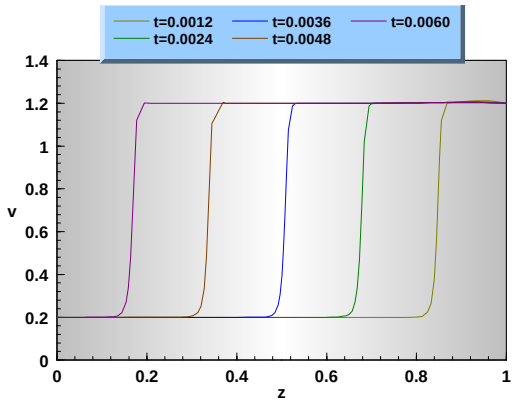

Gráfico 6.46: Resultados obtidos pelo método de malha móvel para o *run3* do exemplo 6 (variável v).

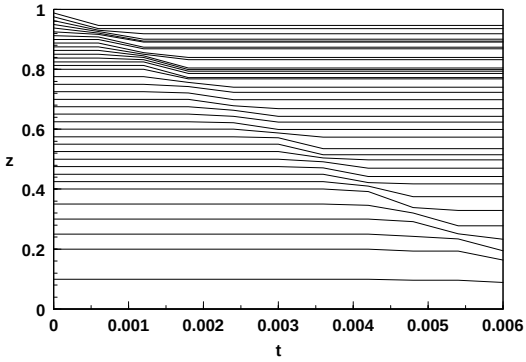

Gráfico 6.47: Evolução da malha inicial de nível 2 para o *run3* do exemplo 6 (variável u).

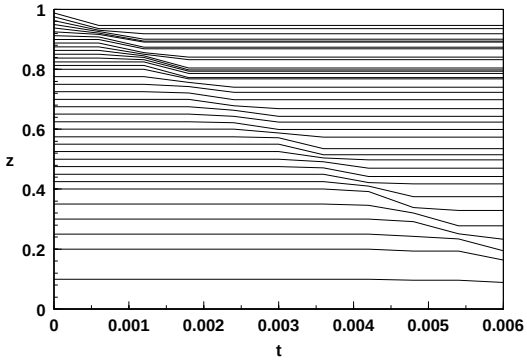

Gráfico 6.48: Evolução da malha inicial de nível 2 para o *run3* do exemplo 6 (variável v).

Devido ao excessivo Δz_{MAX}, em relação à velocidade imprimida no deslocamento nodal, verifica-se um afastamento demasiado elevado dos pontos que deveriam definir a curva da onda, o que conduz ao desenvolvimento de anomalias nos perfis. Os nodos acompanham correctamente o movimento de deslocamento das ondas na direcção negativa das abcissas, a uma velocidade praticamente constante (vd. Gráficos 6.47 e 6.48).

O facto de se permitir uma maior liberdade no afastamento de nodos consecutivos, permite que a introdução de nodos intermédios seja menos frequente. Deste modo, as malhas globais definidas são mais leves do que no caso 1, conduzindo a um tempo de computação consideravelmente inferior. No entanto, essa simplificação no avanço da integração provoca uma diminuição nítida na qualidade dos resultados obtidos.

Os desempenhos computacionais, associados a cada um dos métodos considerados, estão resumidos na tabela seguinte:

Tabela 6.8: Desempenhos computacionais para o modelo de propagação de uma chama.

Algoritmo	***Tempo de Computação (s)***
Refinamento	911.2
Malha Móvel	12788.6
M.E.F.M.[29]	8226.9

Por análise da Tabela 6.8, constata-se que o método de refinamento desenvolve um esforço computacional muito inferior do que os restantes métodos. No entanto, levando em conta as severas limitações verificadas na aplicação deste método a este exemplo, conclui-se que este não se revela adequado e eficaz na resolução do modelo em estudo.

Os resultados obtidos por aplicação do método de malha móvel adaptativa são similares aos apresentados por *Duarte*[29], mas exigem a utilização de tempos de computação mais elevados. Deste modo, o M.E.F.M. revela-se relativamente mais eficiente na execução do presente modelo, apesar dos resultados conseguidos através da malha adaptativa serem já bastante satisfatórios.

6.3.2 - Exemplo 7: Reactor Tubular Não Isotérmico

Este exemplo[29,95] descreve um reactor tubular catalítico não isotérmico. O modelo considerado designa-se por pseudo-homogéneo unidimensional. Desta forma, os balanços mássicos e energéticos efectuados consideram simultaneamente as fases sólida e fluida do reactor. Por outro lado, admite-se que apenas a difusão axial do fluido é importante. No reactor ocorre uma reacção de primeira ordem $A \rightarrow B$.

$$\frac{\delta u}{\delta t} = \frac{1}{Pe} \cdot \frac{\delta^2 u}{\delta z^2} - \frac{\delta u}{\delta z} - Da \cdot u \cdot v \cdot e^{-\gamma \cdot \left(\frac{1}{v} - 1\right)} \tag{6.37}$$

$$\frac{\delta v}{\delta t} = -\frac{\tau}{\tau_{hl}} \cdot \frac{\delta v}{\delta z} + \frac{\tau}{\tau_{hl}} \cdot \beta \cdot Da \cdot u \cdot v \cdot e^{-\gamma \cdot \left(\frac{1}{v} - 1\right)} - \frac{\tau}{\tau_{hl}} \cdot N_{wh} \cdot \left(v - v_w\right) \tag{6.38}$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = Pe \cdot (u - 1) \tag{6.39}$$

$$v(0,t) = 1 \tag{6.40}$$

$$\frac{\delta u(1,t)}{\delta z} = 0 \tag{6.41}$$

Condições Iniciais

$$u(z,0) = 0 \tag{6.42}$$

$$v(z,0) = 1 \tag{6.43}$$

Domínio

$$0 \le z \le 1 \tag{6.44}$$

$$0 \le t \le 1000 \tag{6.45}$$

Parâmetros do Modelo

$u = \frac{C}{C_0}$ - concentração normalizada em relação à concentração da alimentação.

C - concentração do componente A.

C_0 - concentração do componente A na corrente de entrada.

$v = \frac{T}{T_0}$ - temperatura normalizada em relação à temperatura de entrada.

T - temperatura da mistura reaccional.

T_0 - temperatura da mistura reaccional à entrada do reactor.

$v_w = \frac{T_w}{T_{w0}} = 1$ - temperatura normalizada do fluido de arrefecimento em relação à sua temperatura de entrada.

T_w - temperatura do fluido de arrefecimento.

T_{w0} - temperatura do fluido de arrefecimento à entrada.

$z = \frac{\xi}{L}$ - variável espacial normalizada em relação ao comprimento do reactor.

ξ - variável espacial.

L - comprimento do reactor.

$t = \frac{\theta}{\tau}$ - tempo adimensional, normalizado em relação ao tempo de passagem.

θ - variável temporal.

$\tau = \frac{L}{U_i}$ - tempo de passagem de um elemento de fluido no leito.

U_i - velocidade do fluido no interior do reactor.

$Pe = \frac{U_i \cdot L}{D_{eff}} = 10^4$ - número de *Peclet* axial.

D_{eff} - difusividade axial efectiva.

$Da = \frac{k(T_0)}{\varepsilon} \cdot \tau = 0.7$ - número de *Damkhöler*.

$k(T_0)$ - constante de velocidade da reacção nas condições de entrada.

ε - porosidade do leito.

$\gamma = \frac{E_a}{R \cdot T_0} = 21.8$ - número de *Arrhenius*.

E_a - energia de activação da reacção.

R - constante dos gases perfeitos.

$\tau_{hl} = \frac{\varepsilon \cdot \rho_f \cdot C_{Pf} + (1-\varepsilon) \cdot \rho_s \cdot C_{Ps}}{\varepsilon \cdot \rho_f \cdot C_{Pf}}$ - tempo de passagem da onda térmica.

ρ_f - densidade do fluido.

ρ_s - densidade do sólido.

C_{Pf} - calor específico do fluido.

C_{Ps} - calor específico do sólido.

$\beta = \frac{(-\Delta H) \cdot C_0}{\rho_f \cdot C_{Pf} \cdot T_0} = 0.7$ - elevação adiabática de temperatura adimensional.

$(-\Delta H)$ - entalpia da reacção.

$N_{wh} = \dfrac{2 \cdot U \cdot \tau}{\varepsilon \cdot R_0 \cdot \rho_f \cdot C_{Pf}} = 33.7$ - número de unidades de transferência de calor na parede.

U - coeficiente de transferência de calor entre a parede e o fluido reaccional.

R_0 - raio do reactor.

$\dfrac{\tau}{\tau_{hl}} = 2.08 \times 10^{-4}$ - razão entre as velocidades da onda mássica e da onda térmica.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 P. *Biased Upwind* ***(Para as duas variáveis)***
- **Tolerância do Método** – **Var. 1:** 5×10^{-3} **(Z. 1)**; 5×10^{-3} **(Z. 2)**; 1×10^{-2} **(Z. 3)**
  Var. 2: 1×10^{-3} **(Z. 1)**; 5×10^{-3} **(Z. 2)**; 1×10^{-3} **(Z. 3)**
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 31 Nodos
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 0.01 **(Zona 1)** ; 0.5 **(Zona 2)** ; 1.0 **(Zona 3)**
- **Tempo Final** – 1000.0
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 19180.2 s

Zona 1: $t \in [0.0, 1.0[$
Zona 2: $t \in [1.0, 100.0[$
Zona 3: $t \in [100.0, 1000.0]$

As características da solução deste modelo variam substancialmente para períodos de tempo distintos. Inicialmente, devido à aplicação de uma perturbação em degrau em z=0, forma-se uma frente nos perfis da variável u, que se movimenta rapidamente ao longo do reactor, até embater na fronteira direita do domínio (z=1) no instante t=1. Nesse período, as variações na variável v são diminutas, já que a razão entre o avanço das ondas mássica e térmica (τ/τ_{hl}) é muito pequena. A partir desse instante, e após a passagem da onda mássica, os perfis de concentração (u) tornam-se muito suaves e evoluem de uma forma bastante lenta. Por outro lado, ocorre o avanço progressivo da onda térmica, podendo–se visualizar o movimento do *hotspot* no reactor, para tempos consideravelmente mais elevados, devido à alta capacitância térmica do sólido. A ocorrência do máximo no perfil de temperatura (v) é explicada pelo facto da maior parte do reagente reagir à entrada do reactor, provocando um aumento importante da temperatura devido à exotermicidade da reacção. Portanto, devido a estas consideráveis diferenças de comportamento dos perfis, optou-se por dividir o domínio temporal em três zonas, de forma a optimizar o desempenho do algoritmo em função das diversas características da solução. Desta forma, aplicaram-se passos temporais e tolerâncias diferentes, adequadas às características dos perfis em cada uma dessas zonas.

Nos Gráficos 6.49 a 6.52 são apresentados os resultados obtidos através da utilização dos parâmetros listados acima (*run1*). Verifica-se uma definição correcta dos perfis, apesar da espessura da onda mássica desenvolvida ser ligeiramente elevada em relação ao esperado (vd. Gráfico 6.49).

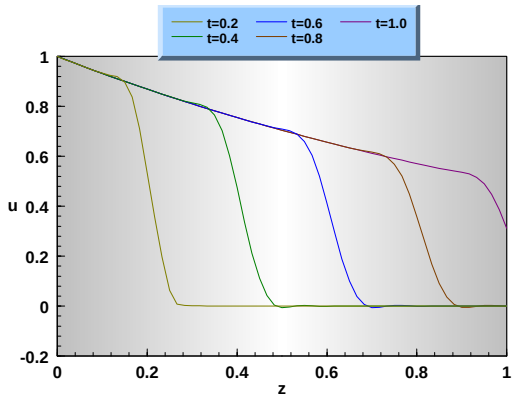

Gráfico 6.49: Resultados obtidos pelo refinamento para o *run1* do exemplo 7 (tf=1.0, var. u).

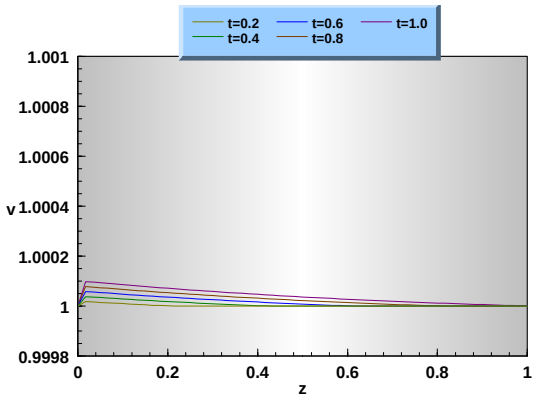

Gráfico 6.50: Resultados obtidos pelo refinamento para o *run1* do exemplo 7 (tf=1.0, var. v).

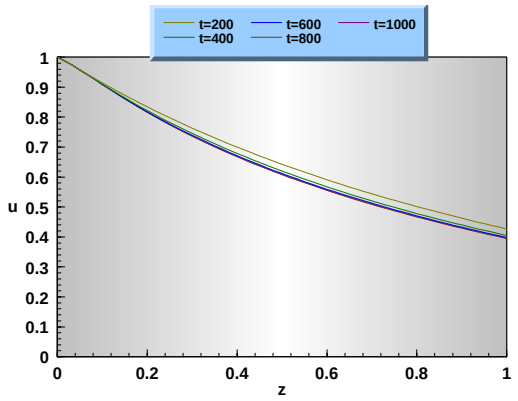

Gráfico 6.51: Resultados obtidos pelo refinamento para o *run1* do exemplo 7 (tf=1.000.0, var. u).

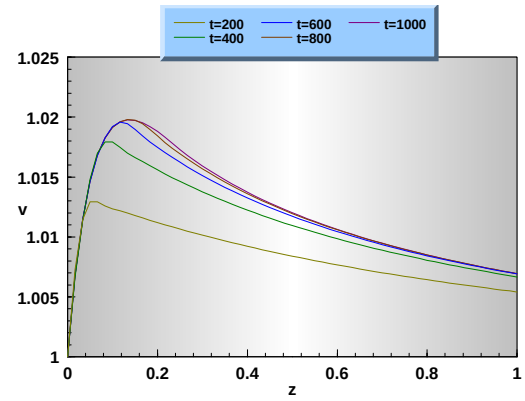

Gráfico 6.52: Resultados obtidos pelo refinamento para o *run1* do exemplo 7 (tf=1000.0, var. v).

Os perfis de refinamento associados ao *run1* encontram-se representados nos Gráficos 6.53 e 6.54 relativos a cada uma das fases da integração: propagação das ondas mássica e térmica, respectivamente.

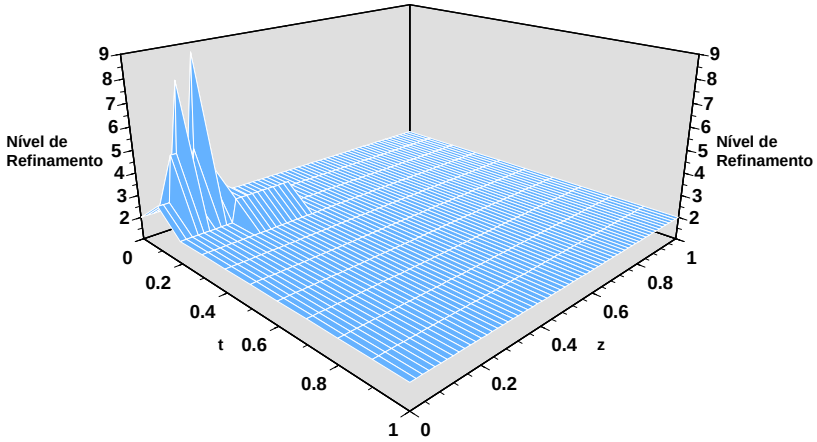

Gráfico 6.53: Resultados do nível de refinamento para o *run1* do exemplo 7 (tfinal=1.0).

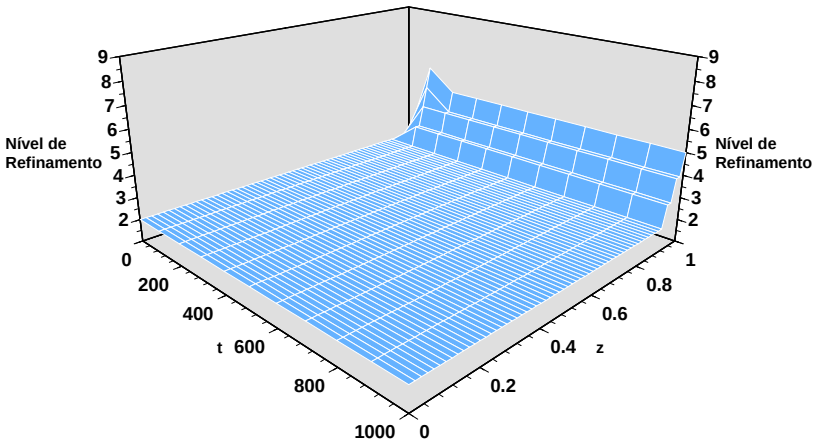

Gráfico 6.54: Resultados do nível de refinamento para o *run1* do exemplo 7 (tfinal=1000.0).

Observa-se a ocorrência de forte refinamento na fase inicial da integração de modo a que o algoritmo possa reproduzir o efeito da introdução da perturbação de arranque. No entanto, a partir dos instantes iniciais, a evolução da frente mássica assim formada não exige a utilização de refinamento significativo (vd. Gráfico 6.53). Verifica-se, igualmente, que para as Zonas 2 e 3 da integração, o método vê-se obrigado a executar constantes operações de refinamento moderado na vizinhança da fronteira direita (vd. Gráfico 6.54). Desta forma, conclui-se que o passo limitante para o avanço do algoritmo, após a passagem da onda mássica no reactor (t>1) é a avaliação da solução na zona adjacente a z=1.

Na globalidade, é possível concluir-se que, nas condições do *run1*, o algoritmo de refinamento possibilita bons resultados, com um esforço computacional razoável.

Na maioria dos exemplos referidos neste trabalho, obtiveram-se bons resultados através da aplicação de diferenças finitas centradas na discretização espacial. No entanto, para o exemplo presente, verifica-se que a utilização de diferenças centradas (*run2*) se revela insatisfatória, tanto do ponto de vista da qualidade dos resultados obtidos, como do esforço de computação exigido. As maiores dificuldades no avanço da integração ocorrem, ao contrário do que seria de esperar, durante o período de deslocamento da onda térmica (t>1), no qual os perfis são relativamente suaves. Como este período é bastante extenso, verifica-se que a ocorrência constante destes problemas de integração originam um aumento considerável do tempo de computação.

Nos Gráficos 6.55 e 6.56 são comparados os perfis de segundo nível para a variável u, obtidos nos primeiros dois passos de integração na zona temporal 2, para os instantes imediatamente posteriores ao choque da onda mássica com a fronteira direita do domínio espacial. Nota-se a ocorrência de forte instabilidade no perfil correspondente ao *run2* (diferenças centradas), gerada a partir de z=1. Este comportamento não seria de esperar, devido ao carácter suave dos perfis de u nesta fase da integração. As oscilações observadas obrigam ao refinamento de uma porção considerável do domínio (vd Gráficos 6.57 e 6.58), que se verifica ser necessário para todos os passos temporais até ao tempo final (vd. Gráfico 6.64). Como é óbvio, este tipo de comportamento origina um aumento drástico do tempo de computação associado ao *run2* (T_{cpu}=104839.0 s) em relação ao obtido nas condições do *run1* (T_{cpu}=19180.2 s). Pode-se concluir, igualmente, que este aumento do T_{cpu} é provocado pelas dificuldades de integração nas zonas 2 e 3, e não na zona 1, onde seria expectável a ocorrência de maiores problemas, devido ao carácter abrupto da frente mássica móvel. De facto, para esta zona: $T_{cpu\ run1}$=2084.8 s e $T_{cpu\ run2}$=3149.2 s. Apesar do tempo de computação para o *run2* ser maior do que o obtido para o *run1*, o seu aumento relativo é muito menor do que o verificado

entre os tempos de execução finais. Assim, os problemas observados são largamente ultrapassados através da utilização de diferenças atrasadas que acompanhem o movimento das ondas, e que, neste caso, se revelam consideravelmente mais eficientes.

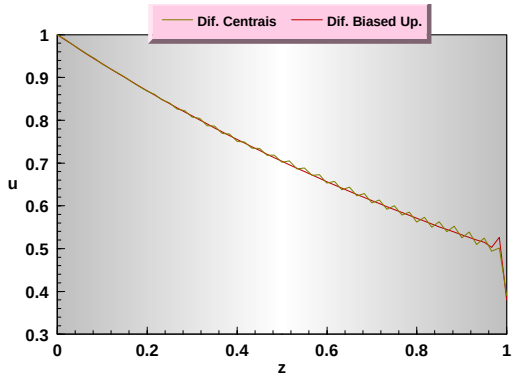

Gráfico 6.55: Comparação dos perfis de nível 2 obtidos para o *run1* e o *run2* do exemplo 7 (t=1.5).

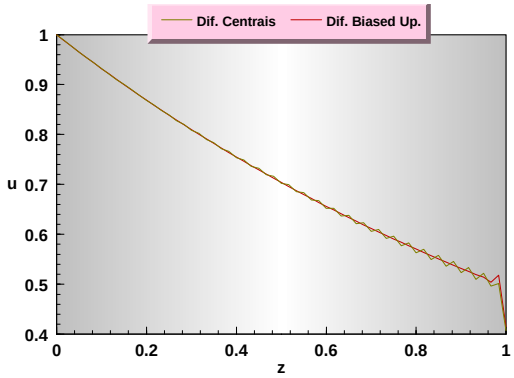

Gráfico 6.56: Comparação dos perfis de nível 2 obtidos para o *run1* e o *run2* do exemplo 7 (t=2.0).

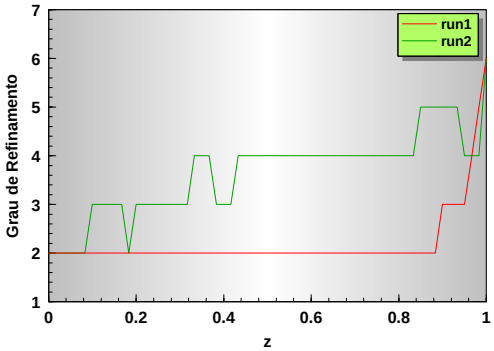

Gráfico 6.57: Comparação dos perfis de refinamento obtidos para o *run1* e o *run2* do exemplo 7 (t=1.5).

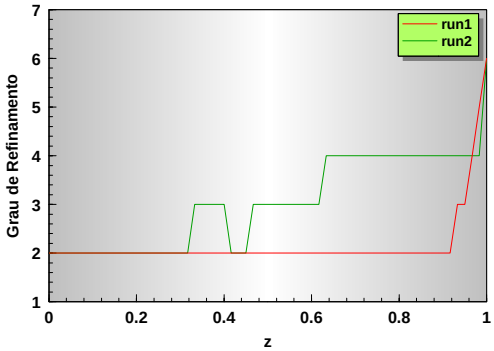

Gráfico 6.58: Comparação dos perfis de refinamento obtidos para o *run1* e o *run2* do exemplo 7 (t=2.0).

Para além das limitações referidas anteriormente, verifica-se que a discretização baseada em diferenças finitas centradas conduz a resultados de pior qualidade para a reprodução da evolução da onda mássica (vd. Gráfico 6.59). Assim, verifica-se as frentes formadas são algo distorcidas e apresentam espessuras mais elevadas do que no caso anterior. Para além disto, observa-se a ocorrência de oscilações razoáveis nos perfis. Durante este período, a evolução da temperatura é semelhante à observada no caso anterior (vd. Gráfico 6.60), embora as variações sejam quase imperceptíveis.

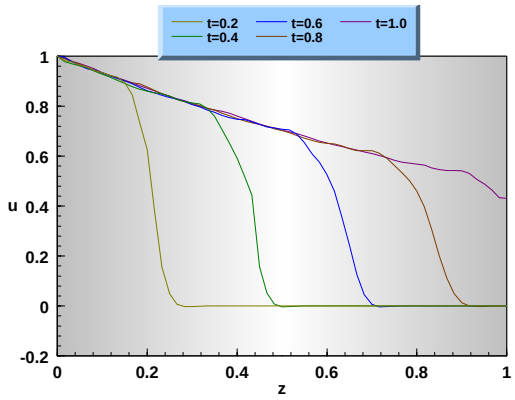

Gráfico 6.59: Resultados obtidos pelo refinamento para o *run2* do exemplo 7 (tf=1.0, var. u).

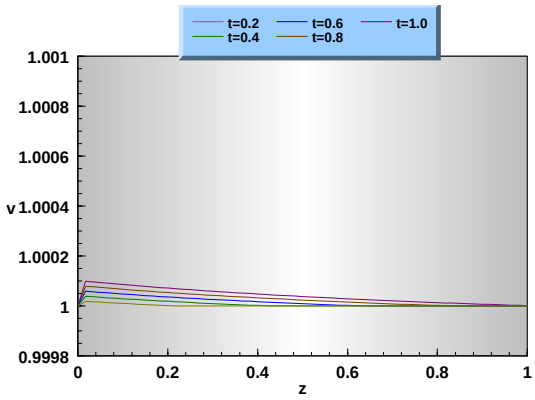

Gráfico 6.60: Resultados obtidos pelo refinamento para o *run2* do exemplo 7 (tf=1.0, var. v).

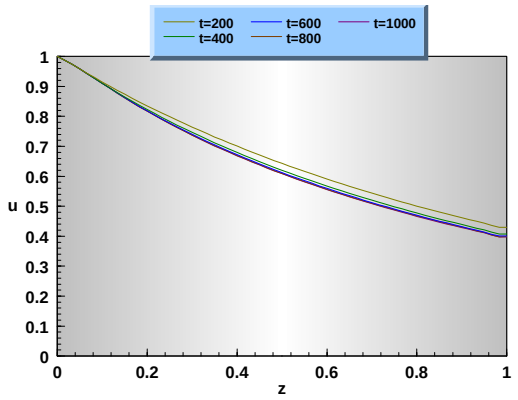

Gráfico 6.61: Resultados obtidos pelo refinamento para o *run2* do exemplo 7 (tf=1000.0, var. u).

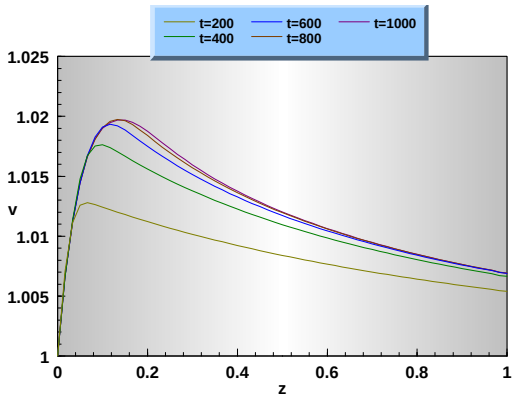

Gráfico 6.62: Resultados obtidos pelo refinamento para o *run2* do exemplo 7 (tf=1000.0, var. v).

A partir do instante t=1, o comportamento dos perfis é o correcto (vd. Gráficos 6.61 e 6.62), apesar de ser necessário um forte refinamento da solução, provocado pela instabilidade gerada na fronteira direita, que origina a reintegração de largas porções do domínio (vd. Gráfico 6.64). Verifica-se, também, uma maior actividade do refinamento na zona 1, relativamente ao caso anterior (vd. Gráfico 6.63). A conjugação destes dois factores leva à exigência de um esforço computacional incomportável e irrealista para a execução do *run2* e, consequentemente, a tempos de computação muito elevados.

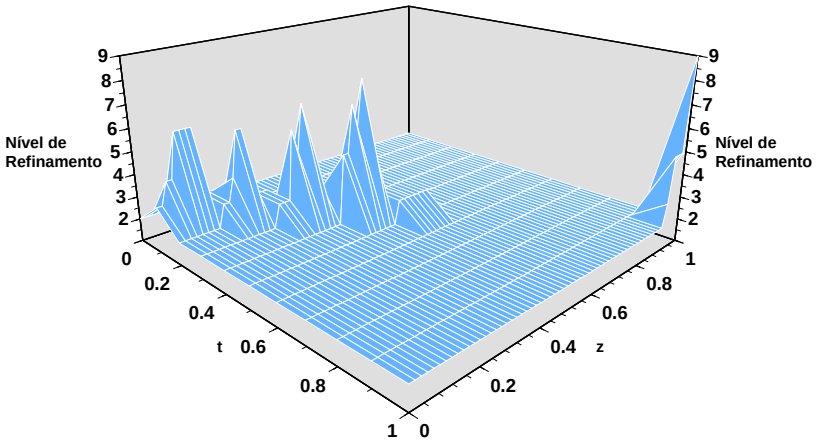

Gráfico 6.63: Resultados do nível de refinamento para o *run2* do exemplo 7 (tfinal=1.0).

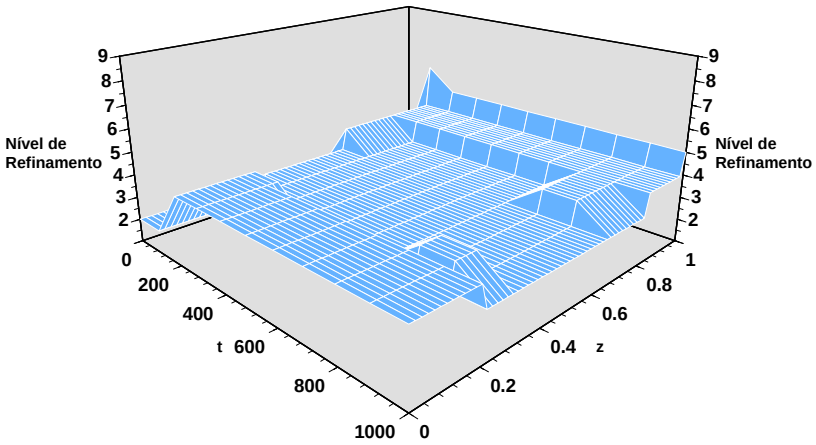

Gráfico 6.64: Resultados do nível de refinamento para o *run2* do exemplo 7 (tfinal=1000.0).

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 P. *Biased Upwind* ***(Para as duas variáveis)***
- **Tolerância do Método** – **Var. 1:** 5×10^{-3} **(Z. 1)**; 5×10^{-3} **(Z. 2)**; 1×10^{-2} **(Z. 3)**
  Var. 2: 1×10^{-3} **(Z. 1)**; 5×10^{-3} **(Z. 2)**; 1×10^{-3} **(Z. 3)**
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Não Uniforme, 28 Nodos; z=[0.0, 1×10^{-3}, 2×10^{-3}, 5×10^{-3}, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.875, 0.9, 0.925, 0.95, 0.975, 1.0]
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 0.01 **(Zona 1)** ; 0.5 **(Zona 2)** ; 1.0 **(Zona 3)**
- **Tempo Final** – 1000.0
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 16315.0 s

Zona 1: t ∈ [0.0, 1.0[
Zona 2: t ∈ [1.0, 100.0[
Zona 3: t ∈ [100.0, 1000.0]

Nos Gráficos 6.65 a 6.68, são apresentados os resultados obtidos pelo algoritmo através da utilização de uma malha não uniforme, caracterizada por uma maior concentração relativa de pontos junto às duas fronteiras espaciais. Assim, são introduzidos pontos próximo de z=0, de forma a ajudar o método a lidar melhor com as dificuldades criadas pela perturbação de arranque. A importância destes nodos é muito limitada, já que os efeitos do arranque apenas se fazem sentir nos instantes iniciais da integração, sendo este período muito reduzido em relação ao domínio temporal total. Por outro lado, são adicionados pontos junto à fronteira direita, de modo a melhorar o comportamento do algoritmo em face de possíveis dificuldades provocadas pela condição fronteira de *Neumann* nessa fronteira. Estes problemas ocorrem frequentemente para exemplos idênticos estudados neste trabalho, tendo sido já referidos no âmbito da análise efectuada a outros *runs* aplicados a este modelo.

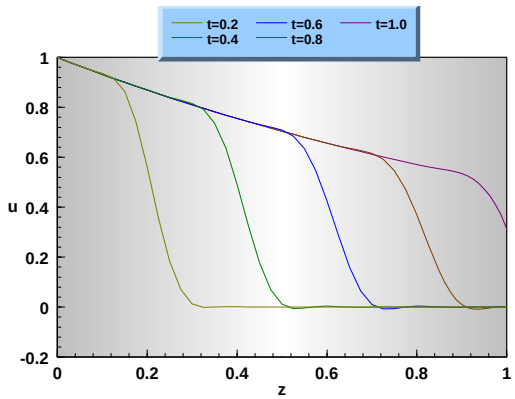

Gráfico 6.65: Resultados obtidos pelo refinamento para o *run3* do exemplo 7 (tf=1.0, var. u).

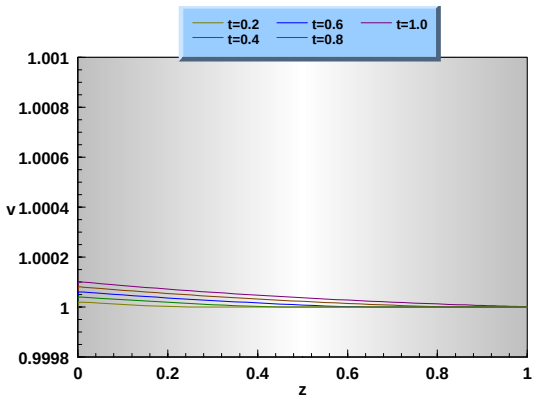

Gráfico 6.66: Resultados obtidos pelo refinamento para o *run3* do exemplo 7 (tf=1.0, var. v).

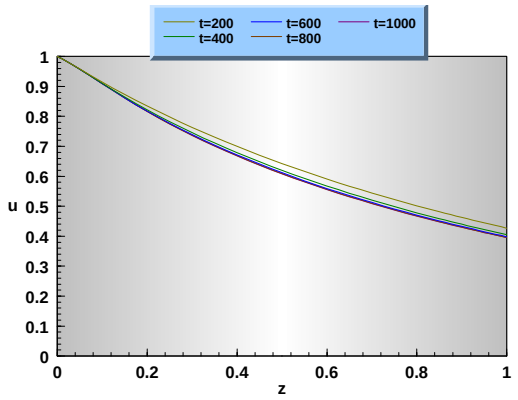

Gráfico 6.67: Resultados obtidos pelo refinamento para o *run3* do exemplo 7 (tf=1000.0, var. u).

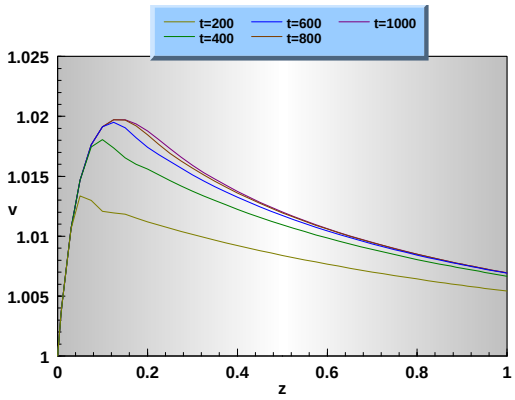

Gráfico 6.68: Resultados obtidos pelo refinamento para o *run3* do exemplo 7 (tf=1000.0, var. v).

Na globalidade, os resultados obtidos são muito semelhantes aos correspondentes ao *run1*, observando-se as diferenças seguintes:

❶ Os picos das curvas de temperatura encontram-se mais próximos da fronteira esquerda (z=0) para os instantes iniciais (vd. Gráfico 6.66). Esta constatação é perfeitamente natural, já que neste caso se colocam nodos da malha base de nível 2 muito mais próximos da fronteira esquerda do que no caso do *run1*.

❷ Observam-se dificuldades nítidas na definição correcta da curvatura do *hotspot* nos perfis de temperatura para tempos intermédios (vd. Gráfico 6.68). Este problema desaparece com o decorrer da integração e deve-se, essencialmente, a um défice de concentração nodal da malha de nível 2 nessa região do domínio.

O comportamento dos perfis de refinamento é perfeitamente idêntico ao observado para o caso do *run1* (vd. Gráficos 6.69 e 6.70).

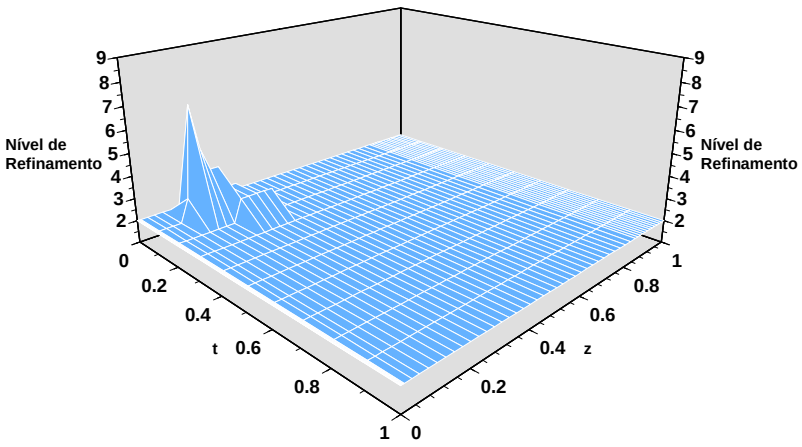

Gráfico 6.69: Resultados do nível de refinamento para o *run3* do exemplo 7 (tfinal=1.0).

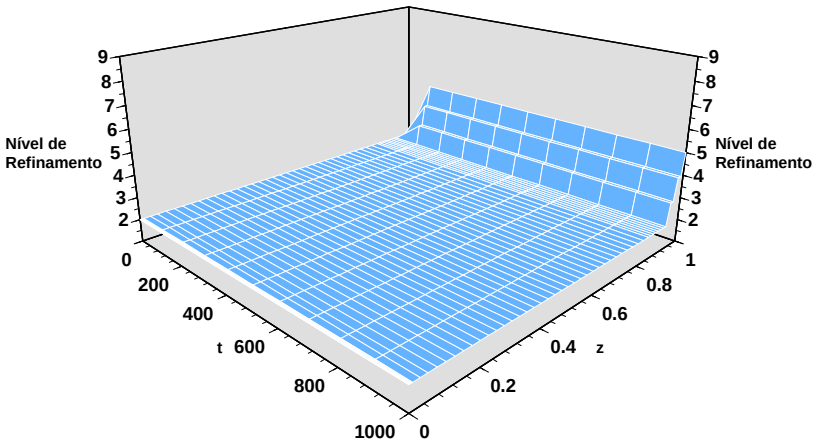

Gráfico 6.70: Resultados do nível de refinamento para o *run3* do exemplo 7 (tfinal=1000.0).

O tempo de computação é menor para esta execução, o que é normal, já que a malha base apresenta menos três nodos em relação ao *run1*. Assim, para malhas base mais leves, as dificuldades de integração diminuem e, consequentemente, o tempo de cpu baixa. No entanto, os resultados obtidos são na globalidade, de pior qualidade, apesar de serem, no entanto, bastante razoáveis.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas ***(Para as duas variáveis)***
- **Tolerância do Método** – **Var. 1:** 1×10^{-4} **(Z. 1)**; 5×10^{-3} **(Z. 2)**
  Var. 2: 5×10^{-3} **(Z. 1)**; 1×10^{-4} **(Z. 2)**
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Não Uniforme, 16 Nodos; z=[0.0, 2×10^{-4}, 5×10^{-4}, 1×10^{-3}, 1.5×10^{-3}, 2×10^{-3}, 5×10^{-3}, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0]
- **Tipo de Interpolação** – Linear
- Δz_{MIN} – 1×10^{-5}
- Δz_{MAX} – 1.3×10^{-2}
- α – 1
- λ – 0.75
- **Passo de Tempo Base** – 0.02 **(Zona 1)** ; 0.5 **(Zona 2)**
- **Tempo Final** – 10.0
- **Tempo de computação (T_{cpu})** – 12458.4 s

Zona 1: $t \in [0.0, 1.0[$
Zona 2: $t \in [1.0, 10.0]$

A aplicação do método de malha móvel adaptativa, nas condições enunciadas acima possibilita a obtenção de perfis de qualidade superior aos do método de refinamento, nomeadamente no que diz respeito à correcta definição dos declives abruptos da frente mássica (vd. Gráficos 6.71 e 6.72). No entanto, a diminuição da influência da dissipação

numérica observada neste caso, é realizada à custa de um aumento significativo do esforço computacional associado à execução. Deste modo, o tempo de cpu necessário para a integração completa da fase de propagação da onda mássica (Zona 1, T_{cpu}=7962.8 s) é consideravelmente mais elevado do que o correspondente aos *runs* efectuados por aplicação do método de refinamento. Por outro lado, verifica-se que a integração das zonas seguintes do domínio temporal (fase de propagação da onda térmica) é bastante morosa, visto que o algoritmo não prevê a hipótese de eliminação de nodos na malha base relativa à variável concentração (u), construída na integração da Zona 1. Como esta malha é bastante densa, optou-se por apenas efectuar a execução até um tempo final (t_{final}=10.0) bastante aquém dos considerados nos *runs* anteriores (t_{final}=1000.0), obtendo-se um tempo de computação já relativamente elevado para este caso: T_{cpu}=12458.4 s.

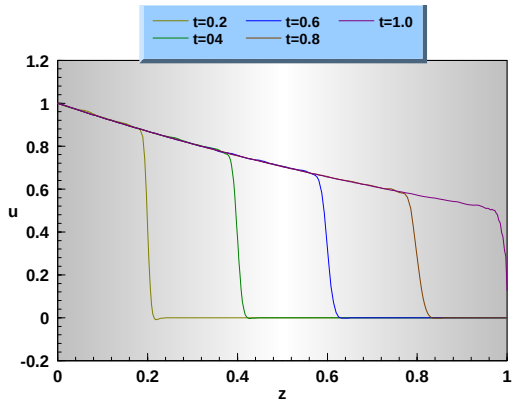

Gráfico 6.71: Resultados obtidos pelo método de malha móvel para o exemplo 7 (tf=1.0, var. u).

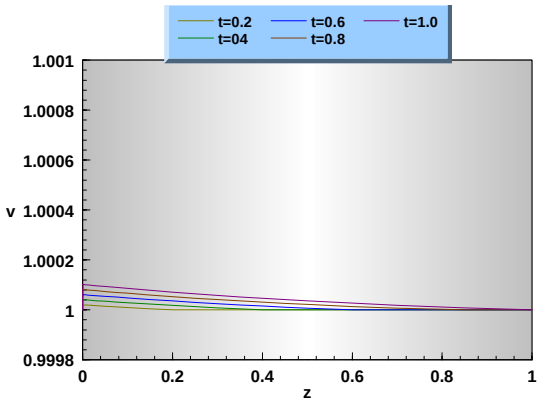

Gráfico 6.72: Resultados obtidos pelo método de malha móvel para o exemplo 7 (tf=1.0, var. v).

Por análise dos perfis de evolução das malhas base referentes a cada uma das variáveis, constatam-se dois comportamentos completamente díspares, que, no entanto são perfeitamente previsíveis. No caso da malha associada à variável concentração (u), verifica-se uma movimentação vigorosa dos nodos base, que acompanham a rápida propagação da frente mássica (vd. Gráfico 6.73). Por outro lado, devido à diminuta velocidade de deslocamento da frente térmica, a sua presença praticamente não se faz sentir nesta fase da integração. Deste modo, a malha base relacionada com a variável temperatura (v) não sofre qualquer deslocamento (vd. Gráfico 6.74).

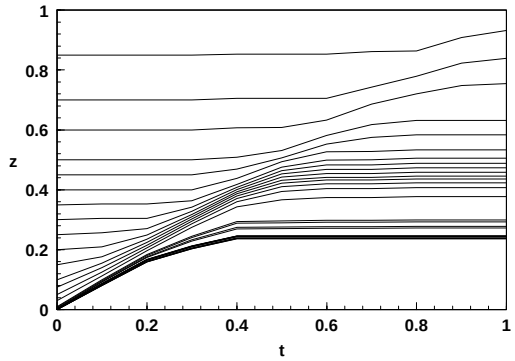

Gráfico 6.73: Evolução da malha inicial de nível 2 para o exemplo 7 (tf=1.0, variável u).

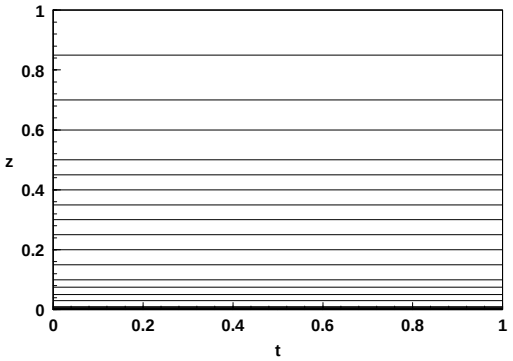

Gráfico 6.74: Evolução da malha inicial de nível 2 para o exemplo 7 (tf=1.0, variável v).

Apesar dos resultados obtidos serem bastante precisos, a sua obtenção exige um esforço computacional demasiado elevado. Deste modo, verifica-se que a presente formulação do método de malha móvel adaptativa não é muito adequado para a integração completa deste exemplo. No entanto, estes problemas poderão ser torneados através da introdução de procedimentos que possibilitem uma maior optimização do desempenho do algoritmo, nomeadamente um critério de remoção nodal.

A comparação entre o desempenho computacional do método de refinamento (referente às condições do *run1*) e o do método de M.E.F.M. utilizado por *Duarte*[29], é efectuada na tabela seguinte.

Tabela 6.9: Desempenhos computacionais para o modelo do reactor tubular não isotérmico.

Algoritmo	***Tempo de Computação (s)***
Refinamento	19180.2
M.E.F.M.[29]	10592.4

Verifica-se que, no geral, o M.E.F.M. se revela mais adequado para a execução deste modelo, já que é mais eficiente do ponto de vista computacional. A qualidade dos perfis obtidos através de cada método é semelhante, apesar dos resultados apresentados por *Duarte*[29] serem ligeiramente melhores, principalmente no que diz respeito à formação das frentes abruptas com uma espessura reduzida.

6.3.3 - Exemplo 8: Propagação de Ondas

O modelo[29,93,104,116] apresentado descreve a propagação de duas ondas em sentidos contrários. Cada uma das ondas 1 e 2 representa um pulso cossinusoidal, centrado inicialmente nas posições z = 0.3 e z = 0.7, respectivamente.

$$\frac{\delta u}{\delta t} = -\frac{\delta u}{\delta z} - 100 \cdot u \cdot v \qquad (6.46)$$

$$\frac{\delta v}{\delta t} = \frac{\delta v}{\delta z} - 100 \cdot u \cdot v \qquad (6.47)$$

Condições Fronteira

$$u(0,t) = 0 \qquad (6.48)$$

$$v(0,t) = 0 \qquad (6.49)$$

$$u(1,t) = 0 \qquad (6.50)$$

$$v(1,t) = 0 \qquad (6.51)$$

Condições Iniciais

$$u(z,0) = 0 \quad \text{se} \quad z \in [0\ ,\ 0.2[\text{ e } z \in\ ]0.4\ ,\ 1] \qquad (6.52)$$

$$u(z,0) = 0.5 \cdot [1 + \cos(10 \cdot \pi \cdot (z - 0.5))] \quad \text{se} \quad z \in [0.2\ ,\ 0.4] \qquad (6.53)$$

$$v(z,0) = 0 \qquad \text{se} \quad z \in [0\ ,\ 0.6[\ \text{e}\ z \in\ ]0.8\ ,\ 1] \qquad (6.54)$$

$$v(z,0) = 0.5 \cdot [1 + \cos(10 \cdot \pi \cdot (z - 0.5))] \qquad \text{se} \quad z \in [0.6\ ,\ 0.8] \qquad (6.55)$$

Domínio

$$0 \le z \le 1 \qquad (6.56)$$

$$0 \le t \le 0.5 \qquad (6.57)$$

Parâmetros do Modelo

u - variável associada à onda 1.
v - variável associada à onda 2.
z - variável espacial.
t - variável temporal.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas ***(Para as duas variáveis)***
- **Tolerância do Método** – 1×10^{-2} ***(Para as duas variáveis)***
- **Tolerância do Integrador** – 1×10^{-6}
- **Grelha Inicial** – Uniforme; 41 Nodos
- **Tipo de Interpolação** – *Splines* Cúbicas 5 Pontos
- **Passo de Tempo Base** – 1×10^{-2}
- **Tempo Final** – 0.50
- **Nº Máximo de Iterações** – 9
- **Tempo de computação (T_{cpu})** – 3189.7 s

Os resultados obtidos através da execução do *run1* (caracterizado pelos parâmetros anteriores) são apresentados nos Gráficos 6.75 a 6.77. Por análise desses perfis, verifica-se que as ondas encontram-se inicialmente separadas, deslocando-se posteriormente em sentidos opostos com velocidade próxima de um, devido ao decréscimo de importância do termo não-linear das equações.

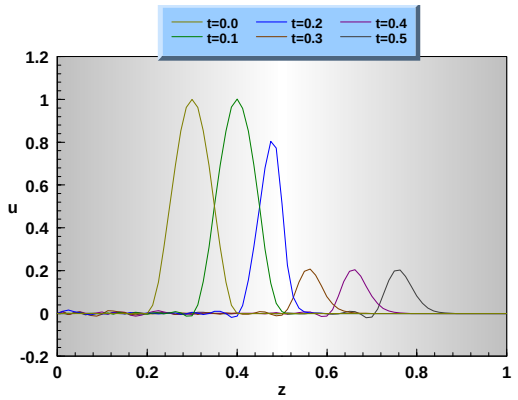

Gráfico 6.75: Resultados obtidos pelo refinamento para o *run1* do exemplo 8 (variável u).

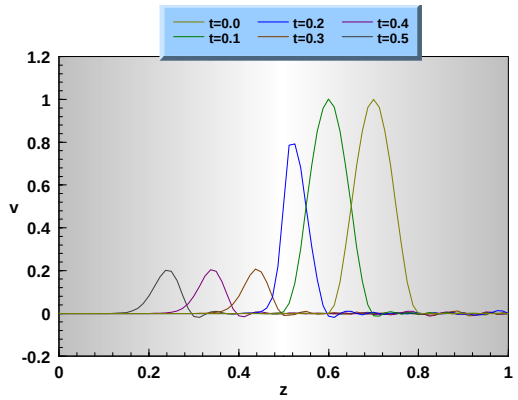

Gráfico 6.76: Resultados obtidos pelo refinamento para o *run1* do exemplo 8 (variável v).

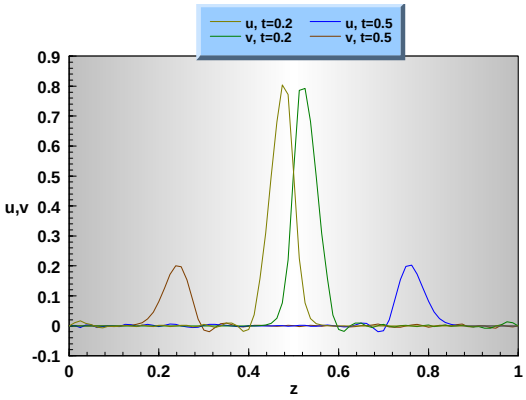

Gráfico 6.77: Comparação entre os perfis de cada variável para o *run1* do exemplo 8.

Por volta de t=0.15, as ondas colidem em z=0.5 e, como o termo não-linear se anula, verifica-se uma diminuição da amplitude das ondas e da sua velocidade de deslocamento. A colisão entre os pontos máximos de cada onda ocorre no instante t≅0.25. O algoritmo reproduz bem este comportamento, apesar de ser visível a introdução de instabilidade significativa nas regiões dos perfis situadas a montante de cada onda (vd. Gráficos 6.75 a 6.77).

As perturbações observadas são provocadas pela utilização de diferenças finitas centradas, que não permitem um acompanhamento satisfatório do movimento de cada onda. Por outro lado, verifica-se que, devido à instabilidade referida, ocorre refinamento junto à fronteira direita do domínio (vd. Gráfico 6.78). Tal comportamento não seria de esperar, já que os perfis correctos da solução na vizinhança de z=1, deveriam ser constantes e nulos.

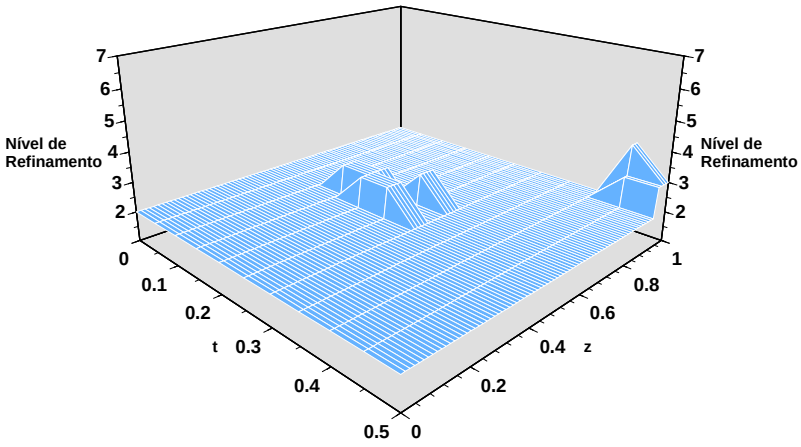

Gráfico 6.78: Resultados do nível de refinamento para o *run1* do exemplo 8.

Nas condições do *run1*, as dificuldades referidas introduzem a exigência de esforço computacional acrescido, obtendo-se perfis de solução insatisfatórios. De modo a ultrapassar estes problemas, foi executado um *run* adicional (*run2*), onde todas as condições foram mantidas, exceptuando o tipo de discretização espacial. Assim, considerou-se a utilização de diferenças finitas atrasadas para a variável u, cujo movimento é positivo, e diferenças adiantadas para a variável v, a qual apresenta deslocamento negativo:

Parâmetros da Execução –

⇘ **Tipo de Diferenças Finitas**	–	5 Pontos *Biased Upwind*	***(Variável 1)***
		5 Pontos *Biased Downwind*	***(Variável 2)***
⇘ **Tolerância do Método**	–	1×10^{-2}	***(Para as duas variáveis)***
⇘ **Tolerância do Integrador**	–	1×10^{-6}	
⇘ **Grelha Inicial**	–	Uniforme; 41 Nodos	
⇘ **Tipo de Interpolação**	–	*Splines* Cúbicas 5 Pontos	
⇘ **Passo de Tempo Base**	–	1×10^{-2}	
⇘ **Tempo Final**	–	0.50	
⇘ **Nº Máximo de Iterações**	–	9	
⇘ **Tempo de computação (T_{cpu})**	–	1985.6 s	

Por observação dos resultados obtidos nas condições do *run2* (vd. Gráficos 6.79 a 6.81), verifica-se que todas as dificuldades ocorridas no caso anterior foram satisfatoriamente torneadas. Neste caso, as discretizações acompanham o movimento das ondas, verificando-se completa ausência de oscilações nos perfis.

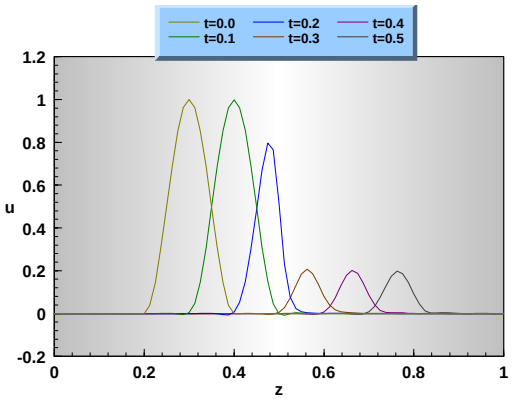

Gráfico 6.79: Resultados obtidos pelo refinamento para o *run2* do exemplo 8 (variável u).

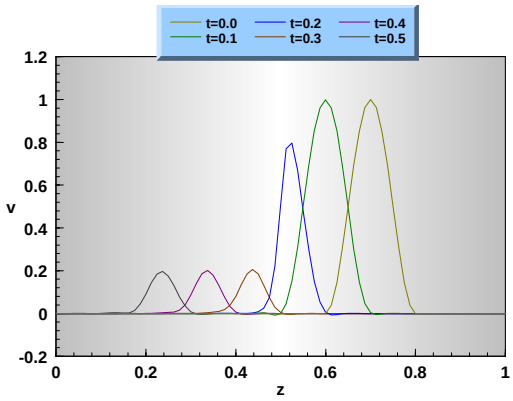

Gráfico 6.80: Resultados obtidos pelo refinamento para o *run2* do exemplo 8 (variável v).

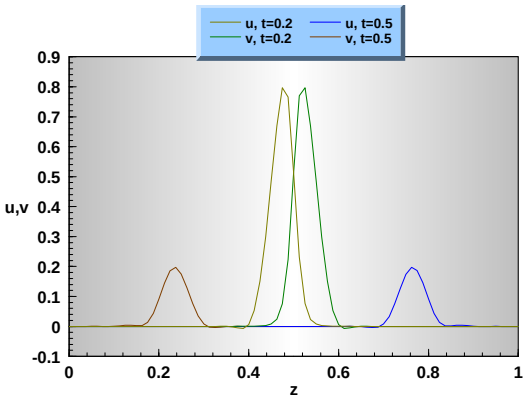

Gráfico 6.81: Comparação entre os perfis de cada variável para o *run2* do exemplo 8.

Por outro lado, verifica-se a necessidade de um refinamento bastante ligeiro, apenas para os instantes e nas zonas onde decorre o cruzamento das ondas (vd. Gráfico 6.82). Desta forma, o tempo de computação obtido é consideravelmente inferior ao do caso anterior.

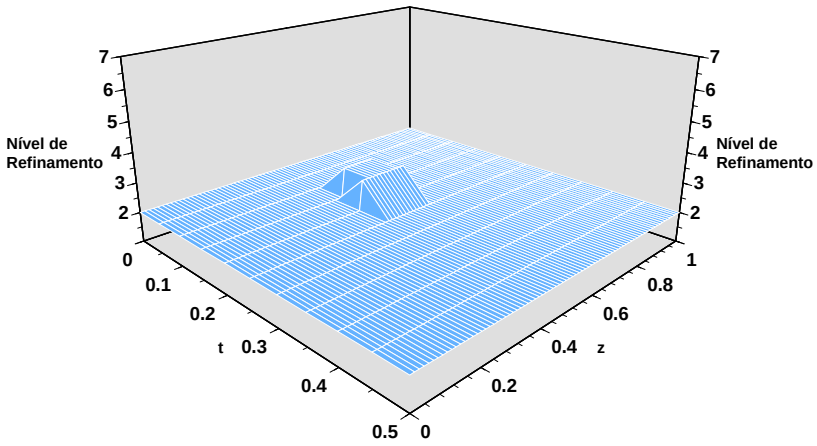

Gráfico 6.82: Resultados do nível de refinamento para o *run2* do exemplo 8.

Assim, conclui-se que, para o presente exemplo, é bastante importante aplicar discretizações não centradas que acompanhem o movimento dos perfis. Deste modo, obtêm-se resultados bastante precisos, com um esforço computacional aceitável. Em ambos os casos, utilizaram-se interpolações com *Splines* cúbicas, devido à forma curvilínea das soluções integradas.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos *Biased Upwind* ***(Variável 1)***
  5 Pontos *Biased Downwind* ***(Variável 2)***
- **Tolerância do Método** – 1×10^{-3} ***(Para as duas variáveis)***
- **Tolerância do Integrador** – 1×10^{-6}
- **Grelha Inicial** – Não Uniforme, 27 Nodos; z=[0.0, 0.1, 0.15, 0.2, 0.225, 0.25, 0.275, 0.3, 0.325, 0.35, 0.375, 0.4, 0.45, 0.5, 0.55, 0.6, 0.625, 0.65, 0.675, 0.7, 0.725, 0.75, 0.775, 0.8, 0.85, 0.9, 1.0]
- **Tipo de Interpolação** – *Splines* Cúbicas 5 Pontos
- $\mathbf{\Delta z_{MIN}}$ – 1×10^{-3}
- $\mathbf{\Delta z_{MAX}}$ – 4×10^{-2}
- α – 1
- λ – 0.9
- **Passo de Tempo Base** – 0.02
- **Tempo Final** – 0.50
- **Tempo de computação (T_{cpu})** – 1106.2 s

Por outro lado, a aplicação do método de malha móvel adaptativa a este exemplo, possibilita a obtenção de perfis semelhantes aos apresentados no caso anterior (vd. Gráficos

6.83 e 6.84). Verifica-se a ocorrência de oscilações ligeiras nas extremidades das ondas, que podem ser suavizadas pela aplicação de um crescente factor de viscosidade (λ). De facto, na execução deste exemplo aplicou-se um valor para λ relativamente elevado (λ=0.9), que conduziu a perfis bastante satisfatórios, com tempos de computação consideravelmente mais reduzidos, em relação aos *runs* efectuados pelo método de refinamento.

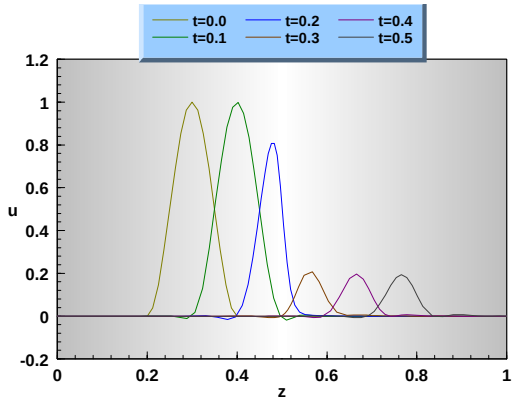

Gráfico 6.83: Resultados obtidos pelo método de malha móvel para o exemplo 8 (variável u).

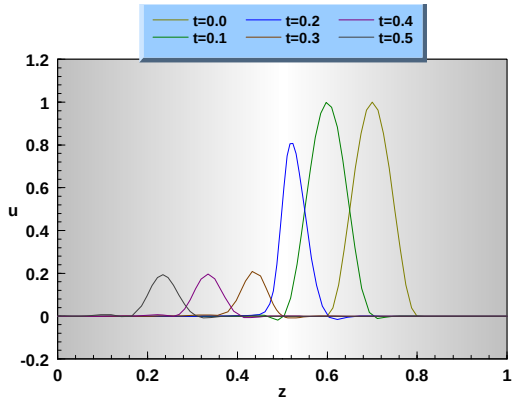

Gráfico 6.84: Resultados obtidos pelo método de malha móvel para o exemplo 8 (variável v).

As malhas base de nível dois, correspondentes a cada variável, acompanham o movimento das ondas correspondentes (vd. Gráficos 6.85 e 6.86). No entanto, este deslocamento parece menos vigoroso após o cruzamento das duas ondas.

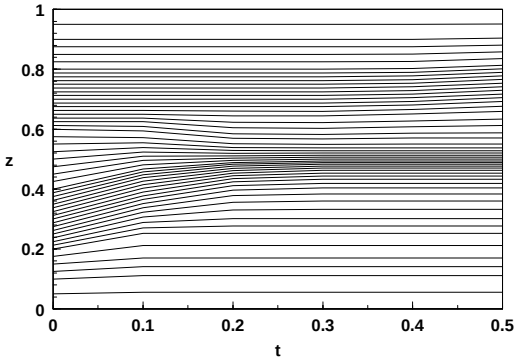

Gráfico 6.85: Evolução da malha inicial de nível 2 para o exemplo 8 (variável u).

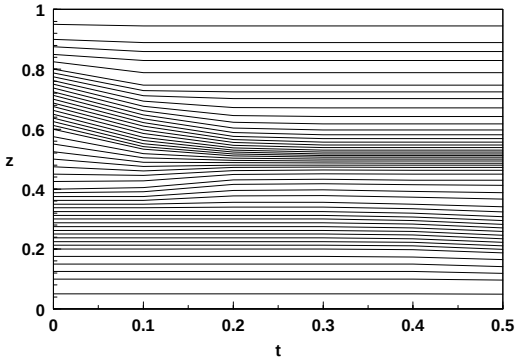

Gráfico 6.86: Evolução da malha inicial de nível 2 para o exemplo 8 (variável v).

Na Tabela 6.10 são comparadas as performances computacionais entre o melhor *run* efectuado com o método de refinamento (*run2*), o *run* executado com o método de malha móvel adaptativa e o algoritmo de elementos finitos móveis utilizado por *Duarte*[29].

Tabela 6.10: Desempenhos computacionais para o modelo de propagação de ondas.

Algoritmo	***Tempo de Computação (s)***
Refinamento	1985.6
Malha Móvel	1106.2
M.E.F.M.[29]	1018.8

O método de refinamento possibilita a obtenção de resultados mais precisos que o M.E.F.M., apesar de necessitar de um tempo computacional comparativamente superior. No entanto, este valor não é incomportável, podendo-se considerar bastante razoável. Por outro lado, o método de malha móvel possibilita resultados semelhantes aos do refinamento, recorrendo a um esforço computacional mais reduzido e apenas ligeiramente superior ao T_{MEFM}. Assim, conclui-se que, no geral, o método de malha móvel se revela o mais adequado para a resolução deste exemplo.

6.3.4 - Exemplo 9: Excitação de um Nervo

O sistema[29] enunciado nesta secção foi resolvido por *Verwer et al*[116] e descreve o fluxo de uma corrente iónica através de uma membrana nervosa semi-infinita. De forma a simular a semi-infinidade da membrana, considera-se um domínio espacial bastante alargado, entre as posições z = 0 e z = 120.

$$\frac{\delta u}{\delta t} = \frac{\delta^2 u}{\delta z^2} + u \cdot (u - a) \cdot (1 - u) - v \tag{6.58}$$

$$\frac{\delta v}{\delta t} = b \cdot (u - c \cdot v) \tag{6.59}$$

Condições Fronteira

$$\frac{\delta u(0,t)}{\delta z} = -\frac{I}{2} \tag{6.60}$$

$$\frac{\delta u(120,t)}{\delta z} = 0 \tag{6.61}$$

Condições Iniciais

$$u(z,0) = 0 \tag{6.62}$$

$$v(z,0) = 0 \tag{6.63}$$

Domínio

$$0 \le z \le 120 \tag{6.64}$$

$$0 \le t \le 200 \tag{6.65}$$

Parâmetros do Modelo

u - potencial electroquímico.
v - estímulo residual.
I = 0.45 - corrente aplicada na fronteira esquerda do nervo.
b = 0.008 - inverso da normalização temporal.
a = 0.139 e c = 2.54 - parâmetros.
z - variável espacial.
t - variável temporal.

Parâmetros da Execução –

⇘ **Tipo de Diferenças Finitas**	–	5 Pontos Centradas	***(Para as duas variáveis)***
⇘ **Tolerância do Método**	–	1×10^{-2}	***(Para as duas variáveis)***
⇘ **Tolerância do Integrador**	–	1×10^{-5}	
⇘ **Grelha Inicial**	–	Uniforme; 41 Nodos	
⇘ **Tipo de Interpolação**	–	*Splines* Cúbicas 5 Pontos	
⇘ **Passo de Tempo Base**	–	1.0	
⇘ **Tempo Final**	–	200.0	
⇘ **Nº Máximo de Iterações**	–	9	
⇘ **Tempo de computação (T_{cpu})**	–	4302.7 s	

O Gráfico 6.87 representa a variação do potencial electroquímico, através de uma sucessão de máximos e mínimos que se propagam ao longo do domínio na direcção da fronteira direita. A variável v (designada por estímulo residual) apresenta um comportamento semelhante (vd. Gráfico 6.88), apesar da magnitude da sua variação ser muito mais baixa, como se pode verificar pela análise do Gráfico 6.89.

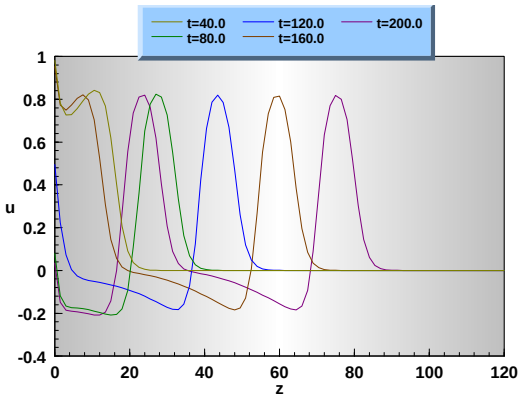

Gráfico 6.87: Resultados obtidos pelo refinamento para o exemplo 9 (variável u).

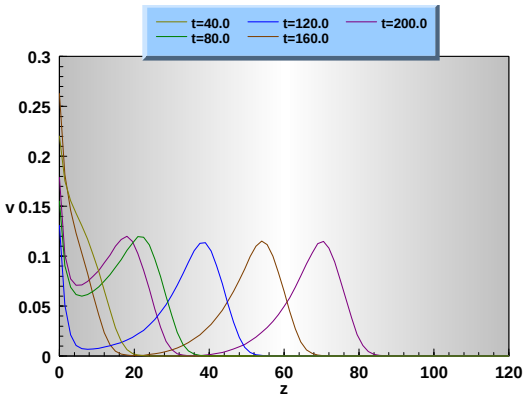

Gráfico 6.88: Resultados obtidos pelo refinamento para o exemplo 9 (variável v).

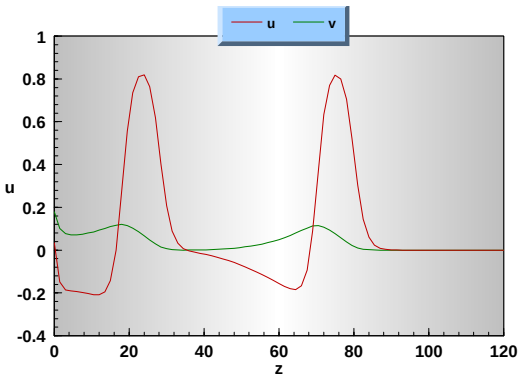

Gráfico 6.89: Comparação entre os perfis de cada variável para o exemplo 9 (t=200.0).

O algoritmo possibilita uma correcta reprodução da evolução dos perfis. No entanto, a obtenção de resultados satisfatórios só é possível através da utilização de uma malha base uniforme apertada (malha de nível 1 com 41 nodos). Desta forma, a aplicação do método exige um esforço computacional significativamente elevado.

Verifica-se, igualmente, que os perfis de refinamento acompanham o movimento das ondas (vd. Gráfico 6.90). Como seria de esperar, as zonas de maior refinamento coincidem com o posicionamento dos picos dos perfis da variável u, cujas ondas apresentam amplitudes mais elevadas.

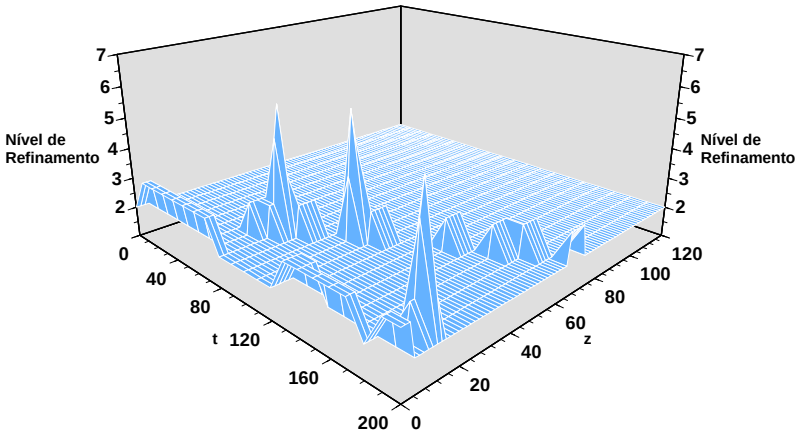

Gráfico 6.90: Resultados do nível de refinamento para o exemplo 9.

Parâmetros da Execução –

↳ **Tipo de Diferenças Finitas**	–	5 Pontos *Biased Up*. ***(Para as duas variáveis)***
↳ **Tolerância do Método**	–	1×10^{-4} ***(Para as duas variáveis)***
↳ **Tolerância do Integrador**	–	1×10^{-6}
↳ **Grelha Inicial**	–	Não Uniforme, 9 Nodos; z=[0.0, 2.0, 4.0, 6.0, 10.0, 20.0, 30.0, 60.0, 120.0]
↳ **Tipo de Interpolação**	–	*Splines* Cúbicas 5 Pontos
↳ Δz_{MIN}	–	1×10^{-2}
↳ Δz_{MAX}	–	2.0
↳ α	–	1
↳ λ	–	0.0
↳ **Passo de Tempo Base**	–	20.0
↳ **Tempo Final**	–	200.0
↳ **Tempo de computação (T_{cpu})**	–	493.8 s

A resolução do presente exemplo através da aplicação do método de malha móvel adaptativa, nas condições apresentadas acima possibilita a obtenção de resultados equivalentes aos do método de refinamento (vd. Gráficos 6.91 e 6.92). No entanto, ao contrário do método anterior, verifica-se ser possível a correcta definição da sucessão de picos da solução, que se propagam ao longo do domínio espacial, recorrendo apenas a uma malha base de primeiro nível muito pouco concentrada (NP_1=9, com os nodos preferencialmente

colocados junto à fronteira esquerda, z=0), o que permite a ocorrência de um tempo computacional de execução extremamente reduzido, em relação ao correspondente ao refinamento. Adicionalmente, utilizam-se, neste caso, discretizações baseadas em fórmulas de diferenças finitas descentradas *upwind*, para a estimativa das derivadas espaciais, que se revelam bastante eficazes no acompanhamento do movimento das ondas.

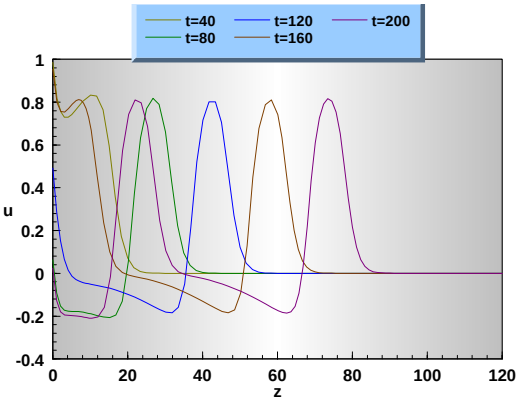

Gráfico 6.91: Resultados obtidos pelo método de malha móvel para o exemplo 9 (variável u).

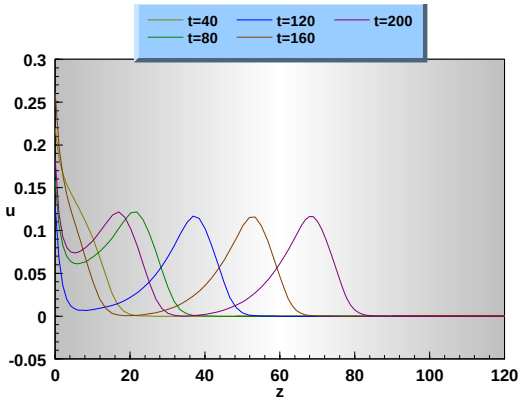

Gráfico 6.92: Resultados obtidos pelo método de malha móvel para o exemplo 9 (variável v).

Curiosamente, constata-se que o deslocamento da malha base é bastante diminuto (vd. Gráficos 6.93 e 6.94), mantendo-se os nodos praticamente fixos no decurso da integração temporal, apesar de se fixar um factor de viscosidade nodal nulo (λ=0). Assim, a reprodução das características e do movimento das curvas sinusoidais, ao longo do tempo, é garantida pela introdução sucessiva de nodos adicionais nas regiões de maior actividade da solução, o que, na prática, corresponde a um refinamento local da malha, e não por um deslocamento nodal significativo. No entanto, verifica-se que as malhas finais de segundo nível, obtidas no final da execução, não são ainda excessivamente densas (NP_2=85 e 79, para as variáveis u e v, respectivamente).

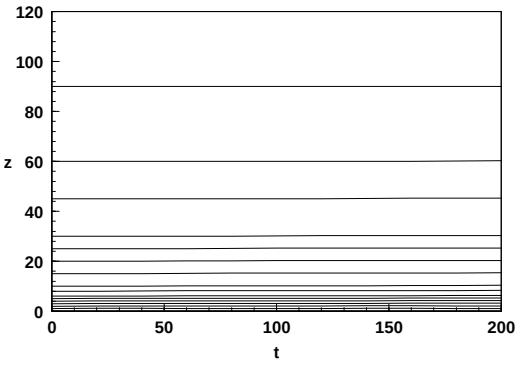

Gráfico 6.93: Evolução da malha inicial de nível 2 para o exemplo 9 (variável u).

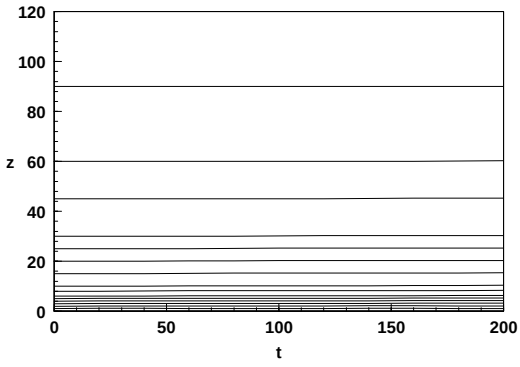

Gráfico 6.94: Evolução da malha inicial de nível 2 para o exemplo 9 (variável v).

Assim, verifica-se que um espaçamento excessivo dos nodos nas malhas base não compromete o desenvolvimento de perfis satisfatórios, na aplicação do método de malha móvel, ao contrário do que se verifica no caso do método de refinamento.

Os desempenhos computacionais de cada um dos métodos são resumidos na tabela seguinte:

Tabela 6.11: Desempenhos computacionais para o modelo de excitação de um nervo.

Algoritmo	*Tempo de Computação (s)*
Refinamento	4302.7
Malha Móvel	493.8
M.E.F.M.[29]	333.7

Os resultados conseguidos através de cada um dos algoritmos desenvolvidos neste trabalho são bastante precisos e semelhantes aos obtidos por *Duarte*[29]. No entanto, o método de refinamento necessita de um esforço computacional consideravelmente mais elevado do que o exigido pelo M.E.F.M., pelas razões referidas anteriormente. Por outro lado, o tempo de cpu associado ao método de malha móvel adaptativa é muito menor, sendo da mesma ordem de grandeza do correspondente ao M.E.F.M., apesar de ser ainda ligeiramente superior a este.

6.3.5 - Exemplo 10: Adsorção num Leito Fixo

Este exemplo[29] consiste num modelo heterogéneo que representa a adsorção de um componente num leito fixo. As limitações difusionais no sólido são desprezadas e, desse modo, não estão envolvidas derivadas espaciais no respectivo balanço (vd. Equação (6.67)).

$$\frac{\delta u_1}{\delta t} = -v \cdot \frac{\delta u_1}{\delta z} - k_1 \cdot u_1 \cdot (Q - u_2) + k_2 \cdot u_2 \qquad (6.66)$$

$$\frac{\delta u_2}{\delta t} = k_1 \cdot u_1 \cdot (Q - u_2) - k_2 \cdot u_2 \qquad (6.67)$$

Condições Fronteira

$$u_1(0,t) = C_0 \qquad (6.68)$$

Condições Iniciais

$$u_1(z,0) = 0 \qquad (6.69)$$

$$u_2(z,0) = 0 \qquad (6.70)$$

Domínio

$$0 \le z \le 1 \qquad (6.71)$$

$$0 \le t \le 50 \qquad (6.72)$$

Parâmetros do Modelo

u_1 - concentração do componente adsorvido no líquido.
u_2 - concentração do componente adsorvido no sólido.
z - variável espacial.
t - variável temporal.
k_1 = 0.01553 - coeficiente de transferência de massa entre o líquido e o sólido.
k_2 = 0.002661 - coeficiente de transferência de massa entre o sólido e o líquido.
Q = 967.3 - capacidade máxima de adsorção do sólido.
v = 6.6845 - velocidade do fluido.
C_0 = 3 - concentração do componente adsorvido à entrada.

Parâmetros da Execução –

Tipo de Diferenças Finitas	–	5 Pontos Centradas	***(Para as duas variáveis)***
Tolerância do Método	–	5×10^{-2}	***(Para as duas variáveis)***
Tolerância do Integrador	–	1×10^{-5}	
Grelha Inicial	–	Uniforme; 21 Nodos	
Tipo de Interpolação	–	*Splines* Cúbicas 5 Pontos	
Passo de Tempo Base	–	1.0	
Tempo Final	–	50.0	
Nº Máximo de Iterações	–	10	
Tempo de computação (T_{cpu})	–	204.6 s	

Os resultados obtidos com os parâmetros apresentados acima (*run1*) são muito semelhantes aos obtidos por *Duarte*[29] (vd. Gráficos 6.95 a 6.98). A solução é integrada sem o recurso a qualquer operação de refinamento. Apesar disso, obtêm-se resultados bastante correctos com um tempo de computação muito baixo.

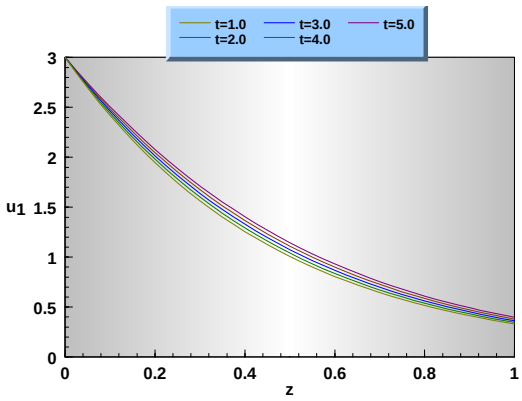

Gráfico 6.95: Resultados obtidos pelo refinamento para o *run1* do exemplo 10 (tf=5.0, variável u).

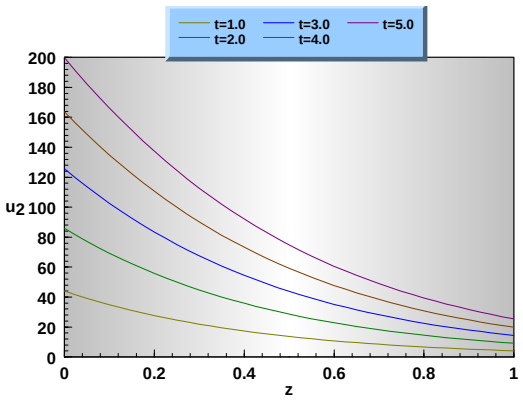

Gráfico 6.96: Resultados obtidos pelo refinamento para o *run1* do exemplo 10 (tf=5.0, variável v).

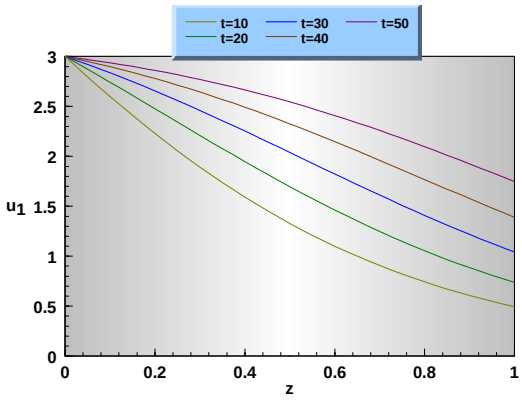

Gráfico 6.97: Resultados obtidos pelo refinamento para o *run1* do exemplo 10 (tf=50.0, variável u).

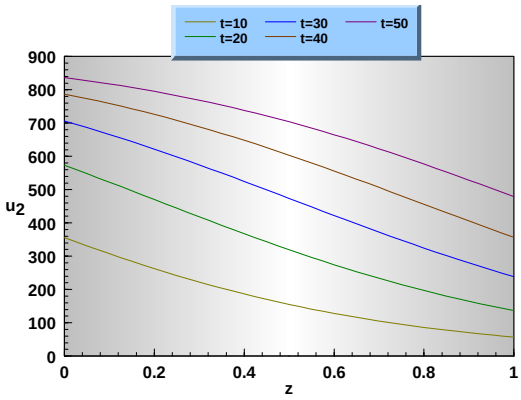

Gráfico 6.98: Resultados obtidos pelo refinamento para o *run1* do exemplo 10 (tf=50.0, variável v).

A taxa de variação da concentração do componente adsorvido no líquido (variável u_1) é muito reduzida (vd. Gráfico 6.95 e 6.97). Pelo contrário, a concentração correspondente no sólido (variável u_2) sofre grandes variações devido ao elevado factor de saturação Q (vd. Gráficos 6.96 e 6.98). De qualquer modo, ambos os perfis são bastante suaves, não provocando quaisquer dificuldades no avanço da integração.

No entanto, verifica-se que quando se impõe uma tolerância muito baixa, que corresponde a um grau de precisão da solução refinada mais elevado e o algoritmo selecciona qualquer subdomínio de refinamento, aquele necessita de reduzir consideravelmente o passo temporal devido a problemas numéricos de integração dos subproblemas gerados. Desta forma o método vê-se obrigado a reproduzir a passagem da onda mássica introduzida pela perturbação em degrau de u_1, efectuada em z=0. O tempo de passagem da onda mássica é muito reduzido ($t \cong 0.15$), comparativamente à dimensão do domínio temporal total (T=50.0). Assim, nas condições do *run2*, este é subdividido em vários intervalos, devido às diferentes características da solução em cada uma dessa zonas. Portanto, inicialmente, durante a fase de propagação da onda mássica (Zona 1), o passo temporal base é muito pequeno e a tolerância especificada para o refinamento é mais baixa. As condições estabelecidas para a execução do *run2* são apresentadas de seguida:

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 Pontos Centradas ***(Para as duas variáveis)***
- **Tolerância do Método** – 5×10^{-3} **(Zona 1)** ***(Para as duas variáveis)***
  5×10^{-2} **(Zona 2 e 3)**
- **Tolerância do Integrador** – 1×10^{-5}
- **Grelha Inicial** – Uniforme; 11 Nodos
- **Tipo de Interpolação** – Linear
- **Passo de Tempo Base** – 5×10^{-4} **(Zona 1)** ; 0.1 **(Zona 2)** ; 1.0 **(Zona 3)**
- **Tempo Final** – 50.0
- **Nº Máximo de Iterações** – 10
- **Tempo de computação (T_{cpu})** – 590.6 s

Zona 1: $t \in [0.0, 0.2[$
Zona 2: $t \in [0.2, 1.0[$
Zona 3: $t \in [1.0, 50.0]$

Com uma malha base de nível 1 de apenas 11 nodos, observam-se ligeiras oscilações na formação da frente móvel (vd. Gráfico 6.99), sendo a sua espessura algo elevada em relação ao esperado. No entanto, é de salientar que estes problemas poderiam ser resolvidos com a utilização de uma grelha mais concentrada, à custa de um aumento considerável do tempo de computação global. Não se verificam quaisquer problemas na evolução dos perfis da variável u_2 nesta fase da integração (vd. Gráfico 6.100).

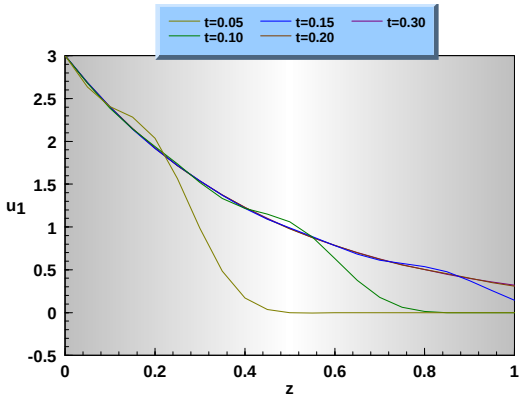

Gráfico 6.99: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=0.30, variável u).

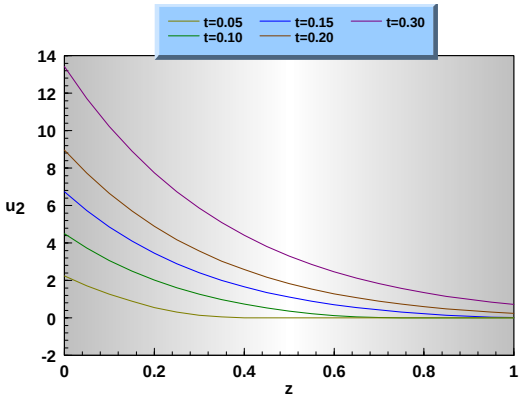

Gráfico 6.100: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=0.30, variável v).

O refinamento é bastante activo no início da execução da Zona 1, de forma a simular o efeito da introdução da perturbação no sistema. No entanto, a sua aplicação é completamente desnecessária em fases mais adiantadas da integração nessa zona e para a totalidade das Zonas 2 e 3 do domínio temporal.

Os resultados obtidos são praticamente idênticos aos do *run1* (vd. Gráficos 6.101 a 6.104), apesar de neste caso se recorrerem a interpolações lineares e a malha ser bastante mais larga.

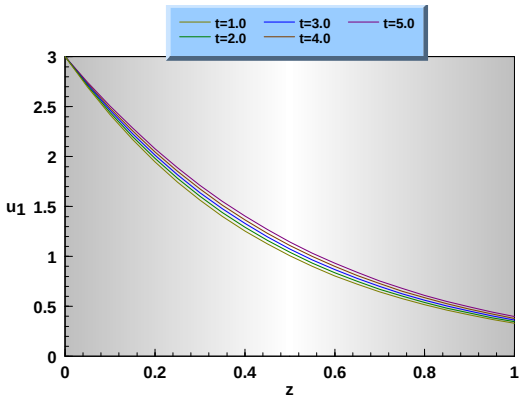

Gráfico 6.101: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=5.0, variável u).

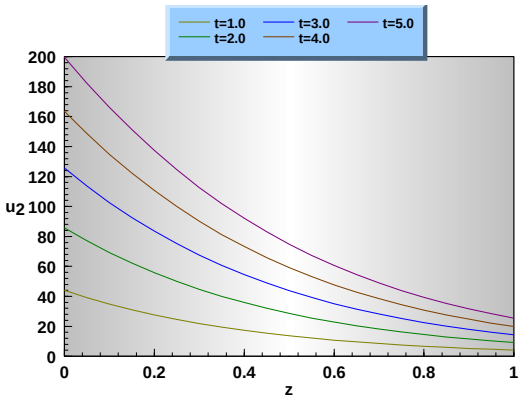

Gráfico 6.102: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=5.0, variável v).

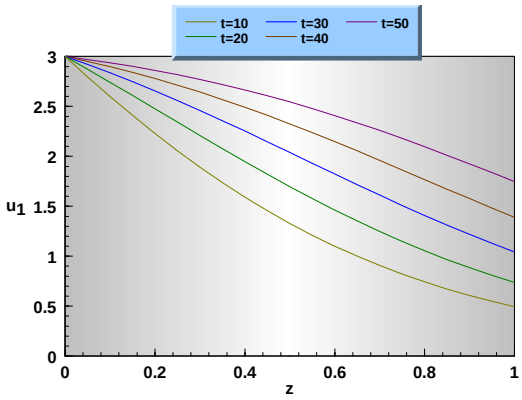

Gráfico 6.103: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=50.0, variável u).

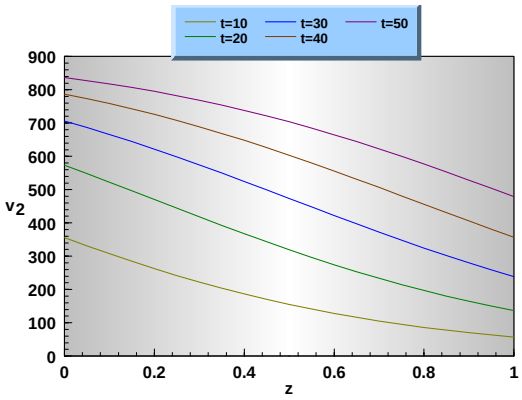

Gráfico 6.104: Resultados obtidos pelo refinamento para o *run2* do exemplo 10 (tf=50.0, variável v).

Devido ao aumento da precisão exigida e ao facto do algoritmo ser obrigado a simular o comportamento inicial do processo, verifica-se um aumento do tempo computacional neste caso, em relação ao obtido nas condições do *run1*.

Na Tabela 6.12 efectua-se a comparação entre o comportamento computacional de cada método numérico considerado no presente trabalho. O tempo de computação apresentado, referente ao método de refinamento, diz respeito ao *run2*, já que nessas condições é possível simular com mais precisão as diferentes características da solução, ao longo da integração.

Tabela 6.12: Desempenhos computacionais para o modelo de adsorção num leito fixo.

Algoritmo	***Tempo de Computação (s)***
Refinamento	590.6
M.E.F.M.[29]	14485.9

Apesar dos resultados obtidos por utilização de cada um dos métodos serem muito semelhantes, verifica-se que a aplicação do M.E.F.M. exige um esforço computacional extremamente elevado. Pelo contrário, o método de refinamento possibilita a obtenção de perfis correctos, em tempos de computação bastante reduzidos. Desta forma, é possível concluir-se que o M.E.F.M. é demasiado potente e complexo para um modelo que não coloca grandes problemas de resolução, apesar da discrepância existente entre a dimensão relativa de cada uma das variáveis que compõem o modelo, e dos seus gradientes.

6.4 – Aplicação dos Métodos Numéricos a Sistemas Mistos de P.D.A.E.´s

6.4.1 - Exemplo 11: Reactor Tubular Não Isotérmico

Neste problema[29] é simulado o comportamento de um reactor tubular não isotérmico, onde se processa uma reacção de primeira ordem do tipo A $\rightarrow$ P. Admite-se que a entalpia é uma função quártica da diferença entre a temperatura da mistura reaccional e a respectiva temperatura de referência (vd. Equação (6.75)).

$$\frac{\delta C}{\delta t} = D \cdot \frac{\delta^2 C}{\delta z^2} - v \cdot \frac{\delta C}{\delta z} - k \cdot C \tag{6.73}$$

$$\frac{\delta H}{\delta t} = k_h \cdot \frac{\delta^2 T}{\delta z^2} - v \cdot \frac{\delta H}{\delta z} + (-\Delta H) \cdot k \cdot C \tag{6.74}$$

$$H = b \cdot (T - T_{REF}) + c \cdot (T - T_{REF})^2 + d \cdot (T - T_{REF})^3 + e \cdot (T - T_{REF})^4 \tag{6.75}$$

Condições Fronteira

$$C(0,t) = 1 \tag{6.76}$$

$$H(0,t) = 7.22 \tag{6.77}$$

$$\frac{\delta C(1,t)}{\delta z} = 0 \tag{6.78}$$

$$\frac{\delta H(1,t)}{\delta z} = 0 \tag{6.79}$$

Condições Iniciais

$$C(z,0) = 0 \tag{6.80}$$

$$H(z,0) = 7.22 \tag{6.81}$$

Domínio

$$0 \leq z \leq 1 \tag{6.82}$$

$$0 \leq t \leq 5 \tag{6.83}$$

Parâmetros do Modelo

C - concentração do reagente.
H - entalpia da mistura reaccional.
T - temperatura da mistura reaccional.
z - variável espacial.
t - variável temporal.
$D = 2.58 \times 10^{-3}$ - difusividade axial efectiva.
v = 0.2 - velocidade do fluido no interior do reactor.
k = 0.15 - constante cinética da reacção.

$k_h = 8.35\times10^{-3}$ - condutividade da fase fluida.
$(-\Delta H) = 100$ - calor de reacção.
$b = 3.635$, $c = -0.0144915$, $d = 1.29\times10^{-5}$, $e = -3.864\times10^{-9}$ - coeficientes da função entálpica.
$T_{REF} = 298$ - temperatura de referência.

Parâmetros da Execução –

- **Tipo de Diferenças Finitas** – 5 P. Tipo variável ***(Para as três variáveis)***
- **Tolerância do Método** – **Var. 1:** 5×10^{-3} **(Z. 1)**; 1×10^{-2} **(Z. 2)**
  Var. 2: 1×10^{-2} **(Z. 1)**; 5×10^{-2} **(Z. 2)**
- **Tolerância do Integrador** – 1×10^{-6}
- **Grelha Inicial** – Uniforme; NP variável
- **Passo de Tempo Base** – 0.01 **(Zona 1)** ; 0.1 **(Zona 2)**
- **Tempo Final** – 5.0

Zona 1: $t \in [0.0, 0.1[$
Zona 2: $t \in [0.1, 5.0]$

Há medida que a reacção decorre, esta desloca-se na direcção da saída do reactor. Devido ao aumento da extensão da reacção, forma-se uma onda entálpica, na qual o *hotspot* percorre o reactor, situando-se já bastante próximo da fronteira direita para t=5. Como o modelo estabelece uma relação quártica entre a entalpia (H) e a temperatura (T), verifica-se o desenvolvimento de perfis com formas muito semelhantes para a descrição de cada uma dessas variáveis.

As condições variáveis de cada *run* efectuado para a integração deste modelo estão resumidas na tabela seguinte:

Tabela 6.13: Condições fixadas para a execução dos *runs* do modelo do reactor tubular não isotérmico.

Run	***NP*** ***(Malha de Nível 1)***	***Tipo de Dsc. Espacial***	***Tipo de Interpolação***	***tcpu (s)*** ***(Tfinal = 50.0)***
1	21	Centrais	Linear	1837.4
2	21	*Biased Up.*	Linear	1806.0
3	31	*Biased Up.*	*Splines* 5 P.	4110.1

Nos Gráficos 6.105 a 6.107, apresentam-se os resultados obtidos pela execução do *run1*, caracterizado por diferenças finitas do tipo centrado. Verifica-se que os resultados são, em geral, bastante semelhantes aos obtidos por *Duarte*[29], apesar de se notar a ocorrência de alguma instabilidade na região próxima a z=1, que se tornou habitual em modelos caracterizados pela movimentação de ondas e descritos por condições de fronteira de *Neumann*, como é o caso deste exemplo. Normalmente, a instabilidade referida apenas se torna importante após a ocorrência do choque da onda na respectiva fronteira, tornando-se visível pela formação de distorções nos perfis, na região do domínio espacial vizinha à fronteira (vd. Gráficos 6.105 a 6.107).

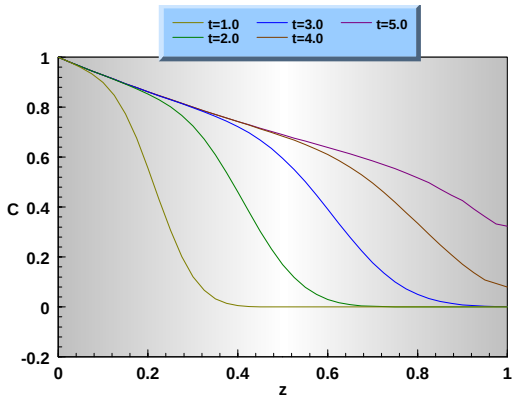

Gráfico 6.105: Resultados obtidos pelo refinamento para o *run1* do exemplo 11 (var. C).

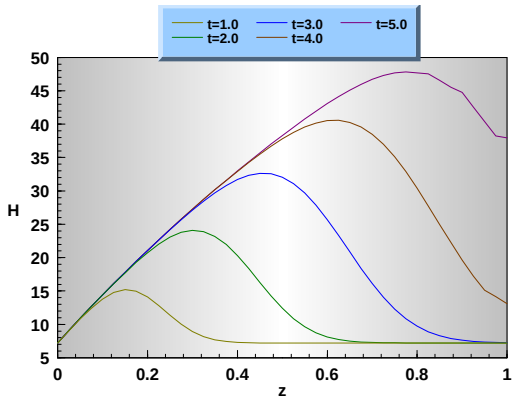

Gráfico 6.106: Resultados obtidos pelo refinamento para o *run1* do exemplo 11 (var. H).

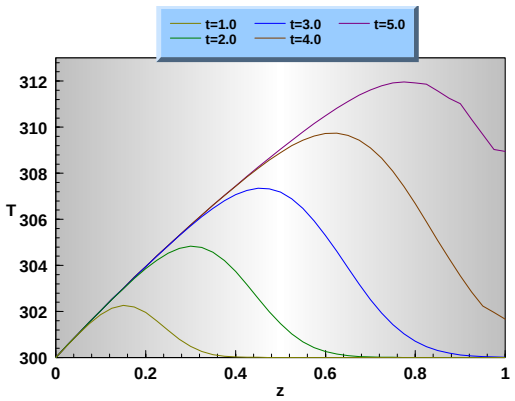

Gráfico 6.107: Resultados obtidos pelo refinamento para o *run1* do exemplo 11 (var. T).

Estas anomalias nos perfis são originadas pelo facto de existir uma grande necessidade de refinamento nessa região no período posterior à passagem da onda no reactor (vd. Gráfico 6.108). Este tipo de comportamento, mais importante para o caso da utilização de fórmulas centradas na discretização espacial, também contribui para o aumento do esforço computacional associado a esses casos.

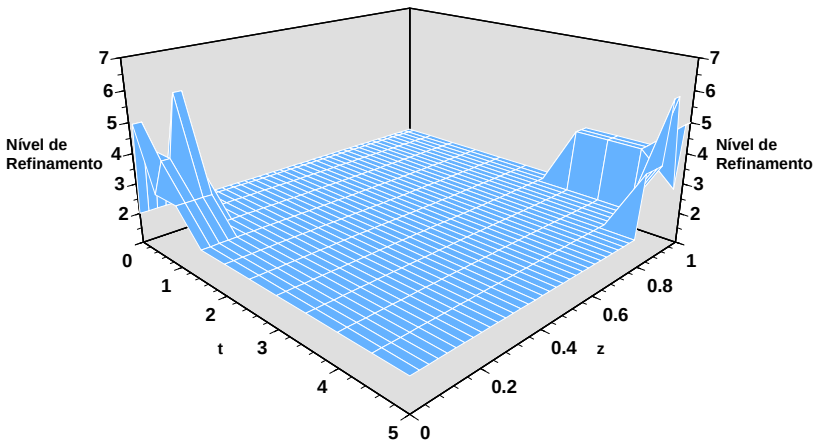

Gráfico 6.108: Resultados do nível de refinamento para o *run1* do exemplo 11.

Nas condições dos *run2* e *run3*, recorre-se a diferenças descentradas do tipo *upwind*, que se revelam mais apropriadas para o acompanhamento do deslocamento positivo de ondas móveis. No caso do *run2* (vd. Gráficos 6.109 a 6.111), constata-se que os perfis calculados são muito similares aos obtidos nas condições do *run1*. No entanto, nota-se um decréscimo acentuado na importância das anomalias apontadas anteriormente, que se tornam praticamente imperceptíveis. Verifica-se, igualmente um ligeiro decréscimo do tempo de cpu referente a este caso, em relação ao anterior.

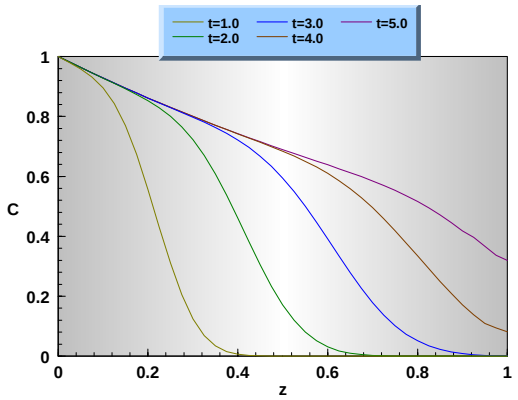

Gráfico 6.109: Resultados obtidos pelo refinamento para o *run2* do exemplo 11 (var. C).

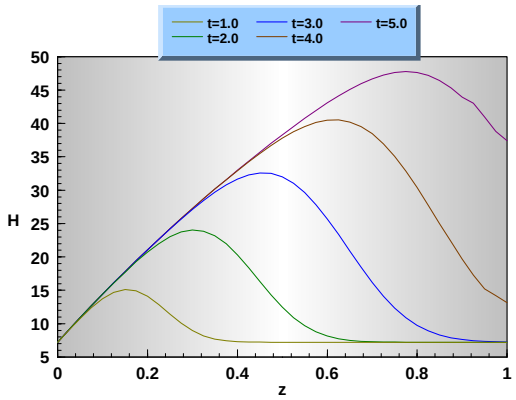

Gráfico 6.110: Resultados obtidos pelo refinamento para o *run2* do exemplo 11 (var. H).

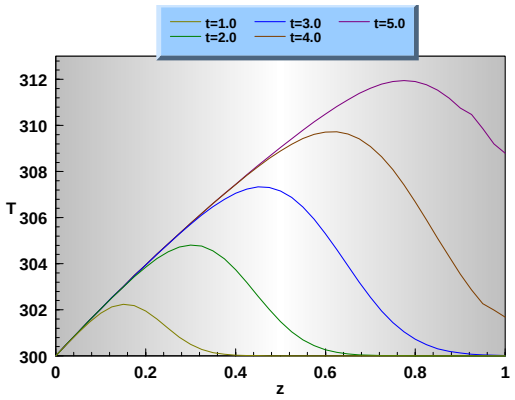

Gráfico 6.111: Resultados obtidos pelo refinamento para o *run2* do exemplo 11 (var. T).

No que diz respeito à análise dos perfis de refinamento (vd. Gráfico 6.112), verifica-se o comportamento esperado, ou seja, a importância do refinamento é mais sentida em dois períodos distintos da integração: no arranque do processo, junto à fronteira esquerda do domínio, de forma a que o algoritmo possa simular convenientemente a perturbação produzida pela introdução súbita da corrente de alimentação no reactor; junto à fronteira direita, após a passagem completa da onda mássica pelo reactor, devido a dificuldades numéricas do algoritmo para lidar com a condição fronteira de *Neumann* aí definida.

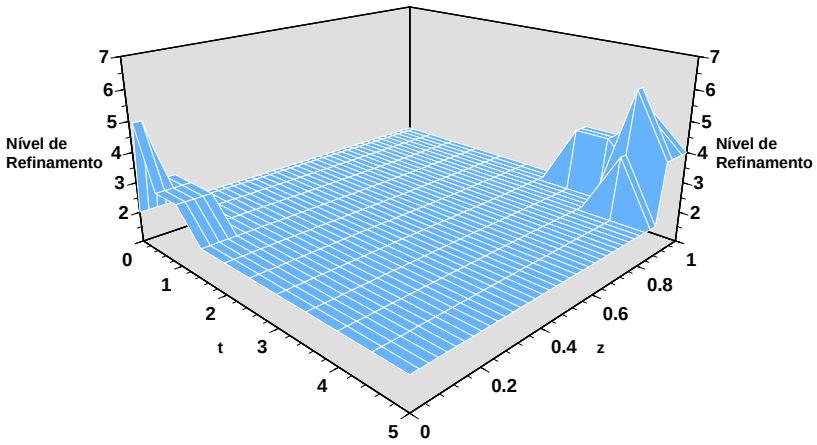

Gráfico 6.112: Resultados do nível de refinamento para o *run2* do exemplo 11.

De qualquer modo, obtêm-se bons resultados nas condições do *run2*, apesar de se utilizarem interpolações lineares e os perfis das variáveis serem suaves e arredondados.

Verifica-se, igualmente, uma diminuição visível da instabilidade observada anteriormente, através da aplicação de uma grelha base mais apertada e da utilização de *splines* cúbicas para as interpolações (vd. Gráficos 6.113 a 6.115). No entanto, nas condições do *run3*, estas vantagens são obtidas à custa de um aumento considerável do tempo de computação necessário para a execução. Deste modo, é possível concluir-se que o parâmetro que mais afecta a qualidade dos resultados neste caso é o tipo de discretização espacial aplicada.

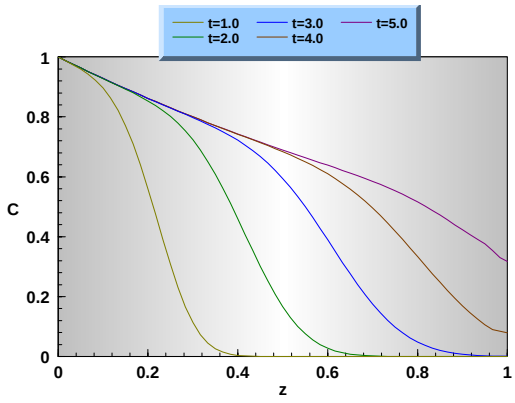

Gráfico 6.113: Resultados obtidos pelo refinamento para o *run3* do exemplo 11 (var. C).

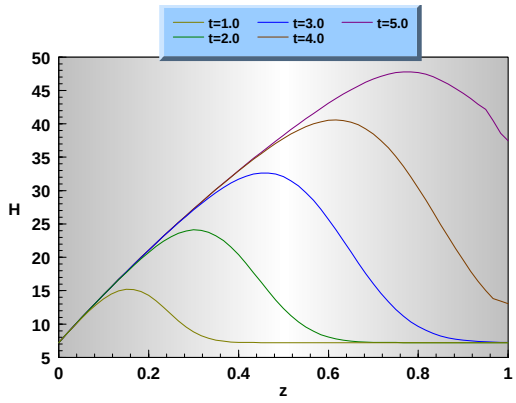

Gráfico 6.114: Resultados obtidos pelo refinamento para o *run3* do exemplo 11 (var. H).

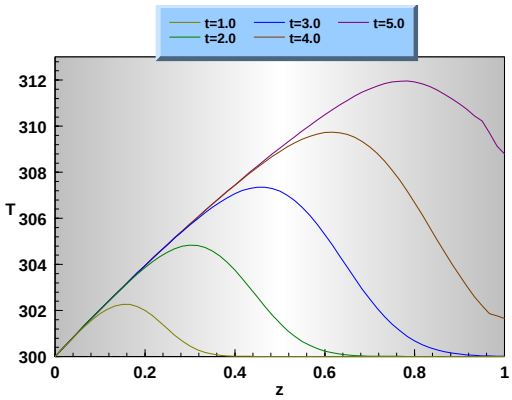

Gráfico 6.115: Resultados obtidos pelo refinamento para o *run3* do exemplo 11 (var. T).

O perfil de refinamento referente ao *run3* é semelhante aos obtidos nos casos anteriores, apresentando as mesmas características gerais (vd. Gráfico 6.116).

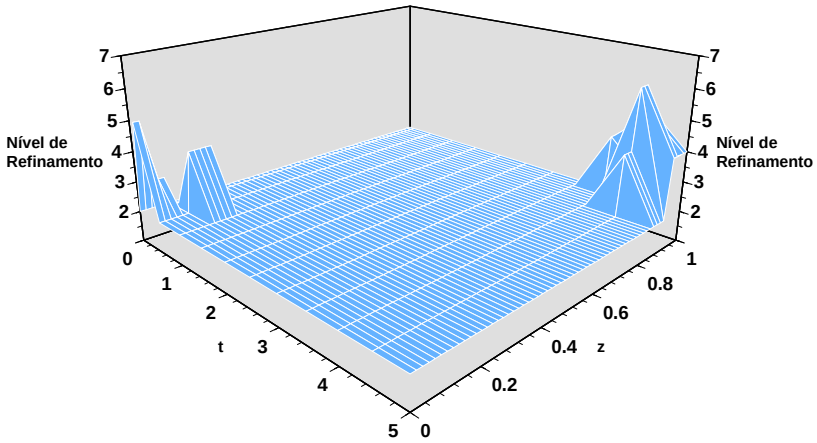

Gráfico 6.116: Resultados do nível de refinamento para o *run3* do exemplo 11.

Este caso demonstra também que, para este tipo de modelos, seria desejável uma revisão da estratégia de tratamento das condições fronteira dos subproblemas gerados que, em princípio, é a maior responsável pela ocorrência das descontinuidades referidas neste exemplo e em outros apresentados anteriormente. Estas anomalias são, normalmente, mais importantes, para execuções onde se utilizem expressões de discretização centradas, as quais sentem algumas dificuldades em acompanhar o movimento das ondas, sem que estas sejam necessariamente muito abruptas.

Torna-se importante frisar que, em todos os casos, se verificou ser indispensável anular o valor da derivada espacial da temperatura (T) na fronteira direita, por acção externa do utilizador. Esta condição não deveria ser necessária devido ao facto de estar implícita através das seguintes definições, presentes no modelo: condição de *Neumann* nula para a fronteira da direita referente à variável entalpia (H); relação quártica entre as variáveis H e T. No entanto, essa especificação não é verificada devido à ocorrência de erros numéricos na avaliação da derivada por intermédio da respectiva expressão de discretização. De facto, a simples utilização da solução nos pontos discretos vizinhos para o cálculo da derivada espacial em z=1, não garante de forma alguma que o seu valor seja nulo. Essa discrepância é bastante importante, já que inviabiliza completamente o avanço da integração.

Os desempenhos computacionais, associados aos métodos estudados neste trabalho, estão resumidos na Tabela 6.14:

Tabela 6.14: Desempenhos computacionais para o modelo do reactor tubular não isotérmico.

Algoritmo	***Tempo de Computação (s)***
Refinamento	1806.0
M.E.F.M.[29]	36126.3

Em geral, verifica-se que os resultados obtidos por aplicação do método de refinamento, associado a discretizações do tipo *upwind*, são muito aceitáveis e semelhantes

aos apresentados por *Duarte*[29]. No entanto, como o M.E.F.M. exige esforços computacionais muito superiores, constata-se que este método se revela demasiado complexo para a resolução deste tipo de modelos, caracterizados por perfis bastante suaves. Desta forma, é possível concluir-se que através do método de refinamento, se obtêm resultados correctos, com tempos de computação razoáveis, não obstante o problema ser fortemente *stiff*. Assim, este método é o mais adequado para a resolução deste exemplo.

6.4.2 - Exemplo 12: Coluna de Adsorção/Reacção

No presente exemplo[29] estuda-se o comportamento de um processo simultâneo de adsorção e reacção químicas numa coluna de adsorção. O soluto A é adsorvido em carvão activado, ao mesmo tempo que se combina com um metal M através de uma reacção irreversível descrita pela equação A + α M $\rightarrow$ P. Desprezam-se os gradientes radiais de concentração no sólido e admitem-se resistências internas e externas à transferência de massa na interface sólido/líquido. Finalmente, considera-se que a isotérmica de equilíbrio é do tipo acção de massas (vd. Equação (6.88)).

$$\frac{\delta u}{\delta t} = \frac{1}{Pe} \cdot \frac{\delta^2 u}{\delta z^2} - \frac{\delta u}{\delta z} - N_f \cdot \left(u - u^*\right) \tag{6.84}$$

$$\frac{\delta v}{\delta t} = N_d \cdot \left(v^* - v\right) - Da \cdot v \cdot M \tag{6.85}$$

$$\frac{\delta M}{\delta t} = -\alpha \cdot Da \cdot \chi \cdot v \cdot M \tag{6.86}$$

$$u - u^* = \frac{N_d}{N_f} \cdot \eta \cdot \left(v^* - v\right) \tag{6.87}$$

$$v^* = \frac{K \cdot u^*}{1 + \left(K - 1\right) \cdot u^*} \tag{6.88}$$

<u>*Condições Fronteira*</u>

$$\frac{\delta u(0,t)}{\delta z} = Pe \cdot \left(u - 1\right) \tag{6.89}$$

$$\frac{\delta u(1,t)}{\delta z} = 0 \tag{6.90}$$

<u>*Condições Iniciais*</u>

$$u(z,0) = 0 \tag{6.91}$$

$$v(z,0) = 0 \tag{6.92}$$

$$M(z,0) = 1 \tag{6.93}$$

<u>*Domínio*</u>

$$0 \le z \le 1 \tag{6.94}$$

$$0 \le t \le 4000 \tag{6.95}$$

Parâmetros do Modelo

$u = \frac{C}{C_0}$ - concentração de A no fluido, normalizada em relação à concentração de entrada.

C - concentração do componente A na fase fluida da mistura reaccional.

C_0 - concentração do componente A na corrente de entrada.

$v = \frac{q}{q_0}$ - concentração de A no sólido, normalizada em relação à sua concentração inicial.

q - concentração de A na fase sólida da mistura reaccional.

q_0 - concentração inicial de A no sólido.

$M = \frac{C_M}{C_{M0}}$ - concentração do metal, normalizada em relação à sua concentração inicial.

C_M - concentração do metal.

C_{M0} - concentração inicial do metal.

$u^* = \frac{C^*}{q_0}$ - concentração de A no fluido em equilíbrio com u, normalizada em relação à concentração inicial q_0.

$v^* = \frac{q^*}{q_0}$ - concentração de A no sólido em equilíbrio com v, normalizada em relação à concentração inicial q_0.

$z = \frac{\xi}{L}$ - variável espacial normalizada em relação ao comprimento da coluna.

ξ - variável espacial.

L - comprimento da coluna.

$t = \frac{\theta}{\tau}$ - tempo adimensional, normalizado em relação ao tempo de passagem do fluido.

θ - variável temporal.

$\tau = \frac{L}{U_i}$ - tempo de passagem de um elemento de fluido na coluna.

U_i - velocidade do fluido no interior da coluna.

$Pe = \frac{U_i \cdot L}{D_{eff}} = 10^4$ - número de *Peclet* axial.

D_{eff} - difusividade axial efectiva.

$Da = k \cdot \rho_b \cdot C_{M0} \cdot \tau = 1.575 \times 10^{-4}$ - número de *Damkhöler*.

k - constante de velocidade da reacção.

ρ_b - massa específica aparente do carvão referida ao volume do leito.

$N_f = \frac{k_f \cdot a_v \cdot \tau}{\varepsilon} = 74.5$ - número de unidades de transferência de massa por difusão no filme.

k_f - coeficiente de transferência de massa.

a_v - área específica da partícula referida ao volume do leito.

ε - porosidade do leito.

$N_d = k_s \cdot a_v \cdot \tau = 2.5$ - número de unidades de transferência de massa por difusão na interface do lado do sólido.

k_s - coeficiente de transferência de massa do lado do sólido.

$\alpha = 0.2$ - coeficiente estequiométrico.

$\chi = \frac{q_0}{C_{M0}} = 0.617$ - razão entre as concentrações iniciais de A no fluido e no metal.

$\eta = \frac{\rho_b \cdot q_0}{\varepsilon \cdot C_0} = 4143$ - factor de capacidade do leito.

K = 1.177 - constante de equilíbrio.

Parâmetros da Execução –

⇘ **Tipo de Diferenças Finitas**	–	5 P. *Biased Upwind*	***(Para as duas variáveis)***
⇘ **Tolerância do Método**	–	1×10^{-2}	***(Para as duas variáveis)***
⇘ **Tolerância do Integrador**	–	1×10^{-5}	
⇘ **Grelha Inicial**	–	Uniforme; 11 Nodos	
⇘ **Tipo de Interpolação**	–	*Splines* Cúbicas 5 Pontos	
⇘ **Passo de Tempo Base**	–	200	
⇘ **Tempo Final**	–	4000	
⇘ **Nº Máximo de Iterações**	–	10	
⇘ **Tempo de computação (T_{cpu})**	–	400.9 s	

Nos Gráficos 6.117 a 6.122, apresentam-se os perfis da solução e de refinamento obtidos nas condições listadas acima. Observa-se que, após se submeter a coluna a uma perturbação em degrau na variável concentração do soluto (u), esta gera uma onda mássica de características difusivas que se desloca na direcção positiva de z. Obtêm-se perfis similares para a variável v e para as concentrações de equilíbrio no sólido e no fluido (u^* e v^*, respectivamente). A concentração de metal no suporte de carvão activado (M) diminui, particularmente, junto à alimentação, devido à maior extensão da reacção nessa zona. Os resultados são muito semelhantes aos obtidos por *Duarte*[29], apesar do esforço computacional dispendido neste caso ser relativamente superior.

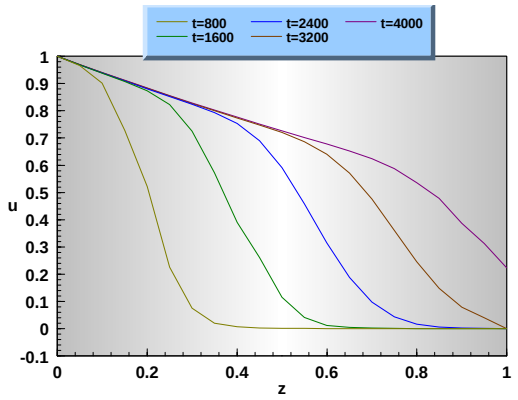

Gráfico 6.117: Resultados obtidos pelo refinamento para o *run1* do exemplo 12 (var. u).

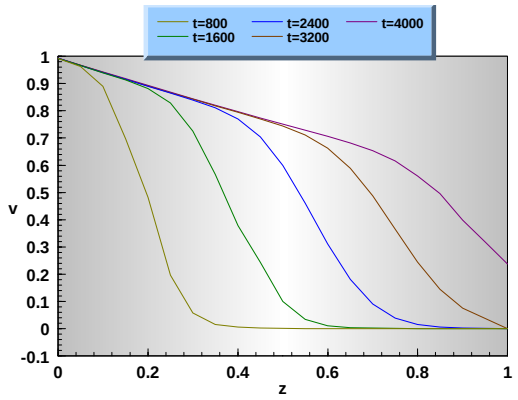

Gráfico 6.118: Resultados obtidos pelo refinamento para o *run1* do exemplo 12 (var. v).

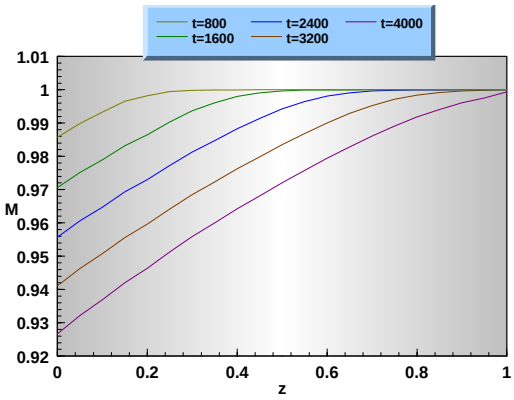

Gráfico 6.119: Resultados obtidos pelo refinamento para o *run1* do exemplo 12 (var. M).

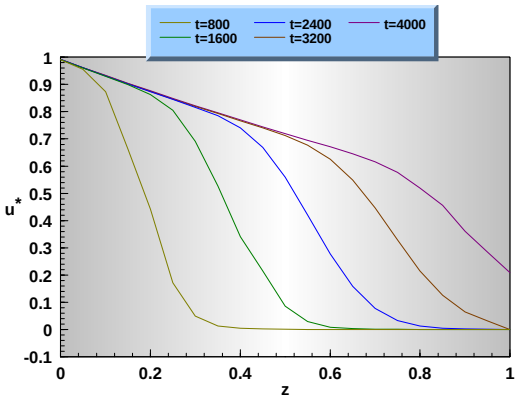

Gráfico 6.120: Resultados obtidos pelo refinamento para o *run1* do exemplo 12 (var. u^*).

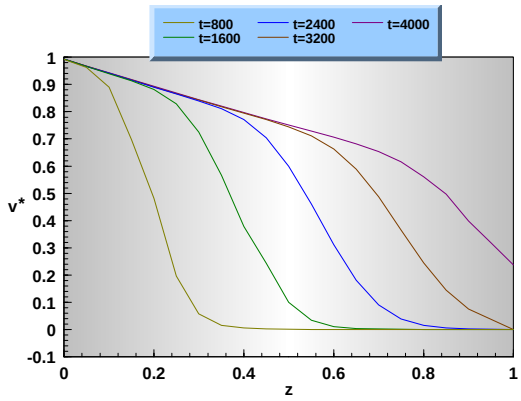

Gráfico 6.121: Resultados obtidos pelo refinamento para o *run1* do exemplo 12 (var. v^*).

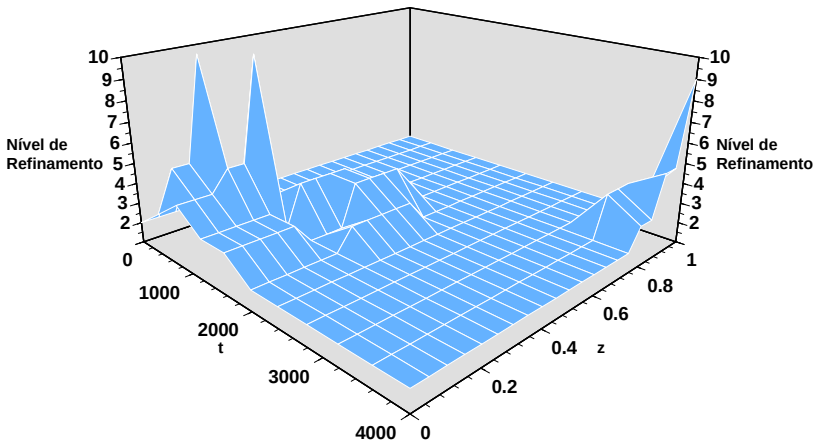

Gráfico 6.122: Resultados do nível de refinamento para o *run1* do exemplo 12.

A utilização de discretizações espaciais baseadas em diferenças finitas centrais (mantendo todos os restantes parâmetros inalterados) conduziu à obtenção de resultados praticamente coincidentes aos do caso anterior (vd. Gráficos 6.123 a 6.127). No entanto, verifica-se que esta execução exige um tempo de computação significativamente superior, para integrar o modelo em estudo: T_{cpu}=1646.9 s. Este facto deve-se essencialmente a:

❶ Maiores dificuldades para as diferenças finitas centradas lidarem com o efeito da perturbação inicial introduzida no sistema. A maior parte do esforço computacional investido na integração é utilizado na fase de arranque.

❷ As expressões centradas têm mais dificuldade em acompanhar o movimento das ondas e, portanto, levam a que o algoritmo necessite de refinar zonas mais extensas do domínio e por períodos de tempo maiores (vd. Gráficos 6.122 e 6.128).

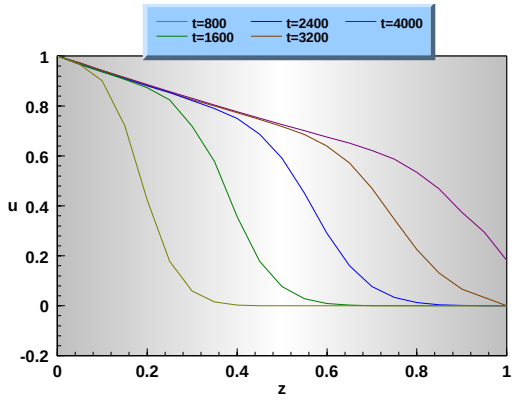

Gráfico 6.123: Resultados obtidos pelo refinamento para o *run2* do exemplo 12 (var. u).

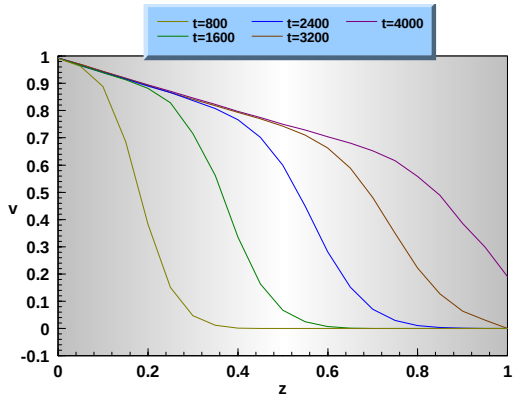

Gráfico 6.124: Resultados obtidos pelo refinamento para o *run2* do exemplo 12 (var. v).

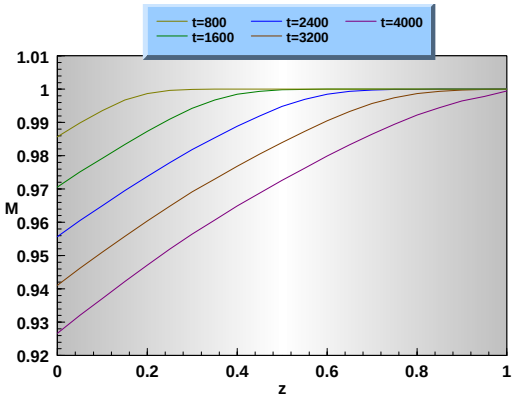

Gráfico 6.125: Resultados obtidos pelo refinamento para o *run2* do exemplo 12 (var. M).

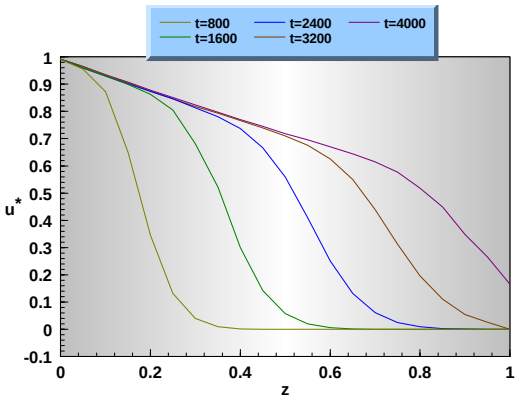

Gráfico 6.126: Resultados obtidos pelo refinamento para o *run2* do exemplo 12 (var. u^*).

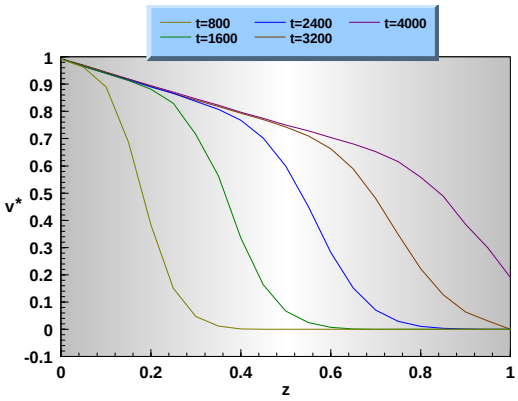

Gráfico 6.127: Resultados obtidos pelo refinamento para o *run2* do exemplo 12 (var. v^*).

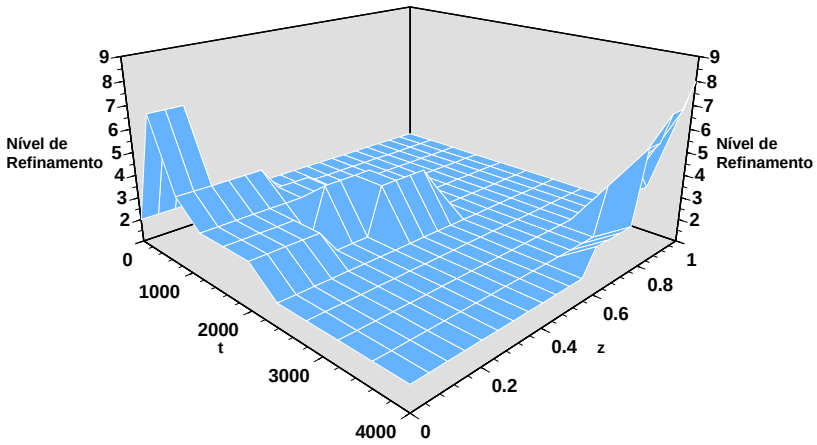

Gráfico 6.128: Resultados do nível de refinamento para o *run2* do exemplo 12.

Desta forma, verifica-se que, como em exemplos anteriores, as diferenças atrasadas melhoram substancialmente a performance do método.

A utilização de malhas não uniformes, não contribui de forma importante para facilitar a resolução do problema, já que nas zonas onde a concentração nodal se revelaria mais útil (área na vizinhança da fronteira esquerda do domínio espacial na fase de arranque do problema) o método não revela dificuldades excessivas de integração. Por outro lado, a extensão desse período é bastante reduzida em relação à do domínio temporal global. De qualquer modo, a aplicação de malhas não uniformes não conduz a vantagens apreciáveis, devido às características básicas da estratégia de refinamento, que já por si, procede à redução do passo espacial nos subdomínios seleccionados. De qualquer modo, a utilização deste tipo de malhas só seria útil, se fosse acompanhado por um procedimento de redefinição da malha base, de forma a que se adicionassem nodos nas áreas onde o refinamento fosse especialmente activo e se retirassem nas zonas onde o mesmo não se revelasse necessário.

Na tabela seguinte, resumem-se as principais performances computacionais referentes a cada um dos métodos considerados no presente trabalho.

Tabela 6.15: Desempenhos computacionais para o modelo da coluna de adsorção/reacção.

Algoritmo	***Tempo de Computação (s)***
Refinamento	400.9
M.E.F.M.[29]	217.7

Pela análise da Tabela 6.15, conclui-se que o M.E.F.M. possibilita a obtenção de resultados precisos com a utilização de tempos de computação relativamente inferiores. No entanto, verifica-se que, tanto num caso como no outro, o esforço computacional exigido para a resolução deste problema é reduzido. De qualquer modo, os perfis calculados através de refinamento são muito aceitáveis e o tempo de cpu associado à execução do método é bastante razoável.

7. Conclusões e Sugestões para Trabalho Futuro

7.1 – Conclusões

Através de uma análise global aos resultados apresentados no Capítulo anterior, torna-se possível chegar a algumas conclusões acerca da eficácia dos métodos enunciados neste trabalho, tendo como referência a performance da formulação do **M.E.F.M.** utilizada por *Duarte*[29]. Por outro lado, dada à experiência adquirida na utilização dos códigos desenvolvidos, salientam-se igualmente alguns pormenores na sua execução, que podem ajudar qualquer utilizador não familiarizado com as peculiaridades dos algoritmos, conduzindo, assim, a uma poupança de tempo considerável na sua adaptação.

De forma a testar a robustez dos métodos, estes foram sujeitos de forma sistemática a modelos de complexidade crescente, desde problemas de P.D.E.´s escalares até sistemas algébrico-diferenciais (estes últimos apenas foram resolvidos com o **Método de Refinamento** que foi sujeito a um teste mais exaustivo). Em todos os casos, principalmente no que diz respeito ao **Método de Malha Móvel**, os algoritmos revelaram desempenhos aceitáveis, quer em relação à precisão dos resultados, como do esforço computacional requerido. Particularmente, o **Método de Malha Móvel** mostrou-se bastante eficaz para uma larga gama de equações hiperbólicas e parabólicas e para problemas com condições fronteira móveis e evolutivas. O **Método de Refinamento** implicou maiores problemas na sua aplicação. No entanto, verifica-se que a maioria das dificuldades poderiam ser ultrapassadas com a introdução de uma estratégia de tratamento das condições fronteira dos subproblemas mais complexa e precisa. Uma alternativa razoável, consistiria no recurso a um procedimento semelhante ao utilizado no **Método de Malha Móvel**, onde esses problemas não são sentidos.

No que concerne à robustez de cada um dos métodos enunciados, constatam-se as seguintes conclusões:

① **Método de Refinamento** ⇨ Os perfis de solução obtidos são, comparativamente, de qualidade inferior para problemas caracterizados pelo desenvolvimento de frentes abruptas. O método demonstra algumas dificuldades na descrição de declives elevados, observando-se, no geral, a ocorrência de dissipação numérica. Nas proximidades das ondas, verifica-se a introdução de oscilações suaves nos perfis. A diluição destes problemas é conseguida com o aumento da concentração nodal da malha base, o que conduz irremediavelmente, a um aumento significativo dos tempos de execução. No entanto, este tipo de anomalias apenas ocorre em casos onde existam variações temporais bruscas da solução. Assim, o método não apresenta dificuldades na reprodução das frentes abruptas quando estas se desenvolvem gradualmente.

A principal dificuldade no avanço da integração observa-se em casos que envolvam o movimento de ondas, após o seu embate em fronteiras descritas por condições de *Neumann*. Nestes problemas, verifica-se a introdução de grande instabilidade em perfis bastante suaves, o que obriga ao refinamento crescente de regiões alargadas do domínio. Deste modo, observa-se um aumento drástico do esforço computacional nestes casos. A solução deste problema passa pela utilização de discretizações de diferenças finitas descentradas, cuja direcção dominante seja oposta ao movimento das ondas. No entanto, existe sempre a possibilidade de ocorrência de distorções nos perfis, junto às fronteiras referidas, normalmente devidas a

excessivo refinamento, provocado pela escolha inadequada dos parâmetros de entrada. Deste modo, o ajuste destes revela-se essencial à obtenção de resultados aceitáveis.

Estes problemas devem-se essencialmente, à simplicidade, e consequente imprecisão, da estratégia de tratamento fronteiras dos subproblemas gerados pelo refinamento sucessivo da malha base. Esta estratégia recorre à fixação de condições fronteira de *Dirichlet* artificiais, que apesar de evitarem a ocorrência de cortes na solução, podem provocar distorções visíveis nos perfis, entre zonas espaciais integradas através de malhas de níveis diferentes.

Por outro lado, para modelos que apresentam perfis de solução suaves, o que implica menores dificuldades de integração, o método revela-se bastante robusto, obtendo-se resultados muito semelhantes aos do **M.E.F.M.**, com tempos de computação consideravelmente inferiores. Este facto deve-se, essencialmente, à grande simplicidade do algoritmo, provando que este é o mais adequado para a resolução desse tipo de modelos. No caso dos restantes exemplos, os resultados obtidos são de menor qualidade, exigindo normalmente, esforços computacionais superiores, mas dentro dos limites do razoável.

② **Método de Malha Móvel** ⇨ Os resultados são, na generalidade de elevada qualidade e muito semelhantes aos obtidos pelo **M.E.F.M.**, obtendo-se tempos de cpu da mesma ordem de grandeza em relação a esse método. No entanto, é importante notar que este método foi apenas aplicado a alguns dos exemplos que colocaram algumas dificuldades na aplicação do **Método de Refinamento**. Observa-se uma boa definição na reprodução dos declives abruptos, com a ocorrência de oscilações muito ligeiras em alguns casos específicos, que desaparecem com o decorrer da integração.

Assim, o algoritmo revela-se bastante eficaz em todos os exemplos testados, possibilitando resultados consideravelmente melhores do que o método anterior. A estratégia de tratamento das fronteiras dos subproblemas, baseada em aproximações lineares, mostra-se bastante adequada, desaparecendo os problemas verificados anteriormente, relacionados com a ocorrência de descontinuidades entre regiões do domínio espacial integradas de forma diferente.

Apesar disso, este método é mais complexo que o anterior, já que implica a utilização de equações diferenciais de movimentação nodal, responsáveis pelo aumento significativo da não linearidade do problema original. No entanto, do ponto de vista formal, é ainda bastante mais simples do que qualquer formulação do **M.E.F.M.**.

Em alguns casos, observa-se um aumento apreciável do tempo de computação associado à aplicação do algoritmo (nomeadamente na passagem de problemas simples para sistemas de duas P.D.E.´s, quando comparados com os tempos correspondentes obtidos com o **M.E.F.M.**). Este acréscimo pode ser explicado através da estrutura do próprio método:

❶ O facto do presente método ser multi-malha, o que leva à necessidade de integração de várias malhas fixas referentes a cada variável durante a fase de estimativa do erro espacial.

❷ As características iterativas do esquema de integração dos subproblemas gerados. Como se permite uma variação controlada dos valores da solução nas fronteiras dos subdomínios, é necessário assegurar a convergência dos resultados obtidos pela integração dos subproblemas em relação aos perfis globais calculados com as malhas fixas na primeira fase. Em alguns problemas, verificam-se algumas dificuldades na obtenção dessa convergência, vendo-se o algoritmo obrigado a expandir consideravelmente os subdomínios inicialmente seleccionados, o que implica um aumento significativo do tempo de cpu, em cada passo.

Por isso, devido à ocorrência do problema ❷, é conveniente utilizar, à partida, tolerâncias bastante apertadas, possibilitando que os subdomínios originalmente seleccionados sejam suficientemente alargados e acelerando o esquema iterativo na obtenção de convergência em ambas as fronteiras. É por essa razão que, para o mesmo exemplo, as tolerâncias absolutas utilizadas na aplicação do **Método da Malha Móvel** são, normalmente mais reduzidas do que as correspondentes ao **Método de Refinamento**.

Devido à não introdução, de momento, de uma estratégia de remoção nodal no procedimento de redefinição das malhas base (que só permite a adição de nodos adicionais intermédios), a presente formulação do método não se revela muito eficaz na resolução de modelos que apresentem soluções evolutivas de características muito diferentes para períodos temporais distintos (nomeadamente, modelos que desenvolvam frentes que se propagam a velocidades muito díspares). Assim, é possível que a integração de uma zona do domínio temporal onde ocorram dificuldades importantes no avanço da integração, promova a construção de malhas base muito densas, que não serão redefinidas correctamente para zonas posteriores, onde esses problemas já não se farão sentir, conduzindo deste modo a um aumento desnecessário do esforço computacional associado à resolução do problema global.

As características multi-malha do **Método da Malha Móvel** não são originadas pelo tipo de equações de movimentação nodal utilizadas, já que estas incluem simultaneamente contribuições de todas as variáveis do modelo, descrevendo, assim, apenas uma malha. A separação das diversas malhas realiza-se na fase de estimativa do erro, onde o algoritmo pode seleccionar subdomínios distintos para cada variável, através da comparação dos erros em cada nodo. Assim, geram-se subproblemas diferentes, formando-se uma malha para cada variável. No entanto, os algoritmos multi-malha pressupõem algumas desvantagens importantes como: o aumento do esforço computacional para a integração nas diferentes malhas; necessidade de um maior número de interpolações de forma a se estabelecer as comunicações entre as diversas malhas.

É possível aplicar igualmente, ao **Método de Refinamento**, uma estratégia de separação de malhas, já que a fase de estimativa do erro é equivalente para os dois métodos. No entanto, tal só faria sentido se se adicionasse ao algoritmo um procedimento de introdução e remoção nodal que possibilitasse o desenvolvimento de malhas separadas entre as variáveis, conforme as propriedades dos perfis. Para o presente algoritmo, as variáveis utilizam sempre a mesma malha base na transição entre dois passos temporais, não fazendo qualquer sentido proceder a uma separação das malhas. Assim, para cada passo, um nodo é seleccionado, quando apenas uma das variáveis não satisfaz o erro máximo estabelecido.

Todas as execuções efectuadas neste trabalho foram realizadas com aproximações de diferenças finitas de cinco pontos. Considerou-se que este número de pontos possibilitaria um maior equilíbrio entre discretizações demasiado simples e excessivamente complexas. Os resultados obtidos são, no geral, satisfatórios, verificando-se melhores performances dos métodos, para diferenças descentradas que acompanhem o movimento das ondas, tanto do ponto de vista da maior precisão dos resultados, como do menor esforço computacional exigido.

No caso de modelos onde ocorra uma grande disparidade na grandeza absoluta dos declives dos perfis de solução, ou seja, verifica-se a formação de frentes abruptas entre zonas praticamente constantes, conclui-se que a aplicação de interpolações lineares se revela bastante adequada, como seria de esperar. As interpolações com *Splines* cúbicas introduzem

oscilações artificiais nos perfis, sendo mais eficazes em casos onde as soluções apresentem características curvilíneas.

No **Método de Refinamento**, a utilização de interpolações é apenas necessária para a avaliação da solução inicial em cada passo temporal, para as malhas de refinamento de nível superior a dois. Os problemas originados por imprecisões de interpolação apenas se tornam visíveis se se utilizarem malhas base demasiado esparsas.

Para o caso do **Método da Malha Móvel**, as operações de interpolação assumem papel mais importante na execução do algoritmo, já que se tornam necessárias em diversos procedimentos:

❶ Transposição da solução entre cada uma das malhas (apenas necessário no caso de sistemas de P.D.E.´s);
❷ Estimativa da evolução da solução nas fronteiras dos subdomínios (interpolações sempre do tipo linear);
❸ Redefinição das malhas, quer no que diz respeito ao afastamento de nodos demasiado próximos, como na introdução de nodos adicionais;
❹ Transposição da solução para a malha base inicial (quando essa opção se encontra activada, o que implica a não redefinição das malhas).

Deste modo, verifica-se que os erros associados à interpolação são mais importantes neste método. No entanto, os efeitos de acumulação sucessiva deste tipo de erros nunca se evidenciaram como preponderantes, em qualquer dos exemplos resolvidos neste trabalho.

De forma a se possibilitar um correcto desempenho de cada algoritmo, é absolutamente crítico ajustar convenientemente alguns dos parâmetros de entrada. No que diz respeito ao **Método de Refinamento**, verifica-se que a manipulação de parâmetros como: o passo temporal base; as tolerâncias e os passos locais espaciais base, são extremamente importantes para a obtenção de perfis aceitáveis. Estes factores são ajustados de forma a assegurar um equilíbrio que impeça o método de sub- ou sobrerefinar a malha, o que pode ocasionar o desenvolvimento de perfis imprecisos como a ocorrência de distorções e oscilações. Como seria de esperar, os valores escolhidos para as tolerâncias são especialmente críticos, já que, para valores demasiado elevados, verificam-se problemas de instabilidade na solução, enquanto que se a exigência de precisão fôr excessivamente apertada, constata-se o aparecimento de anomalias entre regiões integradas com malhas de nível diferente.

Para o **Método da Malha Móvel**, conclui-se que os parâmetros mais importantes são: factor de viscosidade internodal (λ); espaçamentos mínimo e máximo entre nodos adjacentes. No caso de um determinado modelo não necessitar da introdução de valores de λ elevados, verifica-se que o parâmetro crítico a ajustar é o espaçamento mínimo. Tal deve-se ao facto de, ao se dar maior liberdade de movimentação aos nodos, estes terem a tendência de se aproximar mais rapidamente, formando malhas muito apertadas em algumas zonas espaciais, o que pode provocar dificuldades no desempenho do integrador temporal, devido a um aumento da razão de *stiffness* do sistema algébrico diferencial. No caso da evolução nodal ser obrigatoriamente mais lenta, é necessário elevar o valor de λ, de forma a se acompanhar convenientemente o deslocamento das ondas. Para modelos onde as ondas desenvolvidas percorrem todo o domínio, revela-se vantajoso concentrar inicialmente os nodos da malha base numa das extremidades do domínio. Nestes caso, é conveniente ajustar o valor de espaçamento máximo internodal, de forma a impedir o afastamento excessivo entre nodos consecutivos, que pode originar instabilidade nos perfis.

Os valores destes parâmetros têm de ser fixados arbitrariamente, em função da experiência do utilizador, já que não é possível estabelecer uma relação precisa entre o modelo em causa e os parâmetros adequados à sua resolução.

7.2 – Sugestões para Trabalho Futuro

O trabalho realizado na presente tese centrou-se principalmente na elaboração dos códigos gerais para aplicação dos algoritmos apresentados e o consequente teste das suas potencialidades na resolução de exemplos já sobejamente estudados e considerados como de difícil resolução. Os assuntos de investigação a ser abordados como continuação do trabalho realizado poderão passar pela exploração de diversos caminhos interessantes como: o estudo de estratégias alternativas para a resolução de alguns problemas focados, assim como a generalização dos algoritmos e dos códigos para modelos ainda não considerados. Alguns dos temas de estudo possíveis são enunciados em seguida, de uma forma mais específica e detalhada:

❶ Aplicação de algoritmos equivalentes a modelos com duas dimensões espaciais. A generalização da estratégia de refinamento, tal como é concebida a problemas bidimensionais é praticamente directa, enquanto que todas as equações de mobilidade nodal consideradas podem ser deduzidas em domínios bidimensionais de forma equivalente sem qualquer dificuldade.

❷ Introdução de estratégias alternativas no tratamento das condições fronteira dos subproblemas gerados pela fase de estimativa do erro, principalmente no caso do **Método de Refinamento**. Verificou-se que o recurso à fixação destes valores fronteira não conduz necessariamente ao desenvolvimento de perfis adequados. Desse modo, é possível concluir que o procedimento mais fiável será o estabelecimento de um acompanhamento da evolução temporal da solução nas fronteira, baseado em perfis discretos obtidos pela integração das malhas fixas. A partir daí, torna-se possível considerar várias possibilidades como:

☞ A evolução da solução nos pontos vizinhos, imediatamente exteriores ao subdomínio e seleccionados de modo a que a estimativa das derivadas espaciais seja, o mais possível realizada com a mesma fórmula de aproximação aplicada nos pontos interiores (estratégia aplicada no **Método de Malha Móvel** com bons resultados). Deste modo, possibilita-se uma variação controlada da solução nos pontos fronteira.

☞ A estimativa da evolução temporal da solução nos próprios pontos fronteira. Neste caso, as fórmulas de diferenciação espacial são aplicadas exactamente da mesma maneira que no domínio global.

Estas estratégias implicam, necessariamente, o recurso a interpolações no tempo de forma a avaliar a solução nos pontos intermédios, utilizados pelo integrador. A solução no conjunto de pontos, utilizados em cada operação de interpolação, tem de ser avaliada durante a fase de integração das malhas base fixas. Adicionalmente, é essencial efectuar a comparação entre os perfis calculados de forma diferente, de maneira a evitar a ocorrência de descontinuidades no perfil global.

❸ Aplicação dos códigos apresentados na resolução de problemas relevantes na área da Engenharia Química. Os exemplos resolvidos neste trabalho constituem problemas típicos já estudados, que permitem testar o desempenho dos algoritmos em condições bastante variadas e difíceis do ponto de vista numérico. Dado os resultados obtidos, é lógico que o passo seguinte seja a sua utilização em modelos mais complexos, com interesse prático.

❹ Teste de opções já existentes em cada código que não foram exploradas neste trabalho, tais como: a introdução da redefinição da malha base no **Método de Refinamento**, por introdução e remoção de nodos em zonas de maior e menor actividade do refinamento, respectivamente; a análise da influência do cálculo das derivadas de ordem superior a um, a partir das derivadas de ordem inferior; e a influência da subdivisão do intervalo temporal, de forma a que o acompanhamento da solução nas fronteiras dos subdomínios seja mais preciso, para o **Método de Malha Móvel**.

❺ Introdução de um algoritmo multi-malha, associado ao **Método de Refinamento**, que permita a separação das malhas referentes a cada variável. Esta alternativa só é viável se a opção de redefinição da malha base estiver activada, introduzindo uma certa mobilidade nodal estática. Caso contrário, todas as variáveis utilizam a mesma malha base de segundo nível, no arranque de cada passo temporal, não fazendo sentido proceder à criação de malhas múltiplas.

❻ Generalização dos algoritmos para opções alternativas de integração temporal, postas à disposição pela DASSL, que podem melhorar a performance do processo, como a possibilidade de definição analítica da matriz jacobiana,. Por outro lado, seria igualmente interessante, testar a utilização de integradores mais simples, já que as constantes reinicializações da integração, impostas por ambos os algoritmos, impossibilitam o aumento da ordem do método implícito de avanço temporal, utilizado pela DASSL.

❼ Extensão da aplicabilidade dos códigos a problemas caracterizados por derivadas temporais implícitas e derivadas espaciais de ordem superior a dois, além da generalização do código referente ao **Método de Malha Móvel**, a problemas algébrico-diferenciais. Estas alterações são relativamente simples de efectuar, devido à forma como a presente estrutura de cada código foi concebida.

❽ Introdução de um critério de detecção e posterior remoção dos nodos excedentários das malhas base no **Método de Malha Móvel**, que permita uma maior eficiência no controlo da densidade dessas malhas, após a resolução das zonas temporais mais problemáticas da integração global.

Notação Geral

EU – estimativa do erro de truncatura
floor – parâmetro de limite mínimo
n – nível de refinamento
N_{max} – nível de refinamento máximo
NP – número de nodos da malha base
NPDE – número de equações diferenciais parciais do modelo
t – variável temporal
Tcpu – tempo de computação
TOL – tolerância do método
tol INT – tolerância referente ao integrador
u – variáveis dependentes
u_z – primeira derivada espacial das variáveis dependentes
u_{zz} – segunda derivada espacial das variáveis dependentes
u_t – primeira derivada temporal das variáveis dependentes
w – parâmetro de peso
Wh – aproximação da solução na malha fina
W2h – aproximação da solução na malha larga
z – variável espacial
z^L – posição da fronteira esquerda
z^R – posição da fronteira direita

Letras do Alfabeto Grego

α – factor de mobilidade nodal
Δt – passo temporal
Δz – espaçamento internodal
Δz_{MIN} – espaçamento internodal mínimo
Δz_{MAX} – espaçamento internodal máximo
λ – coeficiente de viscosidade nodal

Convenções Gerais

$\dot{y}$ – derivada temporal de y
$\underline{y}$ – vector y
$\underline{\underline{A}}$ – matriz A
$\bullet$ – produto interno entre dois vectores
δ – derivada parcial

Bibliografia

[1] - Abbott, M. B. e D. R. Basco; *Computational Fluid Dynamics*, Longman Scientific & Technical, Singapore (1989).

[2] - Ablow, C. M. e S. Schechter; "Campylotropic Coordinates", *J. Comput. Phys.*, **27** (1978), p. 351.

[3] - Acharya, S. e F. H. Moukalled; "An Adaptive Grid Solution Procedure for Convection-Diffusion Problems", *J. Comput. Phys.*, **91** (1990), p. 32.

[4] - Adjerid, S. e J. E. Flaherty; "A Moving Finite Element Method with Error Estimation and Refinement for One-Dimensional Time Dependent Partial Differential Equations", *SIAM J. Numer. Anal.*, **23** (1986), p. 778.

[5] - Altas, I. e J. W. Stephenson; "A Two-Dimensional Adaptive Mesh Generation Method", *J. Comput. Phys.*, **94** (1991), p. 201.

[6] - Anderson, D. A.; "Adaptive Grid Methods for Partial Differential Equations", em *Advances in Grid Generation*, ed. K. N. Ghia e U. Ghia, ASME, New York (1983), p. 1.

[7] - Anderson, D. A.; "Application of Adaptive Grids to Transient Problems", em *Adaptive Computational Methods for Partial Differential Equations*, ed. I. Babuska, J. Chandra e J. E. Flaherty, SIAM, Philadelphia (1983), p. 208.

[8] - Anderson, D. A. e M. M. Rai; "The Use of Solution Adaptive Grids in Solving Partial Differential Equations", em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 317.

[9] - Anyiwo, J. C.; "Idealized Dynamic Grid Computational for Physical Systems", em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 837.

[10] - Arney, D. C. e J. E. Flaherty; "An Adaptive Local Mesh Refinement Method for Time-Dependent Partial Differential Equations", *Appl. Numer. Math.*, **5** (1989), p. 257.

[11] - Arney, D. C. e J. E. Flaherty; "A Two-Dimensional Mesh Moving Technique for Time-Dependent Partial Differential Equations", *J. Comput. Phys.*, **67** (1986), p. 124.

[12] - Bell, J., M. Berger, J. Saltzman e M. Welcome; "Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws", *SIAM J. Sci. Comput.*, **15** (1994), p. 127.

[13] - Bell, J. B. e G. R. Shubin; "An Adaptive Grid Finite Difference Method for

Conservation Laws", *J. Comput. Phys.*, **52** (1983), p. 569.

[14] - Berger, M. J. e P. Colella; "Local Adaptive Mesh Refinement for Shock Hydrodynamics", *J. Comput. Phys.*, **82** (1989), p. 64.

[15] - Berger, M. J. e J. Oliger; "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations", *J. Comput. Phys.*, **53** (1984), p. 484.

[16] - Blom, J. G., J. M. Sanz-Serna e J. G. Verwer; "On Simple Moving Grid Methods for One-Dimensional Evolutionary Partial Differential Equations", *J. Comput. Phys.*, **74** (1988), p. 191.

[17] - Brackbill, J. U.; "Coordinate System Control: Adaptive Meshes", em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 277.

[18] - Brackbill, J. U. e J. Saltzman; "An Adaptive Computational Mesh for the Solution of Singular Perturbation Problems", em *Numerical Grid Generation Techniques, Proceedings of the Numerical Grid Generation Workshop at Langley Research Center*, ed. R. E. Smith, NASA CP-2166, Washington DC (1980), p. 193.

[19] - Brackbill, J. U. e J. Saltzman; "Adaptive Zoning for Singular Problems in Two Dimensions", *J. Comput. Phys.*, **46** (1982), p. 342.

[20] - Brennan, K. E., S. L. Campbell e L. R. Petzold; *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, North-Holland, New York (1989).

[21] - Budd, C. J., W. Huang e R. D. Russell; "Moving Mesh Methods for Problems with Blow-up", *SIAM J. Sci. Comput.*, **17** (1996), p. 305.

[22] - Carcaillet, R., G. S. Dulikravich e S. R. Kennon; "Generation of Solution-Adaptive Computational Grids Using Optimization", *Comput. Meth. Appl. Mech. Eng.*, **57** (1986), p. 279.

[23] - Carey, G. F. e H. T. Dinh; "Grading Functions and Mesh Redistribution", *SIAM J. Numer. Anal.*, **22** (1985), p. 1028.

[24] - Chen, K.; "Error Equidistribution and Mesh Adaptation", *SIAM J. Sci. Comput.*, **15** (1994), p. 798.

[25] - Daripa, P.; "A New Theory for One-Dimensional Adaptive Grid Generation and its Applications", *SIAM J. Numer. Anal.*, **28** (1991), p. 1635.

[26] - DeBoor, C.; "Good Approximation by Splines with Variable Knots. II", *Conference on the Numerical Solution of Differential Equations. Lectures Notes in Mathematics*, Vol. 363, Springer, Berlin-Heidelberg-New York (1973), p. 12.

[27] - Denny, V. E. e R. B. Landis; "A New Method for Solving Two-Point Boundary Value Problems Using Optimal Node Distribution", *J. Comput. Phys.*, **9** (1972), p. 120.

[28] - Dorfi, E. A. e L. O´C. Drury; "Simple Adaptive Grids for 1-D Initial Value Problems", *J. Comput. Phys.*, **69** (1987), p. 175.

[29] - Duarte, B. P. M.; "Método dos Elementos Finitos Móveis Aplicado à Resolução de Modelos de Frente de Reacção", *Tese de Doutoramento*, Univ. de Coimbra, F.C.T, Coimbra (1994).

[30] - Dwyer, H. A.; "A Discussion of Some Criteria for the Use of Adaptive Gridding", em *Adaptive Computational Methods for Partial Differential Equations*, ed. I. Babuska, J. Chandra e J. E. Flaherty, SIAM, Philadelphia (1983), p. 111.

[31] - Dwyer, H. A.; "Grid Adaptation for Problems in Fluid Dynamics", *AIAA J.*, **22** (1984), p. 1705.

[32] - Dwyer, H. A., R. J. Kee, P. K. Barr e B. R. Sanders; "Transient Droplet Heating at High Peclet Number", *J. Fluids Eng.*, **105** (1983), p. 83.

[33] - Dwyer, H. A., R. J. Kee e B. R. Sanders; "An Adaptive Grid Method for Problems in Fluid Mechanics and Heat Transfer", *AIAA J.*, **18** (1980), p. 1205.

[34] - Dwyer, H. A., F. Raiszadeh e G. Otey; "A Study of Reactive Diffusion Problems with Stiff Integrators and Adaptive Grids", *Lectures Notes in Physics*, Vol. 141, Springer-Verlag, New York/Berlin (1981), p. 170.

[35] - Dwyer, H. A., B. R. Sanders e F. Raiszadeh; "Ignition and Flow Propagation Studies with Adaptive Numerical Grids", *Combust. Flame*, **52** (1983), p. 11.

[36] - Dwyer, H. A., M. D. Smooke e R. J. Kee; "Adaptive Gridding for Finite Difference Solutions to Heat and Mass Transfer Problems", em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 339.

[37] - Eiseman, P. R.; "Alternating Direction Adaptive Grid Generation", *AIAA J.*, **23** (1985), p. 551.

[38] - Eiseman, P. R.; "Adaptive Grid Generation", *Comput. Meth. Appl. Mech. Eng.*, **64** (1987), p. 321.

[39] - Eiseman, P. R.; "Adaptive Grid Generation by Mean-Value Relaxation", *J. Fluids Eng.*, **107** (1985), p. 477.

[40] - Eiseman, P. R.; "Solution Adaptivity Using a Triangular Mesh", *Lectures Notes in Physics*, Vol. 238, Springer-Verlag, New York/Berlin (1985), p. 205.

[41] - Eiseman, P. R.; "Alternating Direction Adaptive Grid Generation for Three-Dimensional Regions", *Lectures Notes in Physics*, Vol. 264, Springer-Verlag, New York/Berlin (1986), p. 258.

[42] - Ewing, R. E., R. D. Lazarov e A. T. Vassilev; "Finite Difference Scheme for Parabolic Problems on Composite Grids with Refinement in Time and Space", *SIAM J. Numer. Anal.*, **31** (1994), p. 1605.

[43] - Ewing, R. E., R. D. Lazarov e P. S. Vassilevski; "Finite Difference Schemes on Grids with Local Refinement in Time and Space for Parabolic Problems: I. Derivation, Stability and Error Analysis", *Computing*, **45** (1990), p. 193.

[44] - Finlayson, B. A.; *Numerical Methods for Problems with Moving Fronts*, Ravenna Park Publishing, Inc., Seattle (1992).

[45] - Fornberg, B.; "Generation of Finite Difference Formulas on Arbitrarily Spaced Grids", *Math. of Comput.*, **51** (1988), p. 699.

[46] - Fornberg, B.; "Fast Generation of Weights in Finite Difference Formulas", em *Recent Developments in Numerical Methods and Software for ODEs/DAEs/PDEs*, ed. G. D. Byrne e W. E. Schiesser, World Scientific, Singapore (1992), p. 97.

[47] - Fraga, E. S. e J. Ll. Morris; "An Adaptive Mesh Refinement Method for Nonlinear Dispersive Wave Equations", *J. Comput. Phys.*, **101** (1992), p. 94.

[48] - Fung, K.-Y., J. Tripp e B. Goble; "Adaptive Refinement with Truncation Error Injection", *Comput. Meth. Appl. Mech. Eng.*, **66** (1987), p. 1.

[49] - Gear, C. W.; *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, Englewood Cliffs, NJ (1971).

[50] - Gear, C. W.; "Simultaneous Numerical Solution of Differential-Algebraic Equations", IEEE Trans. Circuit Theory CT-18 (1971), p. 89.

[51] - Ghia, K. N., U. Ghia e C. T. Shin; "Adaptive Grid Generation for Flows with Local High Gradient Regions", em *Advances in Grid Generation*, ed. K. N. Ghia e U. Ghia, ASME, New York (1983), p. 35.

[52] - Gnoffo, P. A.; "A Vectorized, Finite Volume, Adaptive-Grid Algorithm for Navier-Stokes Calculations", em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 819.

[53] - Gnoffo, P. A.; "Finite Volume Adaptive Grid Algorithm Applied to Planetary Entry Flow Fields", *AIAA J.*, **21** (1983), p. 1249.

[54] - Gough, D. O., E. A. Spiegel e J. Toomre; "Highly Stretched Meshes as Functionals of Solutions", *Lectures Notes in Physics*, Vol. 35, Springer-Verlag,

New York/Berlin (1975), p. 191.

[55] - Greenberg, J. B.; "A New Self-Adaptive Grid Method", *AIAA J.*, **23** (1985), p. 317.

[56] - Gropp, W. D.; "A Test of Mesh Refinement for 2-D Scalar Hyperbolic Problems", *SIAM J. Sci. Stat. Comput.*, **1** (1980), p. 191.

[57] - Gropp, W. D.; "Local Uniform Mesh Refinement with Moving Grids", *SIAM J. Sci. Stat. Comput.*, **8** (1987), p. 292.

[58] - Guiné, R. P. F.; "Resolução de Sistemas de Equações Diferenciais de Derivadas Parciais – Algoritmos de Refinamento Espacial", *Tese de Mestrado*, Univ. de Coimbra, F. C. T., Coimbra (1996).

[59] - Hawken, D. F., J. J. Gottlieb e J. S. Hansen; "Review of Some Adaptive Node-Movement Techniques in Finite-Element and Finite-Difference Solutions of Partial Differential Equations", *J. Comput. Phys.*, **95** (1991), p. 254.

[60] - Hayes, L. J., S. R. Kennon e G. S. Dulikravich; "Grid Orthogonalization for Curvilinear Alternating-Direction Techniques", *Comput. Meth. Appl. Mech. Eng.*, **59** (1986), p. 141.

[61] - Herbst, B. M.; "Moving Finite Element Methods for the Solution of Evolution Equations", *PhD Thesis*, University of the Orange Free State (1982).

[62] - Hindman, R. G., P. Kutler e D. Anderson; "Two-Dimensional Unsteady Euler-Equation Solver for Arbitrarily Sharped Flow Regions", *AIAA J.*, **19** (1981), p. 424.

[63] - Hindman, R. G. e J. Spencer; "A New Approach to Truly Adaptive Grid Generation", AIAA Paper 83-0450 (1983), p. 1.

[64] - Huang, W., Y. Ren e R. D. Russell; "Moving Mesh Methods Based on Moving Mesh Partial Differential Equations", *J. Comput. Phys.*, **113** (1994), p. 279.

[65] - Huang, W., Y. Ren e R. D. Russell; "Moving Mesh Partial Differential Equations (MMPDEs) Based on the Equidistribution Principle", *SIAM J. Numer. Anal.*, **31** (1994), p. 709.

[66] - Huang, W. e R. D. Russell; "Analysis of Moving Mesh Partial Differential Equations with Spatial Smoothing", *SIAM J. Numer. Anal.*, **34** (1997), p. 1106.

[67] - Huang, W. e D. M. Sloan; "A Simple Adaptive Grid Method in Two Dimensions", *SIAM J. Sci. Comput.*, **15** (1994), p. 776.

[68] - Hyman, J. M.; "Adaptive Moving Mesh Methods for Partial Differential Equations", Los Alamos National Laboratory Report LA-UR-82-3690, Los Alamos, New Mexico (1982) (não publicado).

[69] - Hyman, J. M. e B. Larrouturou; "Dynamic Rezone Methods for Partial Differential Equations in One Space Dimension", *Appl. Numer. Math.*, **5** (1989), p. 435.

[70] - Hyman, J. M. e M. Naughton; "Static Rezone Methods for Tensor Product Grids, Large Scale Computations in Fluid Mechanics", *Lectures in Applied Mathematics*, Amer. Math. Soc., Providence (1985), p. 321.

[71] - Jarvis, R. B. e C. C. Pantelides; "DASOLV – A Differential-Algebraic Equation Solver – Version 1.2", Center for Process System Engineering, Imperial College of Sciences - Technology and Medicine, London (1992).

[72] - Johnson, I. W.; "Moving Finite Elements for Diffusion Problems", *PhD Thesis*, University of Reading (1986).

[73] - Kansa, E. J., D. L. Morgan e L. K. Morris; "A Simplified Moving Finite Difference Scheme: Application to Gas Dispersion", *SIAM J. Sci. Stat. Comput.*, **5** (1984), p. 667.

[74] - Kennon, S. R. e G. S. Dulikravich; "Generation of Computational Grids Using Optimization", *AIAA J.*, **24** (1986), p. 1069.

[75] - Klopfer, G. H. e D. S. McRae; "The Nonlinear Modified Equation Approach to Analyzing Finite Difference Schemes", AIAA Paper 81-1029 (1981), p. 317.

[76] - Kreis, R. I., F. C. Thames e H. A. Hassan; "Application of a Variational Method for Generating Adaptive Grids", *AIAA J.*, **24** (1986), p. 404.

[77] - Kulkarni, M. S. e M. P. Dudukovic; "A Robust Algorithm for Fixed-Bed Reactors with Steep Moving Temperature and Reaction Fronts", *Chem. Eng. Sci.*, **51** (1996), p. 571.

[78] - Larrouturou, B.; "A Conservative Adaptive Method for Flame Propagation", *SIAM J. Sci. Stat. Comput.*, **10** (1989), p. 742.

[79] - Lee, D. e Y. M. Tsuei; "A Hybrid Adaptive Gridding Procedure for Recirculating Fluid Flow Problems", *J. Comput. Phys.*, **108** (1993), p. 122.

[80] - MacCormack, R. W.; "The Effect of Viscosity in Hypervelocity Impact Cratering", AIAA Paper 69-354 (1969), p. 1.

[81] - Madsen, N. K.; "Molag: A Method of Lines Adaptive Grid Interface for Nonlinear Partial Differential Equations", em *PDE Software: Modules, Interfaces and Systems*, ed. B. Engquist e T. Smedsaas, North-Holland, Amsterdam (1984), p. 207.

[82] - Matsuno, K. e H. A. Dwyer; "Adaptive Methods for Elliptic Grid Generation", *J. Comput. Phys.*, **77** (1988), p. 40.

[83] - Miller, K. e R. Miller; "Moving Finite Elements I", *SIAM J. Numer. Anal.*, **18** (1981), p. 1019.

[84] - Miller, K.; "Moving Finite Elements II", *SIAM J. Numer. Anal.*, **18** (1981), p. 1033.

[85] - Nakahashi, K. e G. S. Deiwert; "Three-Dimensional Adaptive Grid Method", *AIAA J.*, **24** (1986), p. 948.

[86] - Nakahashi, K. e G. S. Deiwert; "A Practical Adaptive-Grid Method for Complex Fluid-Flow Problems", *Lectures Notes in Physics*, Vol. 218, Springer-Verlag, New York/Berlin (1985), p. 422.

[87] - Nowak, U., J. Frauhammer e U. Nieken; "A Fully Adaptive Algorithm for Parabolic Partial Differential Equations in One Space Dimension", *Comp. & Chem. Eng.*, **20** (1996), p. 547.

[88] - Oliveira, F., P. de Oliveira, M. C. Lopes e J. A. A. M. Castro; "Two Adaptive Grid Methods for Fixed Bed Systems Simulation", *Comp. & Chem. Eng.*, **18** (1994), p. 227.

[89] - Pereyra, V. e E. G. Sewell; "Mesh Selection for Discrete Solution of Boundary Problems in Ordinary Differential Equations", *Numer. Math.*, **23** (1975), p.261.

[90] - Petzold, L. R.; "A Description of DASSL: A Differential/Algebraic System Solver", Sandia Tech. Rep. 82-8637 (1982).

[91] - Petzold, L. R.; "Observations on an Adaptive Moving Grid Method for One-Dimensional Systems of Partial Differential Equations", *Appl. Numer. Math.*, **3** (1987), p. 347.

[92] - Pierson, B. L. e P. Kutler; "Optimal Node Point Distribution for Improved Accuracy in Computational Fluid Dynamics", *AIAA J.*, **18** (1980), p. 49.

[93] - Pipilis, K. G.; "Higher Order Moving Finite Element Methods for Systems Described by Partial Differential-Algebraic Equations", *PhD Thesis*, Dept. of Chem. Eng. and Chem. Techn., Imperial College of Sciences - Technology and Medicine, London (1991).

[94] - Potter, D. E. e G. H. Tuttle; "The Construction of Discrete Orthogonal Coordinates", *J. Comput. Phys.*, **13** (1973), p. 483.

[95] - Quinta Ferreira, R. M.; "Contribuições para o Estudo de Reactores Catalíticos de Leito Fixo – Efeito da Convecção em Catalisadores de Poros Largos e Casos de Catalisadores Bidispersos", *Tese de Doutoramento*, Univ. do Porto, (1988).

[96] - Rai, M. M. e D. A. Anderson; "Application of Adaptive Grids to Fluid Flow Problems with Asymptotic Solutions", *AIAA J.*, **20** (1982), p. 496.

[97] - Rai, M. M. e D. A. Anderson; ”Grid Evolution in Time Asymptotic Problems”, em *Numerical Grid Generation Techniques, Proceedings of the Numerical Grid Generation Workshop at Langley Research Center*, ed. R. E. Smith, NASA CP-2166, Washington DC (1980), p. 409.

[98] - Ren, Y. e R. D. Russell; “Moving Mesh Techniques Based Upon Equidistribution, and their Stability”, *SIAM J. Sci. Stat. Comput.*, **13** (1992), p. 1265.

[99] - Revilla, M. A.; “Simple Time and Space Adaptation in One-Dimensional Evolutionary Partial Differential Equations”, *Int. J. Numer. Meth. Eng.*, **23** (1986), p. 2263.

[100] - Russell, R. D. e J. Christiansen; “Adaptive Mesh Selection Strategies for Solving Boundary Value Problems”, *SIAM J. Numer. Anal.*, **15** (1978), p. 59.

[101] - Saltzman, J. e J. U. Brackbill; ”Applications and Generalizations of Variational Methods for Generating Adaptive Meshes”, em *Numerical Grid Generation*, ed. J. F. Thompson, North-Holland, Amsterdam (1982), p. 865.

[102] - Sanz-Serna, J. M. e I. Christie; “A Simple Adaptive Technique for Non-linear Wave Problems”, *J. Comput. Phys.*, **67** (1986), p. 348.

[103] - Schiesser, W. E.; *The Numerical Method of Lines: Integration of Partial Differential Equations*, Academic Press, Inc., San Diego (1991).

[104] - Sereno, C. A.; ”Método dos Elementos Finitos Móveis: Aplicações em Engenharia Química”, *Tese de Doutoramento*, Univ. do Porto, (1989).

[105] - Smooke, M. D. e M. L. Koszykowski; “Fully Adaptive Solutions of One-Dimensional Mixed Initial-Boundary Value Problems with Applications to Unstable Problems in Combustion”, *SIAM J. Sci. Stat. Comput.*, **7** (1986), p. 301.

[106] - Strain, J.; “Fast Adaptive Methods for the Free-Space Heat Equation”, *SIAM J. Sci. Comput.*, **15** (1994), p. 185.

[107] - Thompson, J. F.; “Grid Generation Techniques in Computational Dynamics”, *AIAA J.*, **22** (1984), p. 1505.

[108] - Thompson, J. F.; “A Survey of Dynamically-Adaptive Grids in the Numerical Solution of Partial Differential Equations”, *Appl. Numer. Math.*, **1** (1985), p. 3.

[109] - Thompson, J. F., Z. U. A. Warsi e C. W. Mastin; “Boundary-Fitted Coordinate Systems for Numerical Solution of Partial Differential Equations – A Review”, *J. Comput. Phys.*, **47** (1982), p. 1.

[110] - Thompson, J. F., Z. U. A. Warsi e C. W. Mastin; *Numerical Grid Generation*,

North- Holland, New York (1985).

[111] - Trangenstein, J. A.; "Adaptive Mesh Refinement for Wave Propagation in Nonlinear Solids", *SIAM J. Sci. Comput.*, **16** (1995), p. 819.

[112] - Trompert, R. A. e J. G. Verwer; "A Static-Regridding Method for Two-Dimensional Parabolic Partial Differential Equations", *Appl. Numer. Math.*, **8** (1991), p. 65.

[113] - Trompert, R. A. e J. G. Verwer; "Analysis of the Implicit Euler Local Uniform Grid Refinement Method", *SIAM J. Sci. Comput.*, **14** (1993), p. 259.

[114] - Turkel, E.; "Progress in Computational Physics", *Comput. Fluids*, **11** (1984), p. 121.

[115] - Verwer, J. G., J. G. Blom, R. M. Furzeland e P. A. Zegeling; "A Moving Grid Method for One-Dimensional PDEs Based on the Method of Lines", em *Adaptive Methods for Partial Differential Equations*, ed. J. E. Flaherty, P. J. Paslow, M. S. Shephard e J. D. Vasilakis, SIAM, Philadelphia (1989), p. 160.

[116] - Verwer, J. G., J. G. Blom e J. M. Sanz-Serna; "An Adaptive Moving Grid Method for One-Dimensional Systems of Partial Differential Equations", *J. Comput. Phys.*, **82** (1989), p. 454.

[117] - White Jr., A. B.; "On Selection of Equidistributing Meshes for Two-Point Boundary-Value Problems", *SIAM J. Numer. Anal.*, **16** (1979), p. 472.

[118] - White Jr., A. B.; "On the Numerical Solution of Initial/Boundary-Value Problems in One Space Dimension", *SIAM J. Numer. Anal.*, **19** (1982), p. 683.

[119] - Winkler, K.-H. A., D. Mihalas e M. L. Norman; "Adaptive-Grid Methods with Asymmetric Time-Filtering", *Comput. Phys. Comm.*, **36** (1985), p. 121.

[120] - Winkler, K.-H. A., M. L. Norman e D. Mihalas; "Implicit Adaptive-Grid Radiation Hydrodynamics", em *Multiple Time Scales*, ed. J. U. Brackbill e B. I. Cohen, Academic Press, London (1985), p. 145.

[121] - Winkler, K.-H. A., M. L. Norman e M. J. Newman; "Adaptive Mesh Techniques for Fronts in star Formation", *Phys. D*, **12** (1984), p. 408.

[122] - Winslow, A. M.; "Numerical Solution of the Quasilinear Poisson Equation in a Nonuniform Triangle Mesh", *J. Comput. Phys.*, **1** (1966), p. 149.

[123] - Yanenko, N. N., V. D. Liseikin, V. M. Kovenya, V. M. Fomin e E. V. Vorozhtsov; "On Some Methods for the Numerical Simulation of Flows with Complex Structure", *Comput. Meth. Appl. Mech. Eng.*, **17/18** (1979), p. 659.

[124] - Yanenko, N. N., E. A. Kroshko, V. D. Liseikin, V. M. Fomin, V. P. Shapeev e Y. A. Shitov; "Methods for the Construction of Moving Grids for Problems of Fluid

Dynamics with Big Deformation", *Lectures Notes in Physics*, Vol. 59, Springer-Verlag, New York/Berlin (1976), p. 454.

[125] - Yanenko, N. N., V. D. Liseikin e V. M. Kovenya; "The Method of the Solution of Gaz Dynamical Problems in Moving Meshes", *Lectures Notes in Physics*, Vol. 91, Springer-Verlag, New York/Berlin (1979), p. 48.

Apêndice A - Apresentação Resumida do Integrador Implícito DASSL

O integrador implícito DASSL – *Differential-Algebraic System Solver* (*Petzold*[90], *Brennan et al*[20]) foi desenvolvido para a resolução de sistemas de equações algébrico-diferenciais da forma:

$$F(u,u',t) = 0\ , \qquad t \in [t_0, T] \tag{A.1}$$

sendo:

u(t) - vector estado, função da variável independente t (que geralmente, representa o tempo ou uma dimensão espacial;

$u' = \frac{d\,u}{d\,t}$ - primeira derivada de u em ordem a t.

A DASSL foi especialmente concebida para lidar com sistemas mistos de equações diferenciais e algébricas com características *stiff* e aplica uma estratégia preditiva-correctiva de ordem e passo variáveis, desenvolvida por *Gear*[50]. O vector de derivadas u'_{n+1} (correspondente ao tempo t_{n+1}) é aproximado, no estágio corrector, por uma fórmula de diferenças finitas *backward*, sendo o sistema de equações algébricas resultante, resolvido para o vector estado u_{n+1}, através de um método de *Newton* modificado. O tamanho do passo e a ordem de integração são ajustados automaticamente, de forma a possibilitar uma melhoria na performance do processo, dentro dos limites determinados pela tolerância definida por parte do utilizador, para o erro admissível em cada passo.

No que concerne aos parâmetros directamente relacionados com o integrador (vd. Apêndice D), os códigos desenvolvidos neste trabalho, apenas requerem, por parte do utilizador, a definição das respectivas tolerâncias de integração, que de uma forma geral, limitam o máximo erro tolerável na execução de cada passo, da forma:

$$|\text{Erro Local}| \le \text{ATOL} + \text{RTOL} \times |u| \tag{A.2}$$

em que:

ATOL – tolerância absoluta;

RTOL – tolerância relativa.

Em cada passo, as chamadas do integrador realizadas a partir do código, são efectuadas na sua forma *standard*, ou seja, sem que qualquer das opções adicionais, possíveis de utilizar, esteja activada (INFO(I) = 0, I = 1, … , 15). Este facto implica que, entre outras coisas, a matriz jacobiano das derivadas parciais não seja fornecido na sua forma analítica, tendo obrigatoriamente de ser aproximada pelo integrador através de diferenças finitas, para além de outros pormenores que, em casos particulares, poderiam melhorar o desempenho do integrador.

A aplicação da DASSL aos métodos apresentados neste trabalho poderá não ser a ideal, já que estes implicam uma constante reinicialização da integração. Assim, o integrador não tem possibilidades de aumentar a ordem da integração, o que conduz à utilização constante do método de *Euler backward* de passo variável. Deste modo, verifica-se que as peculiaridades de ambos os métodos estudados não possibilitam o aproveitamento das totais potencialidades do integrador.

Apêndice B - Descrição Resumida do Método de Elementos Finitos Móveis (M.E.F.M.)

O objectivo deste Apêndice consiste na apresentação de uma panorâmica geral sobre o **Método de Elementos Finitos Móveis** (M.E.F.M.), referido frequentemente nesta tese, porque se trata do método escolhido como referência, na avaliação da performance dos algoritmos desenvolvidos no presente trabalho. Não se pretende aqui efectuar uma descrição muito pormenorizada das diferentes formulações que têm sido apresentadas ao longo do tempo, referentes a este método. Este trabalho já foi anteriormente realizado através da revisão bibliográfica apresentada por *Duarte*[29] e um número considerável de outros autores.

A formulação inicial do M.E.F.M. foi proposta por *K. Miller* e *R. N. Miller* em 1981[83,84]. Originalmente, o processo foi desenvolvido para a resolução de P.D.E.´s evolutivas unidimensionais, sujeitas a condições fronteira de *Dirichlet*. Para tal, considera-se a P.D.E.,

$$u_t = Lu \tag{B.1}$$

válida no domínio fixo $a \le z \le b$, caracterizada por condições fronteira fixas de *Dirichlet*, sendo L um operador espacial não linear.

A técnica de discretização de elementos finitos aproxima a solução em cada elemento através da expressão:

$$U(z,t) = \sum_{j=1}^{N+1} U_j \cdot \alpha_j(z, \underline{s}(t)) \tag{B.2}$$

onde:

$\underline{U} = [U_0, U_1, \cdots, U_{N+1}]$ - vector da solução nos nodos;

$\underline{s} = [s_0, s_1, \cdots, s_{N+1}]$ - vector das posições nodais;

α_j - funções base elementares implicitamente dependentes do tempo através das posições nodais;

z - posição espacial.

Por derivação de (B.2) em ordem ao tempo obtém-se:

$$U_t(z,t) = \sum_{j=1}^{N+1} \dot{U}_j(t) \cdot \alpha_j(z, \underline{s}(t)) + \dot{s}_j(t) \cdot \beta_j(z, \underline{U}(t), \underline{s}(t)) \tag{B.3}$$

em que β_j são igualmente funções base, definidas por:

$$\beta_j(z, \underline{U}(t), \underline{s}(t)) = \frac{\delta U}{\delta s_j} \tag{B.4}$$

Deste modo, torna-se necessário para a implementação deste método, recorrer a uma estratégia de aproximação, de forma a avaliar os valores de β_j. Inicialmente, o método formulado baseava-se em aproximações lineares (linhas rectas poligonais) pelo que:

$$\beta_j = -U_z \cdot \alpha_j \tag{B.5}$$

Assim, as funções α_j e β_j são definidas por:

$$\alpha_j = \begin{cases} \dfrac{z - s_{j-1}}{s_j - s_{j-1}}, & s_{j-1} \le z \le s_j \\ \dfrac{s_{j+1} - z}{s_{j+1} - s_j}, & s_j \le z \le s_{j+1} \\ 0, & -\infty < z < s_{j-1} \quad \text{e} \quad s_{j+1} < z < +\infty \end{cases} \tag{B.6}$$

e

$$\beta_j = \begin{cases} -m_j \cdot \dfrac{z - s_{j-1}}{s_j - s_{j-1}}, & s_{j-1} \le z \le s_j \\ -m_{j+1} \cdot \dfrac{s_{j+1} - z}{s_{j+1} - s_j}, & s_j \le z \le s_{j+1} \\ 0, & -\infty < z < s_{j-1} \quad \text{e} \quad s_{j+1} < z < +\infty \end{cases} \tag{B.7}$$

onde $m_j = \dfrac{U_j - U_{j-1}}{s_j - s_{j-1}}$ corresponde ao declive da solução no elemento finito j estimado por diferenças finitas.

A solução da P.D.E. ($\underline{U}$) e a nova malha ($\underline{s}$) são determinadas por minimização da norma quadrada L_2 dos resíduos da aproximação à solução em todo o domínio, em ordem das derivadas temporais da solução nos nodos e das velocidades nodais: $\left\| U_t - LU \right\|_{L_2}^2$.

A substituição das derivadas pela correspondentes aproximações resulta num sistema de O.D.E.´s do tipo:

$$\underline{\underline{A}}(\underline{y}) \cdot \dot{\underline{y}} = \underline{g}(\underline{y}) \tag{B.8}$$

em que:

$\underline{y} = [U_1, s_1, \cdots, U_N, s_N]^T$

$\underline{\underline{A}}$ - matriz de massas quadrada e tridiagonal, constituída por blocos $\underline{\underline{A}}_{k,k}$ de dimensões 2×2, que mantêm uma estrutura constante no tempo.

É possível que, em determinadas circunstâncias, o determinante de qualquer bloco $\underline{\underline{A}}_{k,k}$ se torne nulo, tornando o sistema (B.8) indeterminado e consequentemente, de resolução impossível. As condições que provocam a degenerescência do sistema de O.D.E.´s podem ser de dois tipos distintos:

❶ Choque Nodal ⇨ consiste na coalescência de um nodo com o seu precedente.

❷ Paralelismo Elementar ⇨ definido pela colinearidade das aproximações da solução em dois elementos consecutivos.

Cada uma destas singularidades dão origem a equações linearmente dependentes que tornam o sistema (B.8) irresolúvel. De forma a ultrapassar estes problemas, *Miller*[84] recorre à introdução de funções de penalização na função objectivo de forma a adicionar forças de viscosidade internodal que regularizem o movimento da malha.

Posteriormente, outros investigadores apresentaram numerosos trabalhos baseados no método original, que possibilitam a sua aplicação a uma gama cada vez mais generalizada de modelos típicos. Paralelamente, têm-se desenvolvido estratégias alternativas de forma a ultrapassar os problemas encontrados, através da utilização de métodos alternativos para a avaliação da aproximação elementar (Ex: Funções Quadráticas, Cúbicas de *Hermite*) e para eliminação das singularidades (Ex: supressão nodal, manipulação matricial).

De facto, não se enquadra no âmbito deste trabalho, uma análise aprofundada da evolução do procedimentos de mobilidade de elementos finitos. No entanto, afigura-se óbvio que este tipo de estratégias se caracterizam por uma complexidade formal consideravelmente superior à dos métodos desenvolvidos e testados neste trabalho.

Apêndice C - Métodos de Dedução de Fórmulas para a Estimativa de Derivadas Espaciais

C.1 – Introdução

Os métodos de integração de sistemas de parâmetros distribuídos, baseados no **Método das Linhas**, tornam necessária a utilização de um procedimento de estimativa numérica do valor de uma ou várias derivadas espaciais (para modelos uni- ou multi-dimensionais, respectivamente). Essa estimativa é geralmente efectuada através de aproximações polinomiais. Deste modo, considera-se, para uma dimensão (sendo extensível a P.D.E.´s multi-dimensionais), a variação da variável dependente u(x) em relação à variável independente x:

$$u(x) = a_0 + a_1 \cdot (x - x_i) + a_2 \cdot (x - x_i)^2 + a_3 \cdot (x - x_i)^3 + \quad \cdots \tag{C.1}$$

onde: x_i - valor de x a especificar;
a_0, a_1, a_2, a_3, … - constantes a calcular.

De forma a determinar o valor da primeira constante faz-se $x = x_i$, obtendo-se imediatamente: $a_0 = u(x_i)$. Em seguida, diferencia-se a equação (C.1) em relação a x, obtendo-se a expressão:

$$\frac{d\,u(x)}{d\,x} = a_1 + 2 \cdot a_2 \cdot (x - x_i) + 3 \cdot a_3 \cdot (x - x_i)^2 + \quad \cdots \tag{C.2}$$

a partir da qual, para faz-se $x = x_i$, se obtém $a_1 = \frac{d\,u(x_i)}{d\,x}$. Deste modo, diferenciações sucessivas de (C.1), seguidas da atribuição $x = x_i$, conduzem à relação geral:

$$a_n = \left(1 / n! \right) \frac{d^n\,u(x_i)}{d\,x^n} \tag{C.3}$$

A substituição das expressões (C.3) em (C.1) para $n \to \infty$, conduz à equação,

$$u(x) = u(x_i) + \frac{d\,u(x_i)}{d\,x} \cdot (x - x_i) + 1/2 \cdot \frac{d^2\,u(x_i)}{d\,x^2} \cdot (x - x_i)^2 + 1/6 \cdot \frac{d^3\,u(x_i)}{d\,x^3} \cdot (x - x_i)^3 + \quad \cdots \tag{C.4}$$

correspondente à bem conhecida Série de *Taylor*, que consiste na base matemática para muitas das aproximações em análise numérica. Neste caso, será utilizada na avaliação de expressões algébricas para estimativa de derivadas espaciais de ordem variável em malhas de espaçamento arbitrário.

A introdução destas expressões numéricas de aproximação no sistema de P.D.E./A.´s original, transforma-o num sistema de O.D.E./A.´s que é resolvido no tempo, através de um integrador computacional (que neste caso, se trata do integrador implícito DASSL).

C.2 – Aplicação das Séries de *Taylor* para a Estimativa de Derivadas

C.2.1 – Malhas Uniformes[103] – Exemplo da dedução das fórmulas para o cálculo de primeiras derivadas através de diferenças finitas centradas de quarta ordem.

A partir das expansões em série de *Taylor*, é possível proceder à dedução de fórmulas algébricas para a aproximação de derivadas em pontos discretos. Como forma de ilustrar essa possibilidade, descreve-se de seguida, o desenvolvimento de uma expressão para a estimativa da derivada de primeira ordem $\frac{d\,u(x_i)}{d\,x} = u_x(x_i)$, utilizando os valores discretos da solução nos pontos x_{i-2}, x_{i-1}, x_{i+1} e x_{i+2} de uma malha uniforme. Assim, as aproximações de *Taylor* em cada um desses pontos têm a forma:

$$u(x_{i-2}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(x_{i-2}-x_i) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(x_{i-2}-x_i)^2 + \quad \cdots \qquad \text{(C.5)}$$

$$u(x_{i-1}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(x_{i-1}-x_i) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(x_{i-1}-x_i)^2 + \quad \cdots \qquad \text{(C.6)}$$

$$u(x_{i+1}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(x_{i+1}-x_i) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(x_{i+1}-x_i)^2 + \quad \cdots \qquad \text{(C.7)}$$

$$u(x_{i+2}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(x_{i+2}-x_i) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(x_{i+2}-x_i)^2 + \quad \cdots \qquad \text{(C.8)}$$

Considerando um espaçamento constante entre nodos consecutivos $\Delta x = x_i - x_{i-1}$, as expressões anteriores podem ser escritas como:

$$u(x_{i-2}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(-2\cdot\Delta x) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(-2\cdot\Delta x)^2 + \quad \cdots \qquad \text{(C.9)}$$

$$u(x_{i-1}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot(-\Delta x) + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(-\Delta x)^2 + \quad \cdots \qquad \text{(C.10)}$$

$$u(x_{i+1}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot\Delta x + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot\Delta x^2 + \quad \cdots \qquad \text{(C.11)}$$

$$u(x_{i+2}) = u(x_i) + \frac{d\,u(x_i)}{d\,x}\cdot 2\cdot\Delta x + \frac{1}{2}\cdot\frac{d^2\,u(x_i)}{d\,x^2}\cdot(2\cdot\Delta x)^2 + \quad \cdots \qquad \text{(C.12)}$$

Pretende-se, então, deduzir uma expressão para $u_x(x_i)$, através do estabelecimento de uma combinação linear entre as equações anteriores que permita a anulação da derivada de maior ordem possível. Neste caso, devido à utilização de quatro pontos, torna-se possível eliminar todos os termos correspondentes às derivadas até à quarta ordem. Assim, multiplica-

se cada uma das expressões por uma constante (**a**, **b**, **c** ou **d**), cujos valores são calculados de forma a satisfazer os requisitos pretendidos. De seguida, adicionam-se todas as expressões entre si de modo a se obter uma única equação. Portanto, para manter o termo da primeira derivada impõe-se a condição:

$$-2\cdot a - b + c + 2\cdot d = 1 \tag{C.13}$$

Por outro lado, para eliminar os termos referentes às derivadas de segunda, terceira e quarta ordem, é necessário que se verifiquem as relações seguintes:

$$4\cdot a + b + c + 4\cdot d = 0 \tag{C.14}$$
$$-8\cdot a - b + c + 8\cdot d = 0 \tag{C.15}$$
$$16\cdot a + b + c + 16\cdot d = 0 \tag{C.16}$$

A resolução deste sistema de equações lineares conduz à respectiva solução:

$$a = 2/4!; \quad b = -16/4!; \quad c = 16/4!; \quad d = -2/4!$$

Substituindo estes valores na expressão soma referida anteriormente, obtém-se uma equação geral centrada de aproximação de 4ª. ordem para a derivada de interesse u_x no ponto discreto x_i:

$$\frac{d\,u(x_i)}{d\,x} = \frac{1}{4!\cdot\Delta x}\cdot\left(2\cdot u(x_{i-2}) - 16\cdot u(x_{i-1}) + 0\cdot u(x_i) + 16\cdot u(x_{i+1}) - 2\cdot u(x_{i+2})\right) + O\left(\Delta x^4\right) \tag{C.17}$$

O coeficiente correspondente a $u(x_i)$ equivale ao simétrico da soma de todos os outros coeficientes. No entanto, esta fórmula é impossível de utilizar para os pontos das extremidades do domínio, sem a criação de pontos fictícios. Assim, para se obter a aproximação de $u_x(x_1)$, utilizam-se as expansões de *Taylor* da solução nos pontos x_2, x_3, x_4 e x_5:

$$a\cdot u(x_2) = a\cdot u(x_1) + a\cdot\frac{d\,u(x_1)}{d\,x}\cdot\Delta x + a\cdot 1/2\cdot\frac{d^2\,u(x_1)}{d\,x^2}\cdot\Delta x^2 + \cdots \tag{C.18}$$

$$b\cdot u(x_3) = b\cdot u(x_1) + b\cdot\frac{d\,u(x_1)}{d\,x}\cdot 2\cdot\Delta x + b\cdot 1/2\cdot\frac{d^2\,u(x_1)}{d\,x^2}\cdot(2\cdot\Delta x)^2 + \cdots \tag{C.19}$$

$$c\cdot u(x_4) = c\cdot u(x_1) + c\cdot\frac{d\,u(x_1)}{d\,x}\cdot 3\cdot\Delta x + c\cdot 1/2\cdot\frac{d^2\,u(x_1)}{d\,x^2}\cdot(3\cdot\Delta x)^2 + \cdots \tag{C.20}$$

$$d\cdot u(x_5) = d\cdot u(x_1) + d\cdot\frac{d\,u(x_1)}{d\,x}\cdot 4\cdot\Delta x + d\cdot 1/2\cdot\frac{d^2\,u(x_1)}{d\,x^2}\cdot(4\cdot\Delta x)^2 + \cdots \tag{C.21}$$

Como anteriormente, é necessário impor a condição (C.22) de forma a reter a primeira derivada:

$$a + 2\cdot b + 3\cdot c + 4\cdot d = 1 \tag{C.22}$$

Para anular as segunda, terceira e quarta derivadas, as constantes têm de verificar as restrições seguintes:

$$a + 4 \cdot b + 9 \cdot c + 16 \cdot d = 0 \quad \text{(C.23)}$$
$$a + 8 \cdot b + 27 \cdot c + 64 \cdot d = 0 \quad \text{(C.24)}$$
$$a + 16 \cdot b + 81 \cdot c + 256 \cdot d = 0 \quad \text{(C.25)}$$

A solução deste sistema é: $a = {}^{96}\!/_{4!}$; $b = -{}^{72}\!/_{4!}$; $c = {}^{32}\!/_{4!}$; $d = -{}^{6}\!/_{4!}$ obtendo-se a expressão algébrica para $u_x(x_1)$,

$$\frac{d\,u(x_1)}{d\,x} = \frac{1}{4! \cdot \Delta x} \cdot \left(-50 \cdot u(x_1) + 96 \cdot u(x_2) - 72 \cdot u(x_3) + 32 \cdot u(x_4) - 6 \cdot u(x_5)\right) + O\left(\Delta x^4\right) \quad \text{(C.26)}$$

Para a obtenção da aproximação para $u_x(x_2)$, procede-se de modo similar, utilizando-se as quatro séries de *Taylor* referentes a $u(x_1)$, $u(x_3)$, $u(x_4)$ e $u(x_5)$. O sistema de equações algébricas correspondente é:

$$-a + b + 2 \cdot c + 3 \cdot d = 1 \quad \text{(C.27)}$$
$$a + b + 4 \cdot c + 9 \cdot d = 0 \quad \text{(C.28)}$$
$$-a + b + 8 \cdot c + 27 \cdot d = 0 \quad \text{(C.29)}$$
$$a + b + 16 \cdot c + 81 \cdot d = 0 \quad \text{(C.30)}$$

A resolução deste sistema permite a dedução da fórmula de diferenciação:

$$\frac{d\,u(x_2)}{d\,x} = \frac{1}{4! \cdot \Delta x} \cdot \left(-6 \cdot u(x_1) - 20 \cdot u(x_2) + 36 \cdot u(x_3) - 12 \cdot u(x_4) + 2 \cdot u(x_5)\right) + O\left(\Delta x^4\right) \quad \text{(C.31)}$$

Procedendo-se a operações análogas para as aproximações de $u_x(x_{N-1})$ e $u_x(x_N)$, chega-se às respectivas fórmulas:

$$\frac{d\,u(x_{N-1})}{d\,x} = \frac{1}{4! \cdot \Delta x} \cdot \left(-2 \cdot u(x_{N-4}) + 12 \cdot u(x_{N-3}) - 36 \cdot u(x_{N-2}) + 20 \cdot u(x_{N-1}) + 6 \cdot u(x_N)\right) + O\left(\Delta x^4\right) \quad \text{(C.32)}$$

$$\frac{d\,u(x_N)}{d\,x} = \frac{1}{4! \cdot \Delta x} \cdot \left(6 \cdot u(x_{N-4}) - 32 \cdot u(x_{N-3}) + 72 \cdot u(x_{N-2}) - 96 \cdot u(x_{N-1}) + 50 \cdot u(x_N)\right) + O\left(\Delta x^4\right) \quad \text{(C.33)}$$

Finalmente, sumariza-se todas as expressões apresentadas anteriormente numa matriz de diferenciação, válida para todos os pontos do domínio,

$$\frac{d\,\bar{u}}{d\,x} = \frac{1}{4!\cdot\Delta x}\cdot\begin{bmatrix} -50 & 96 & -72 & 32 & -6 \\ -6 & -20 & 36 & -12 & 2 \\ 2 & -16 & 0 & 16 & -2 \\ -2 & 12 & -36 & 20 & 6 \\ 6 & -32 & 72 & -96 & 50 \end{bmatrix}\cdot\bar{u} + O\left(\Delta x^4\right) \qquad \text{(C.34)}$$

A equação (C.34) corresponde à fórmula de aproximação por diferenças finitas centradas de quarta ordem para primeiras derivadas, sendo aplicável em toda a extensão do domínio espacial.

A aplicação de um procedimento semelhante possibilita o cálculo de fórmulas correspondentes a derivadas de diferentes ordens e, através da utilização de variados números de pontos e de diversas disposições destes. Os tipos de diferenças finitas são distinguidos pelas diferentes disposições dos nodos considerados da forma:

☞ ***Diferenças Centradas*** ⇨ O conjunto de pontos considerado contém igual número de pontos em ambas as direcções a partir do nodo de interesse.

☞ ***Diferenças Descentradas*** ⇨ O conjunto de pontos considerado situa-se em apenas uma das direcções a partir do nodo de interesse:
Backward ou ***Upwind*** – Direcção negativa das abcissas x.
Forward ou ***Downwind*** – Direcção positiva das abcissas x.

☞ ***Diferenças Desc. Biased*** ⇨ O conjunto de pontos considerado apresenta uma direcção dominante a partir do nodo de interesse, na qual se utilizam mais pontos:
Backward ou ***Upwind*** – Direcção negativa dominante.
Forward ou ***Downwind*** – Direcção positiva dominante.

Deste modo, uma discretização do tipo descentrado *upwind* com cinco pontos utilizará os quatro pontos imediatamente à esquerda do nodo de interesse, enquanto que uma fórmula do tipo *biased downwind* com cinco pontos aplicará o ponto adjacente da esquerda e os três pontos vizinhos à direita do nodo de interesse (se se convencionar que a direcção positiva das abcissas x é da esquerda para a direita, como é óbvio).

Dependendo do número de pontos usado e do tipo de diferenças finitas aplicado, é também necessário calcular separadamente as fórmulas, para os pontos situados na vizinhança de uma ou de ambas as extremidades do domínio espacial, de uma forma análoga à apresentada anteriormente.

C.2.2 – Generalização para Malhas Não Uniformes – Diferenças finitas centradas de quarta ordem para primeiras derivadas.

O método anterior aplica-se de uma forma simples em malhas uniformes porque, neste caso, é possível pôr em evidência a variável Δx (passo espacial) em cada expansão, já que este se mantém constante ao longo da malha. No entanto, no caso de malhas não uniformes, nas quais o espaçamento entre os pontos não é necessariamente constante na extensão de toda

a malha, a manipulação das expansões de *Taylor* torna-se necessariamente mais complexa. Deste modo, estas expressões tomam, para um ponto genérico i situado no interior da malha, a seguinte forma (considerando a dedução equivalente à fórmula de diferenças centradas de quarta ordem anterior):

$$a \cdot u(x_{i-2}) = a \cdot u(x_i) + a \cdot \frac{d\,u(x_i)}{d\,x} \cdot (-\Delta x_{i-1} - \Delta x_i) + a \cdot \frac{1}{2} \cdot \frac{d^2\,u(x_i)}{d\,x^2} \cdot (-\Delta x_{i-1} - \Delta x_i)^2 + \quad \cdots \tag{C.35}$$

$$b \cdot u(x_{i-1}) = b \cdot u(x_i) + b \cdot \frac{d\,u(x_i)}{d\,x} \cdot (-\Delta x_i) + b \cdot \frac{1}{2} \cdot \frac{d^2\,u(x_i)}{d\,x^2} \cdot (-\Delta x_i)^2 + \quad \cdots \tag{C.36}$$

$$c \cdot u(x_{i+1}) = c \cdot u(x_i) + c \cdot \frac{d\,u(x_i)}{d\,x} \cdot \Delta x_{i+1} + c \cdot \frac{1}{2} \cdot \frac{d^2\,u(x_i)}{d\,x^2} \cdot \Delta x_{i+1}^{\ 2} + \quad \cdots \tag{C.37}$$

$$d \cdot u(x_{i+2}) = d \cdot u(x_i) + d \cdot \frac{d\,u(x_i)}{d\,x} \cdot (\Delta x_{i+1} + \Delta x_{i+2}) + d \cdot \frac{1}{2} \cdot \frac{d^2\,u(x_i)}{d\,x^2} \cdot (\Delta x_{i+1} + \Delta x_{i+2})^2 + \quad \cdots \tag{C.38}$$

em que: $\Delta x_i = x_i - x_{i-1}$, $i = 2$, $\cdots$, N

Procedendo como anteriormente, verifica-se que para se manter a primeira e anular as segunda, terceira e quarta derivadas, as constantes **a**, **b**, **c** e **d** têm de obrigatoriamente verificar as condições seguintes:

$$a \cdot A + b \cdot B + c \cdot C + d \cdot D = 1 \tag{C.39}$$
$$a \cdot A^2 + b \cdot B^2 + c \cdot C^2 + d \cdot D^2 = 0 \tag{C.40}$$
$$a \cdot A^3 + b \cdot B^3 + c \cdot C^3 + d \cdot D^3 = 0 \tag{C.41}$$
$$a \cdot A^4 + b \cdot B^4 + c \cdot C^4 + d \cdot D^4 = 0 \tag{C.42}$$

em que:

$A[\Rightarrow x_{i-2}] = -(\Delta x_{i-1} + \Delta x_i)$; $B[\Rightarrow x_{i-1}] = -\Delta x_i$;

$C[\Rightarrow x_{i+1}] = \Delta x_{i+1}$; $D[\Rightarrow x_{i+2}] = (\Delta x_{i+1} + \Delta x_{i+2})$

Cada uma das constantes A, B, C e D está relacionada com um dos pontos da discretização (neste caso, x_{i-2}, x_{i-1}, x_{i+1} e x_{i+2}, respectivamente).

Seria possível resolver o sistema anterior e deduzir as funções que relacionam as constantes [a, b, c, d] com [A, B, C, D] e, consequentemente com [Δx_{i-1}, Δx_i, Δx_{i+1}, Δx_{i+2}]. No entanto, essas relações são demasiado complexas para que a sua utilização possa ser viável. Assim, resta seguir uma alternativa melhor aplicável, sem que seja necessário recorrer à dedução e aplicação de quatro funções bastante complicadas. Deste modo, opta-se por executar a resolução do sistema apresentado, em cada avaliação da estimativa da derivada num ponto discreto da malha (na qual todas as posições nodais são perfeitamente conhecidas à partida), considerando os espaçamentos entre os pontos adjacentes. Portanto, para cada ponto onde se pretenda estimar a correspondente derivada, substitui-se no sistema os valores dos espaçamentos entre os pontos vizinhos [Δx_{i-1}, Δx_i, Δx_{i+1}, Δx_{i+2}]. De seguida, calculam-se os valores do vector de constantes [a, b, c, d] para esse ponto, por resolução do sistema de equações algébricas assim obtido. Agora, está-se em condições de calcular o valor da estimativa da derivada de primeira ordem através do somatório das expressões (C.35) a (C.38), obtendo-se a respectiva fórmula geral:

$$\frac{d\,u(x_i)}{d\,x} = a\cdot u(x_{i-2}) + b\cdot u(x_{i-1}) - (a+b+c+d)\cdot u(x_i) + c\cdot u(x_{i+1}) + d\cdot u(x_{i+2}) \qquad (C.43)$$

Por outro lado, este estratégia obriga à resolução de um sistema de equações lineares 4×4, para cada avaliação da derivada, o que constitui a sua maior desvantagem, já que obriga a um número considerável de operações algébricas.

Como anteriormente, o procedimento para os pontos junto às fronteiras é ligeiramente diferente, já que nesses pontos é impossível utilizar uma disposição dos nodos equivalente ao caso geral, sem se recorrer à introdução de pontos fictícios adicionais e exteriores à malha global. No entanto, neste caso, não é necessário deduzir uma nova expressão para o sistema a resolver em cada ponto, para a avaliação dos pesos. A forma geral deste sistema é utilizável em todos os pontos do domínio. O que difere é a definição das constantes [A, B, C, D]. Assim, em cada ponto, estas são definidas por:

x_1: $A[\Rightarrow x_2] = \Delta x_2$; $B[\Rightarrow x_3] = (\Delta x_2 + \Delta x_3)$;
$C[\Rightarrow x_4] = (\Delta x_2 + \Delta x_3 + \Delta x_4)$; $D[\Rightarrow x_5] = (\Delta x_2 + \Delta x_3 + \Delta x_4 + \Delta x_5)$

x_2: $A[\Rightarrow x_1] = -\Delta x_2$; $B[\Rightarrow x_3] = \Delta x_3$;
$C[\Rightarrow x_4] = (\Delta x_3 + \Delta x_4)$; $D[\Rightarrow x_5] = (\Delta x_3 + \Delta x_4 + \Delta x_5)$

x_{N-1}: $A[\Rightarrow x_{N-4}] = -(\Delta x_{N-3} + \Delta x_{N-2} + \Delta x_{N-1})$; $B[\Rightarrow x_{N-3}] = -(\Delta x_{N-2} + \Delta x_{N-1})$;
$C[\Rightarrow x_{N-2}] = -\Delta x_{N-1}$; $D[\Rightarrow x_N] = \Delta x_N$

x_N: $A[\Rightarrow x_{N-4}] = -(\Delta x_{N-3} + \Delta x_{N-2} + \Delta x_{N-1} + \Delta x_N)$; $B[\Rightarrow x_{N-3}] = -(\Delta x_{N-2} + \Delta x_{N-1} + \Delta x_N)$;
$C[\Rightarrow x_{N-2}] = -(\Delta x_{N-1} + \Delta x_N)$; $D[\Rightarrow x_{N-1}] = -\Delta x_N$

Usando as constantes definidas desta forma, procede-se à estimativa da primeira derivada pela aproximação de quarta ordem, nesses pontos pelo procedimento descrito anteriormente, obtendo-se em cada caso, expressões semelhantes a (C.43).

É, igualmente possível a utilização deste método para avaliar derivadas de ordens diferentes, com a aplicação de diferenças finitas de tipos variados, em malhas não uniformes.

C.2.3 – Caso Particular – Estimativa da segunda derivada, em pontos fronteira descritos por condições de Neumann, através de fórmulas de quarta ordem.

Nesta secção procede-se à descrição da estratégia (baseada no método anterior), adaptada para a estimativa da segunda derivada nos pontos fronteira (x_1 e/ou x_N), através da utilização dos valores da solução nos pontos adjacentes e da primeira derivada no próprio ponto fronteira. Este procedimento é necessário no caso do modelo ser sujeito a condições de *Neumann* num ou em ambos os pontos fronteira. Neste caso, utiliza-se um conjunto de cinco pontos, obtendo-se, deste modo, aproximações de quarta ordem.

Para x_1, parte-se das expressões de *Taylor* referentes aos pontos vizinhos x_2, x_3, x_4 e x_5. Para estimar a segunda derivada, multiplica-se cada uma dessas equações pelas constantes **a**, **b**, **c** e **d**, que serão associadas a cada uma dos pontos referidos acima. Agora, pretende-se

encontrar uma combinação entre cada série de forma a que se retenha o termo referente às segundas derivadas e se anulem os termos correspondentes às primeira, terceira, quarta e quinta derivadas. Neste exemplo, é necessário recorrer aos termos de quinta ordem, já que é introduzida uma nova constante **e**, relacionada com a primeira derivada em x_1. Assim, obtém-se o sistema de equações lineares seguinte:

$$a \cdot A + b \cdot B + c \cdot C + d \cdot D + e = 0 \qquad \text{(C.44)}$$
$$a \cdot A^2 + b \cdot B^2 + c \cdot C^2 + d \cdot D^2 = 2 \qquad \text{(C.45)}$$
$$a \cdot A^3 + b \cdot B^3 + c \cdot C^3 + d \cdot D^3 = 0 \qquad \text{(C.46)}$$
$$a \cdot A^4 + b \cdot B^4 + c \cdot C^4 + d \cdot D^4 = 0 \qquad \text{(C.47)}$$
$$a \cdot A^5 + b \cdot B^5 + c \cdot C^5 + d \cdot D^5 = 0 \qquad \text{(C.48)}$$

onde A, B, C e D são definidos de forma equivalente à apresentada na secção anterior para o ponto fronteira x_1.

Desse modo, dependendo das posições relativas entre os pontos, calculam-se os valores de cada constante, através da resolução do sistema. É fácil verificar então, que a estimativa da segunda derivada em x_1 pode ser calculada por:

$$\frac{d^2\, u(x_1)}{d\, x^2} = -(a+b+c+d)\cdot u(x_1) + a \cdot u(x_2) + b \cdot u(x_3) + c \cdot u(x_4) + d \cdot u(x_5) + e \cdot \frac{d\, u(x_1)}{d\, x} \qquad \text{(C.49)}$$

Através de um procedimento semelhante, pode-se avaliar a segunda derivada no outro ponto fronteira x_N, utilizando, igualmente, a solução discreta nos pontos adjacentes e o valor da primeira derivada em x_N.

C.3 – Geração de Fórmulas de Diferenças Finitas em Grelhas Arbitrariamente Espaçadas

Na presente secção, procede-se à descrição de um método mais simples e expedito para a geração de fórmulas de diferenças finitas, desenvolvido por *Fornberg*[45,46]. Este método utiliza um esquema recursivo que permite o cálculo dos pesos para qualquer ordem de derivada (incluindo a ordem zero, correspondente à interpolação), aproximados a qualquer grau de precisão, numa grelha arbitrária unidimensional. Cada avaliação apenas necessita de quatro operações aritméticas, sendo bastante adequada a sua aplicação a grelhas dinâmicas e, consequentemente, não uniformes.

Portanto, sendo $M \geq 0$, a ordem mais elevada da derivada que se pretende aproximar, a partir de um conjunto de N+1 pontos da grelha (de coordenadas x: α_0, …, α_N; $N \geq 0$), o problema consiste em calcular os pesos, de modo que as aproximações,

$$\left.\frac{\delta^m}{\delta\, x^m}\right|_{x=x_0} \approx \sum_{\nu=0}^{n} \delta_{n,\nu}^m \cdot f(\alpha_\nu); \qquad m = 0,1,\cdots,M; \qquad n = m, m+1, \cdots, N \qquad \text{(C.50)}$$

sejam caracterizadas pelo grau de precisão óptimo (geralmente n-m+1, embora possa ser mais elevado para casos particulares).

Derivação do Algoritmo:

Por uma questão de simplicidade, considera-se a aproximação das derivadas no ponto $x_0 = 0$. Então, sendo $\{\alpha_0, \alpha_1, \ldots, \alpha_N\}$ números reais e distintos,

$$\omega_n(x) = \prod_{k=0}^{n}(x - \alpha_k) \tag{C.51}$$

e o polinómio:

$$F_{n,\nu}(x) = \frac{\omega_n(x)}{\omega'_n(\alpha_\nu)\cdot(x - \alpha_\nu)} \tag{C.52}$$

é o de menor grau que assume o valor unitário para $x = \alpha_\nu$, anulando-se em $x = \alpha_k$, $0 \le k \le n$, $k \neq 0$. Para uma função arbitrária f(x) e os nodos $x = \alpha_\nu$, a interpolação polinomial de *Lagrange* toma a forma:

$$p(x) = \sum_{\nu=0}^{n} F_{n,\nu}(x)\cdot f(\alpha_\nu) \tag{C.53}$$

Os pesos desejados exprimem a forma como os valores de $\left[\delta^m\, p(x) \Big/ \delta\, x^m\right]_{x=0}$ variam com as alterações em $f(\alpha_\nu)$. Como só um dos termos de p(x) depende da variação em cada $f(\alpha_\nu)$, verifica-se então que:

$$\delta^m_{n,\nu} = \left[\frac{d^m}{d\, x^m} F_{n,\nu}(x)\right]_{x=0} \tag{C.54}$$

Assim, o polinómio de grau n $F_{n,\nu}$, pode ser escrito da forma:

$$F_{n,\nu}(x) = \sum_{m=0}^{n} \frac{\delta^m_{n,\nu}}{m!}\cdot x^m \tag{C.55}$$

A partir de (C.52), e sabendo que: $\begin{cases}\omega(x) = (x - \alpha_n)\cdot\omega_{n-1}(x)\\ \omega'_n(x) = (x - \alpha_n)\cdot\omega'_{n-1}(x) + \omega_{n-1}(x)\end{cases}$, obtém-se,

$$F_{n,\nu}(x) = \frac{x - \alpha_n}{\alpha_\nu - \alpha_n}\cdot F_{n-1,\nu}(x) \tag{C.56}$$

e

$$F_{n,n}(x) = \frac{\omega_{n-1}(x)}{\omega_{n-1}(\alpha_n)} = \frac{\omega_{n-2}(\alpha_{n-1})}{\omega_{n-1}(\alpha_n)}\cdot(x - \alpha_{n-1})\cdot F_{n-1,n-1}(x) \qquad (n > 1) \tag{C.57}$$

Substituindo a expressão (C.55) em (C.56) e (C.57), e equacionando as potências de x, deduz-se as relações de recursão entre os pesos:

$$\delta_{n,0}^{m} = \frac{1}{\alpha_n - \alpha_\nu} \cdot \left(\alpha_n \cdot \delta_{n-1,0}^{m} - m \cdot \delta_{n-1,0}^{m-1} \right) \tag{C.58}$$

e

$$\delta_{n,n}^{m} = \frac{\omega_{n-2}(\alpha_{n-1})}{\omega_{n-1}(\alpha_n)} \cdot \left(m \cdot \delta_{n-1,n-1}^{m-1} - \alpha_{n-1} \cdot \delta_{n-1,n-1}^{m} \right) \tag{C.59}$$

A aplicação destas expressões conduz ao desenvolvimento de um algoritmo relativamente simples para o cálculo dos pesos $\delta_{n,\nu}^{m}$, correspondentes ao cálculo de aproximações a derivadas de ordem m, num ponto x_0, e utilizando n+1 pontos de abcissas $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$ pertencentes a uma grelha arbitrariamente espaçada.

Apêndice D - Descrição da Estrutura dos Códigos

Para cada um dos algoritmos de integração descritos anteriormente (**Refinamento** e **Mobilidade Nodal Dinâmica**) foi elaborado um código geral em linguagem FORTRAN 77 aplicável a uma vasta variedade de modelos. Os programas computacionais são pensados de forma a possibilitarem uma utilização por parte de utilizadores não necessariamente familiarizados com a estrutura específica de cada algoritmo. Desse modo, as informações fornecidas pelo utilizador são, na medida do possível, apenas as indispensáveis de forma a caracterizar o modelo a executar e as condições em que se pretende realizar a aplicação do método. Assim, os dados a fornecer a cada código, são de dois tipos:

❶ Informações gerais sobre as características do problema a resolver, sob a forma de várias subrotinas auxiliares.

❷ Parâmetros essenciais da execução do algoritmo de integração, introduzidos a partir de um ficheiro de dados.

Os dois códigos foram concebidos de forma a utilizarem exactamente as mesmas subrotinas para a descrição do modelo (informação de tipo ❶).

Neste momento, a aplicação de cada código é restringida a problemas unidimensionais diferenciais ou algébrico-diferenciais, explícitos em relação às derivadas temporais e com derivadas espaciais de ordem não superior a dois. Adicionalmente, o código referente ao método adaptativo de movimentação nodal dinâmica, não é aplicável a problemas que envolvam equações algébricas. No entanto, qualquer das limitações acima referidas é facilmente ultrapassável, já que a estrutura de cada código permite a introdução destas generalizações sem grande dificuldade.

D.1 – Algoritmo de Refinamento - Programa REFIN

D.1.1 – Estrutura Geral.

O código REFIN é desenvolvido para a aplicação do algoritmo de refinamento a problemas algébrico-diferenciais que podem ser generalizados da seguinte forma:

$$u^i{}_t = f_i\left(u, u_z, u_{zz}\right), \qquad i = 1, \cdots, NPDE \tag{D.1}$$

e

$$0 = g_j\left(u\right), \qquad j = NPDE + 1, \cdots, N \tag{D.2}$$

Condições fronteira:

$$u\left(z^L, t\right) = u^L \tag{D.3}$$

$$u\left(z^R, t\right) = u^R \tag{D.4}$$

Condição inicial:

$$u\left(z,0\right) = u^0\left(z\right) \quad ; \quad z \in \left[z^L, z^R\right] \tag{D.5}$$

em que, $$u(z,t) = \left[u^1(z,t),\ u^2(z,t),\ \cdots\ ,u^N(z,t)\right]$$

Desse modo foi elaborado um programa em FORTRAN 77, cuja estrutura geral é esquematizada na Figura D.1. Cada uma das caixas pequenas corresponde a uma subrotina ou função, enquanto que as duas caixas maiores dizem respeito ao programa principal do código (REFIN) e ao conjunto de subrotinas que constituem o integrador DDASSL (o primeiro D da sigla significa que o programa utilizado corresponde à versão de dupla precisão). As setas do esquema representam as interligações entre as subrotinas do código. Assim, uma seta significa que a subrotina ou o programa donde parte chama a subrotina ou a função para onde se dirige.

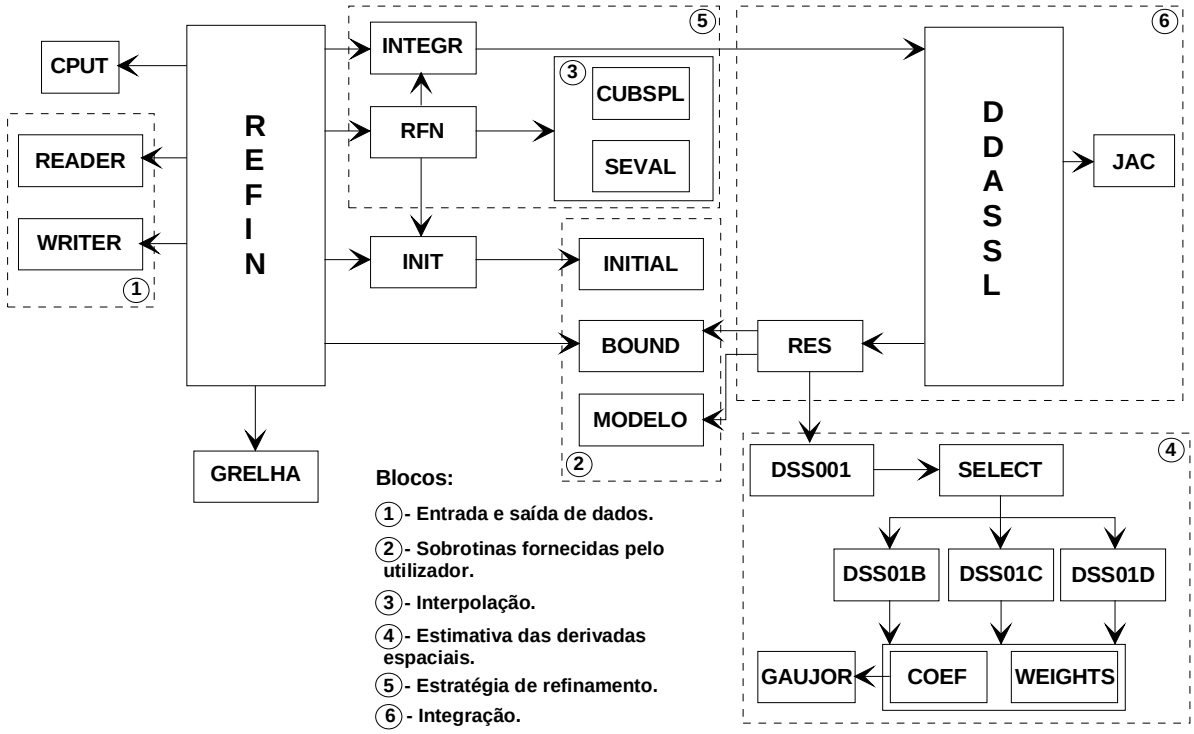

Figura D.1: Esquema simplificado da estrutura do código de refinamento

No programa principal REFIN são efectuadas algumas operações gerais que podem ser descritas resumidamente da seguinte forma:

❶ Determinação do tempo de computação, através da chamada da subrotina CPUT no início e no final da execução do programa. Considera-se o tempo de cpu como a diferença entre os dois tempos obtidos nesses dois instantes.

❷ Introdução dos dados e escrita dos resultados, efectuados pelas chamadas das subrotinas READER e WRITER, respectivamente.

❸ Avaliação resumida do tipo de condições fronteira do problema, pela chamada da subrotina BOUND, fornecida pelo utilizador (com **LFLAG** = 1).

❹ Inicialização das integrações com malhas fixas, executadas com a chamada da subrotina INTEGR, que efectua todas as atribuições de variáveis, directamente relacionadas com o integrador DDASSL. No caso de se tratar do primeiro passo temporal, a correspondente inicialização da integração é efectuada pela chamada da subrotina INIT. Esta rotina procede a avaliação dos perfis iniciais, quer pela utilização das respectivas expressões analíticas (através da chamada da subrotina INITIAL), quer pela leitura directa das soluções discretas, nos ficheiros para tal definidos.

❺ Cálculo e comparação dos erros espaciais e consequente selecção dos nodos "insatisfatórios" em cada nível de refinamento. A aplicação da fase de integração dos subproblemas de diversos níveis, assim gerados, é executada com a chamada da subrotina RFN.

❻ Construção dos perfis de solução globais, por junção dos resultados de diferentes níveis obtidos por integração de cada subproblema.

❼ Ajuste do passo temporal, caso ocorram dificuldades de desempenho do integrador ou o nível de refinamento máximo seja ultrapassado. Assim,

☞ No primeiro caso, o algoritmo reduz o passo temporal base em factores de dois, até obter um passo que termine imediatamente antes do instante de tempo onde o integrador assinale problemas de avanço. Em seguida, o algoritmo executa outro passo de tamanho semelhante e, posteriormente tenta avançar o mais possível até se aproximar do tempo final original. Quando esse instante é atingido, inicia-se a integração seguinte com o passo base inicial.

☞ No caso do nível máximo ser ultrapassado, o algoritmo apenas reduz o passo base a metade, sendo todas as outras operações, necessárias à finalização dessa integração, semelhantes às descritas no caso anterior.

❽ Redefinição facultativa da malha base de segundo nível, efectuada através da introdução de nodos adicionais, em intervalos que necessitaram de refinamento no passo temporal imediatamente anterior, e a remoção destes, nos intervalos onde se verificou não ser necessário qualquer refinamento, para o mesmo passo. Esta operação é realizada com a chamada da subrotina GRELHA, que executa esse procedimento.

As restantes subrotinas que constituem o código são agrupadas em seis blocos (vd. Figura D.1), consoante a função atribuído a cada conjunto de programas, correspondente a uma operação essencial à execução do algoritmo. Considerando, assim, cada bloco separadamente, tem-se que:

Bloco ⓘ: Entrada e saída de dados.

O Bloco ① é constituído por duas subrotinas cuja função consiste somente na leitura dos parâmetros de entrada para a execução (subrotina READER) e na escrita dos resultados pretendidos pelo utilizador (subrotina WRITER).

O ficheiro de dados, denominado por DATA, é construído pelo utilizador e define todas as características necessárias para a execução. Os parâmetros possíveis de especificar pelo utilizador no ficheiro DATA são resumidos na Tabela D.1, juntamente com a sua finalidade na evolução da execução.

Tabela D.1: Resumo das variáveis de entrada do programa REFIN fornecidas pelo ficheiro DATA.

Nome da Variável	*Tipo de Variável*	*Valores que Assume*	*Significado*	*Observações*
NPDAE	Escalar Inteira		Nº de equações total do modelo	**NPDAE** = **NPDE** + **NPAE**
NPDE			Nº de equações diferenciais	
NPAE			Nº de equações algébricas	
ZE	Escalar Dupla Precisão		Posição da fronteira esquerda	
ZD			Posição da fronteira direita	
TINI			Instante de tempo inicial	Tempo inicial da integração
TFIN			Instante de tempo final	Tempo final da integração
ATOL			Tolerância absoluta	Tolerâncias referentes ao integrador
RTOL			Tolerância relativa	
NPA	Escalar Inteira		Nº de pontos de discretização	Referente ao tipo de diferenças finitas escolhido
TIPO(I)	Vector Caracter	CENTRAIS	Tipo de diferenças finitas	Diferenças centradas
		NCENTRAL		Diferenças descentradas; I = 1, … , **NPDAE**
		NCBIASED		Diferenças descentradas *biased*
TIPO1(I)		UP	Orientação predominante das diferenças finitas	Diferenças *Upwind* ; I = 1, … , **NPDAE**
		DOWN		Diferenças *Downwind*
NP	Escalar Inteira		Nº de pontos de interpolação	Apenas necessário se **NINTPL** = 1
NPR			Nº de parâmetros do modelo	
ITER			Nível máximo de refinamento	Menor ou igual a 10
NINTPL		0	Parâmetro que controla o tipo de interpolação	Interpolações lineares
		1		Interpolações de *Splines* cúbicas com **NP** pontos
NGRINI		0	Parâmetro que controla a forma como é introduzida a malha base	Malha de nível 1 uniforme com **NPD** pontos
		1		Malha de nível 1 introduzida como dado
NGRINT		0	Parâmetro que controla a forma de redefinição da malha base de nível 2	Nunca é redefinida
		1		Apenas nos tempos de transição **TCRIT**
		2		Apenas em cada passo base **DELTAT**
		3		Em todos os passos
NINIT		0	Parâmetro que controla o modo de leitura das condições iniciais	Através da sua forma analítica (rotina INITIAL)
		1		Na forma discreta, por um ficheiro de dados
NDERV2		0	Parâmetro que define a forma como é calculada a 2ª derivada	Por utilização directa da solução
		1		Por diferenciação da primeira derivada
NIPRINT		0	Parâmetro que define as variáveis que são consideradas na escrita dos resultados	Nada é imprimido
		1		Apenas os perfis referentes à malha de nível 2
		2		A solução na malha refinada global
		3		Todos os perfis referentes a todos os níveis
THETA(I)	Vector D. P.		Valores dos parâmetros do modelo	I = 1, … , **NPR**
NPD	Escalar Inteira		Nº de pontos da malha base inicial	Malha de nível 1
NSDT			Nº de subzonas temporais	Nº de partições do domínio temporal
NZONA			Subzona temporal de arranque	Partição em que a integração arranca
DELTAT(I)	Vector D. P.		Passos de tempo base	I = 1, … , **NSDT**
NPRT(I)	Vector Inteira		Intervalos de impressão	I = 1, … , **NSDT**
TOLER(I,J)	Matriz D. P.		Tolerâncias para o algoritmo	I = 1, … , **NPDE** ; J = 1, … , **NSDT**
TCRIT(I)	Vector D. P.		Tempos de transição entre zonas	I = 1, … , **NSDT**-1 ; Para **NSDT** > 1
Z(1,J)	Matriz D. P.		Posições na malha base inicial	J = 1, … , **NPD** ; Apenas para **NGRINI** = 1
NAME(I)	Vector Car.		Nomes dos ficheiros de leitura	I = 1, … , **NPDAE** ; Apenas para **NINIT** = 1

Os intervalos de impressão (**NPRT**) especificam a listagem dos perfis pretendidos entre intervalos temporais de dimensão **DELTAT(K)**×**NPRT(K)**, partindo do tempo inicial (**TINI**). No entanto, a partir da segunda partição do domínio temporal (K > 1; quando a opção de subdivisão deste é utilizada), este intervalo apenas continua a ser considerado se **DELTAT(K)**×**NPRT(K)** < **TCRIT(K-1)**. Caso contrário, a saída dos resultados será executada entre intervalos equivalentes, mas tendo como instante de partida t = **TCRIT(K-1)**.

Quando tal é solicitado (**NINIT** = 1), as soluções iniciais do problema são introduzidas no sistema, através da leitura de um ficheiro de dados. O nomes dos ficheiros (**NAME**), correspondentes a cada uma das variáveis do modelo em estudo, são definidos pelo utilizador, sendo completamente arbitrários, desde que não ocupem mais do que oito caracteres. Por uma questão de comodidade, convenciona-se que a estrutura dos ficheiros,

onde as condições iniciais são armazenadas de uma forma discreta, deve ser semelhante à dos ficheiros de resultados criados pelo código, onde são escritos os perfis globais da solução (ficheiros fort.2*). Desse modo, na construção de cada ficheiro **NAME**, é obrigatório definir quatro colunas de dados, das quais apenas a segunda e a terceira são verdadeiramente importantes. Assim, na segunda coluna são colocados os valores das posições da malha inicial. Este vector tem de necessariamente conter os valores para as posições da malha base de segundo nível gerada pelo código, a partir da malha inicial especificada pelos parâmetros de entrada. A terceira coluna é reservada para a definição dos perfis de solução iniciais. Obviamente, que cada valor da solução terá que corresponder à abcissa colocada na mesma linha.

Por outro lado, no que diz respeito à escrita de resultados, o utilizador pode escolher entre várias opções que definem vários níveis de pormenor, variando desde a listagem de nenhum resultado, até à escrita de praticamente todos os perfis calculados correspondentes à totalidade dos níveis de refinamento utilizados. O parâmetro de entrada que controla essa impressão designa-se por **NIPRINT**. Assim, conforme o valor de **NIPRINT**, o programa imprime os diversos resultados em ficheiros denominados por "fort.**n**", onde **n** corresponde a:

n = 50 + i	⇨	Perfil inicial (t = **TINI**) da variável **i**;
n = 200 + 10 × i	⇨	Solução na grelha base de nível 2, para a variável **i**;
n = 300 + 10 × i	⇨	Solução na grelha global constituída por todos os nodos de cada nível de refinamento, para a variável **i**;
n = 100 + 10 × i + j	⇨	Perfis de nível **j** da variável **i**;
n = 50	⇨	Tempo de computação da execução.

A definição de um valor crescente para **NIPRINT** permite a obtenção de um maior número de resultados. Assim, para **NIPRINT** = K; K = 1, … , 3; o programa lista o tipo de resultados associados a esse valor (referidos na Tabela D.1), além dos correspondentes a valores de **NIPRINT** inferiores a K. Para K = 3, obtém-se o grau máximo de informação com a impressão de todos os resultados disponíveis, nos ficheiros apropriados.

Nos ficheiros fort.2*, para além dos perfis globais da solução, são igualmente imprimidos os graus de refinamento correspondentes a cada nodo (variável interna **IP**), ou seja, o nível a que foi necessário refinar a malha base de modo a se obter a convergência do algoritmo, para aquele nodo. Assim, o perfil escrito nesses ficheiros, corresponde ao conjunto das soluções obtidas em cada ponto da malha base de segundo nível, por integração da malha de grau **IP** respectiva.

Bloco ②: Subrotinas fornecidas pelo utilizador.

As subrotinas que constituem o bloco ② são fornecidas pelo utilizador, possibilitando uma definição, o mais possível completa, das características do modelo a resolver. Desse modo, o utilizador necessita de construir três subrotinas, que são introduzidas no código global e se relacionam com:

Subrotina INITIAL	⇨	Fornece a condição inicial analítica do problema, quando esta é requerida;
Subrotina BOUND	⇨	Fornece as condições fronteira e as suas características em ambas as extremidades do domínio espacial;
Subrotina MODELO	⇨	Fornece as expressões das funções **f** e **g** do modelo diferencial ou algébrico diferencial;

As diversas variáveis a definir em cada uma dessas subrotinas estão resumidas na tabela seguinte:

Tabela D.2: Resumo das variáveis presentes nas subrotinas definidas pelo utilizador.

Subrotina	*Nome das Variáveis*	*Tipo de Variável*	*Valores que Assume*	*Significado*	*Observações*
INITIAL	**Y**	Escalar D. P.		Valor das variáveis no instante inicial	Apesar de ser uma variável escalar é necessário definir J valores para Y com J = 1, … , Nº Eq., através do recurso a comandos IF
BOUND	**LBE(I)**	Vector Inteira	0	Tipo de Fronteira Esquerda	Valores fixos ou definidos no tempo
			1		Variação não definida directamente à partida
	LBD(I)		0	Tipo de Fronteira Direita	Valores fixos ou definidos no tempo
			1		Variação não definida directamente à partida
	YZB(I,J)	Matriz D. P.		Valores na fronteira	Solução: C. *Dirchlet*; 1ª Derivada: C. *Neumann*
	BD(I,J)		0	Restrições em cada fronteira	Existem restrições
			1		Não há qualquer restrição: variação livre
	DYBDT(I,J)			Derivadas temporais	Valores das derivadas temporais nas fronteiras
	Y(I,J)			Solução nas fronteiras	Valores da solução em cada fronteira
	ORDMAX(I)	Vector D. P.		Ordem máxima	Valor da ordem máxima da derivada
MODELO	**EQ**	Escalar D. P.		Resíduos do modelo	Valores das funções **f** e **g** do modelo
	THETA(I)	Vector D. P.		Parâmetros do modelo	Valores dos parâmetros do modelo
	Y(I)			Solução	Valores da solução num ponto
	YZ(I)			Primeira derivada	Valores da primeira derivada espacial
	YZZ(I)			Segunda derivada	Valores da segunda derivada espacial

I = 1, … , Nº eq. do modelo
J = 1 ⇨ Fronteira Esquerda; J = 2 ⇨ Fronteira Direita

Para a ordenação dos componentes da solução nos respectivos vectores, é necessário ter em conta que, os primeiros elementos têm que ser associados às variáveis diferenciadas no tempo (I = 1, … , **NPDE**). As variáveis restantes são consideradas nas posições seguintes (I = **NPDE**+1, … , **NPDAE**). Assim, na construção da subrotina MODELO, atribuem-se aos primeiros **NPDE** índices J, as funções relacionadas com as equações diferenciais (funções **f**), enquanto que os restantes são associados às expressões algébricas (funções **g**).

Bloco ③: Subrotinas que implementam as operações de interpolação.

Neste bloco agrupam-se as rotinas referentes à avaliação da solução interpolada nos perfis de arranque de nível superior a dois. Depois de seleccionada a dimensão do vector de pontos de interpolação (definido pelo parâmetro **NP**), na malha de nível dois, utiliza-se a subrotina CUBSPL para o cálculo dos pesos correspondentes a cada ponto, através da utilização de *Splines* cúbicas. Em seguida, recorre-se a esses pesos e à função SEVAL, para a estimativa da solução interpolada na abcissa pretendida. No caso de se fixar **NINTPL** = 0, todas as interpolações são efectuadas pela aproximação linear da solução, entre os pontos da malha de segundo nível que englobem a posição espacial de interesse.

Bloco ④: Subrotinas para a estimativa das derivadas espaciais.

O bloco ④ engloba todas as subrotinas utilizadas na estimativa do valor das derivadas espaciais através de aproximações de diferenças finitas. Deste modo, a subrotina DSS001

consiste no programa geral que define a direcção espacial em que a diferenciação é realizada. Esta rotina está preparada para lidar com problemas de qualquer dimensão até um máximo de três coordenadas espaciais. A selecção do tipo de diferenças finitas a realizar é realizada pela subrotina SELECT, que dependendo do valor definido para o parâmetro **TIPO**, chama as seguintes subrotinas:

Subrotina DSS01B ⇨ **TIPO** = NCBIASED; Diferenças finitas descentradas *biased* de **NPA** pontos, com direcção preferencial **TIPO1**.

Subrotina DSS01C ⇨ **TIPO** = CENTRAIS; Diferenças finitas centradas de **NPA** pontos.

Subrotina DSS01D ⇨ **TIPO** = NCENTRAL; Diferenças finitas descentradas de **NPA** pontos, com direcção preferencial **TIPO1**.

Os pesos correspondentes a cada ponto do vector básico de dimensão **NPA**, são calculadas através do método recursivo de *Fornberg*[45,46], pela subrotina WEIGHTS. Apenas num caso especial se utiliza o método de resolução de sistemas de equações lineares (vd. Apêndice C), para a avaliação desses coeficientes. Esse caso corresponde à estimativa da segunda derivada em pontos fronteira sujeitos a condições de *Neumann*, onde se adiciona a contribuição do valor fixo da primeira derivada no ponto fronteira, à fórmula de estimativa geral. O cálculo dos coeficientes correspondentes é efectuado pela subrotina COEF, que utiliza a rotina GAUJOR para a resolução de cada sistema linear, pelo método de eliminação de *Gauss-Jordan*.

Bloco ⑤: Subrotinas que implementam a estratégia de refinamento.

O bloco ⑤ abrange todas as subrotinas relacionadas com a implementação do algoritmo de refinamento. Desse modo, a subrotina RFN procede à construção dos subdomínios referentes a cada nível de refinamento e à consequente inicialização de cada subproblema gerado. A integração temporal destes subproblemas é efectuada pela chamada da subrotina INTEGR que condiciona todos os parâmetros de execução do integrador implícito DDASSL. No caso de se tratar do primeiro passo de integração, a rotina RFN necessita de recorrer à subrotina INIT para a inicialização destes problemas. Caso contrário, a avaliação dos perfis de arranque da cada subproblema é obtida por interpolação do perfil de nível dois obtido no passo anterior. Estas interpolações podem ser lineares (**NINTPL** = 0) ou são calculadas através da aplicação de *Splines* cúbicas (chamada das rotinas que constituem o bloco ③).

Bloco ⑥: Subrotinas referentes à integração temporal.

O núcleo do bloco ⑥, referente à integração temporal, consiste na *package* DDASSL que engloba as subrotinas que executam o avanço do algoritmo no tempo (vd. Apêndice A). Este integrador implícito necessita da definição de duas subrotinas adicionais: a subrotina JAC, onde se processa o cálculo analítico dos componentes da matriz jacobiana de cada modelo a integrar (como neste caso, o jacobiano é estimado por diferenças finitas, esta rotina

é definida como *dummy*); a subrotina RES que, em cada instante temporal, fornece os resíduos relacionados com as equações do modelo. Assim, esta subrotina é bastante mais complexa, não se limitando apenas a calcular os resíduos (pela chamada da subrotina MODELO), como a definir as características fronteira de cada problema ou subproblema integrado (através da chamada da subrotina BOUND) e a proceder ao cálculo das estimativas para as derivadas espaciais (recorrendo à rotina DSS001).

A forma como o código é, de momento, definido, implica a verificação de algumas limitações que podem ser resumidas da forma seguinte:

Número máximo de equações do modelo (**NPDAE**)	⇨	5
Nível máximo de refinamento da malha base (**ITER**)	⇨	10
Número máximo de parâmetros do modelo (**NPR**)	⇨	10
Número máximo de subintervalos no domínio temporal (**NSDT**)	⇨	8
Número máximo de pontos seleccionáveis em cada nível (Variável Interna)	⇨	100
Número máximo de nodos para a malha base de nível 1 (**NPD**)	⇨	101
Número máximo de nodos em cada subdomínio (Variável Interna)	⇨	301
Dimensão máxima do vector de nodos de interpolação (**NP**)	⇨	10
Número máximo de subdomínios gerados em cada nível (Variável Interna)	⇨	30
Dimensão máxima do vector de nodos de discretização (**NPA**)	⇨	15

Há ainda que considerar a escolha das variáveis **NPD** e **ITER,** de maneira a que a seguinte expressão seja satisfeita:

$$(2^{(\mathrm{ITER}-1)} \times \mathrm{NPD}) - (2^{(\mathrm{ITER}-1)} - 1) \leq 10241 \qquad \text{(D.6)}$$

Por exemplo, se se pretender uma malha base de nível um com 21 nodos, o valor máximo definido para **ITER** terá de ser necessariamente 10. Por outro lado, se **NPD** = 41, o nível máximo de refinamento será 9.

Estes valores máximos podem ser alterados por manipulação das dimensões dos vectores ou matrizes adequados. No entanto, um aumento excessivo do valor desses parâmetros poderá originar problemas na execução do código, conduzindo a exigências de memória incomportáveis, além de poder afectar negativamente os tempos de computação.

D.1.2 – Apresentação de um Exemplo.

Nesta secção, procede-se à apresentação e análise dos dados a fornecer pelo utilizador para uma hipotética execução de um exemplo estudado neste trabalho. O problema seleccionado refere-se ao modelo algébrico-diferencial que simula o comportamento de uma coluna de adsorção/reacção (**Exemplo 12**). As condições da execução, apresentadas em seguida, não correspondem às utilizadas em nenhum dos *runs* estudados para este exemplo, pretendendo, apenas, ilustrar a aplicação de uma vasta gama de opções postas à disposição pelo código. Nem todas as potencialidades deste foram convenientemente testadas, no decurso do presente trabalho.

Deste modo, considera-se o seguinte ficheiro DATA, que define um conjunto possível de parâmetros de execução do algoritmo de refinamento, para o modelo em questão:

```
5,3,2                                                          NPDAE,NPDE,NPAE
0.D0,1.D0                                                      ZE,ZD
0.D0,4.0D+3                                                    TINI,TFIN
1.D-5,1.D-5                                                    ATOL,RTOL
5                                                              NPA
NCBIASED,UP                                                    TIPO(I),TIPO1(I)
,                                                              I=1, ... , NPDAE
,
,
,
5,8                                                            NP,NPR
10,1,1,0,1                                                     ITER,NINTPL,NGRINI,NGRINT,NINIT
0,3                                                            NDERV2,NIPRINT
1.D4,74.5D0,2.5D0,1.575D-4,0.2D0,0.617D0,4143D0,1.177D0        THETA(I), I=1, ... , NPR
14                                                             NPD
1,1                                                            NSDT,NZONA
2.0D+2                                                         DELTAT(I), I=1, ... , NSDT
2                                                              NPRT(I), I=1, ... , NSDT
1.D-2                                                          TOLER(I,J)
1.D-2                                                          I=1, ... , NPDE
1.D-2                                                          J=1, ... , NSDT
TCRIT(I), I=1, ... , NSDT-1
0.D0                                                           Z(1,J)
1.D-2                                                          J=1, ... , NPD
2.D-2
1.D-1
2.D-1
3.D-1
4.D-1
5.D-1
6.D-1
7.D-1
8.D-1
9.D-1
9.5D-1
1.D+0
data.210                                                       NAME(I)
data.220                                                       I=1, ... , NPDAE
data.230
data.240
data.250
```

Pela análise do ficheiro de dados anterior, verifica-se que o problema é constituído por cinco equações (**NPDAE** = 5): três diferenciais e duas algébricas (**NPDE** = 3 e **NPAE** = 2). O domínio espacial corresponde ao intervalo [0 , 1] (**ZE** = 0.0 e **ZD** = 1.0), enquanto que a integração temporal se desenrola entre os instantes t = 0 e t = 4000 (**TINI** = 0.0 e **TFIN** = 4000.0). As tolerâncias absoluta e relativa referente ao esquema de integração temporal são fixadas em 1×10^{-5} (**ATOL** = **RTOL** = 1×10^{-5}). A discretização espacial é realizada através de fórmulas de diferenças finitas com cinco pontos (**NPA** = 5), do tipo descentrado *biased upwind* (**TIPO(1)** = NCBIASED e **TIPO1(1)** = UP). Torna-se apenas necessário especificar a discretização relativamente à primeira variável, já que todas as restantes não envolvem diferenciações espaciais, definindo-se os parâmetros respectivos por espaços em branco. A execução é realizada através da utilização de interpolações por *Splines* cúbicas (**NINTPL** = 1), com cinco pontos (**NP** = 5). O modelo é definido por oito parâmetros (**NPR** = 8), cujos valores são armazenados no vector **THETA**. Por outro lado, fixa-se um valor máximo de dez para o nível máximo de refinamento (**ITER** = 10), sendo a execução efectuada com uma grelha inicial base de nível um não uniforme de 14 nodos (**NPD** = 14), cujas posições são lidas a partir do ficheiro de dados (**NGRINI** = 1), sendo introduzidas na primeira linha da matriz **Z**. A opção de redefinição da malha base não se encontra activada (**NGRINT** = 0), enquanto que a solução inicial é lida, na forma discreta (**NINIT** = 1), a partir de ficheiros de nome data.i correspondentes a cada uma das variáveis (**NAME(I)** = data.i, i = 210, 220, 230, 240, 250). Desde que contenham no máximo oito caracteres, os nomes escolhidos para estes

ficheiros são completamente arbitrários. No entanto, a sua estrutura não o é, tendo que obedecer a um certo conjunto de regras, já referidas anteriormente. As derivadas de segunda ordem são calculadas directamente, por manipulação dos valores da solução (**NDERV2** = 0), e todos os resultados disponíveis são listados pelo código (**NIPRINT** = 3). O domínio temporal não é subdividido em vários intervalos (**NSDT** = 1) e portanto, a integração terá necessariamente de se iniciar no primeiro intervalo (**NZONA** = 1). O passo temporal base é fixado em 200 (**DELTAT(1)** = 200), enquanto que o intervalo de impressão é de dois (**NPRT(1)** = 2), ou seja todos os perfis são imprimidos entre espaçamentos temporais de 2×200 = 400. Finalmente, o algoritmo é obrigado a verificar uma tolerância absoluta de 0.01 (**TOLER** = 0.01) para todas as variáveis sujeitas a diferenciação temporal (que neste caso são apenas três). Não é necessário definir qualquer tempo de transição entre subintervalos (**TCRIT**), porque não se recorreu a qualquer divisão do domínio temporal. De qualquer modo, é obrigatório criar a linha correspondente, que é deixada em branco.

Na construção do ficheiro, todas as variáveis têm de ser especificadas, mesmo que não estejam a ser utilizadas, com a excepção da matriz **Z** (para **NGRINI** = 0) e do vector **NAME** (quando **NINIT** = 0).

Deste modo, todas as informações referentes às condições pretendidas para a execução são introduzidas no código. Falta ainda definir as características específicas do modelo em si. Estas são associadas ao programa através da construção de três subrotinas adicionais. As subrotinas correspondentes à definição do exemplo 12 são apresentadas de seguida:

```
      SUBROUTINE INITIAL(J,Z,Y)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA

      IF(J.EQ.1)THEN
      Y=0.D0
      ELSEIF(J.EQ.2)THEN
      Y=0.D0
      ELSEIF(J.EQ.3)THEN
      Y=1.D0
      ELSEIF(J.EQ.4)THEN
      Y=0.D0
      ELSEIF(J.EQ.5)THEN
      Y=0.D0
      ENDIF

      RETURN
      END
```

Na subrotina INITIAL são introduzidas as expressões analíticas para as soluções iniciais referentes a cada variável. Neste caso, verifica-se que, há excepção da variável três (M), cujo perfil é unitário, todas as outras apresentam perfis iniciais constantes e nulos. No entanto, no caso da execução considerada, não é de todo necessária a especificação destes perfis, já que a solução inicial é lida a partir de um ficheiro, na forma discreta (**NINIT** = 1). Assim, a subrotina INITIAL podia perfeitamente ser definida como *dummy*. De qualquer modo, optou-se por a apresentar, de maneira a ilustrar a sua forma, caso se pretendesse a sua utilização.

```
      SUBROUTINE BOUND(T,LBE,LBD,LFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION LBE(5),LBD(5),YZB(5,2),Y(5,2),DYBDT(5,2)
      DIMENSION ORDMAX(5),BD(5,2)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA
      COMMON /BODR1/ Y,YZB,DYBDT,BD
      COMMON /BODR2/ ORDMAX

      LBE(1)=1
      LBD(1)=1
      LBE(2)=1
      LBD(2)=1
      LBE(3)=1
      LBD(3)=1
      LBE(4)=1
      LBD(4)=1
      LBE(5)=1
      LBD(5)=1

      IF(LFLAG.EQ.1)RETURN

      YZB(1,1)= THETA(1)*(Y(1,1)-1.D0)
      YZB(1,2)= 0.0

      BD(2,1)= 1
      BD(2,2)= 1
      BD(3,1)= 1
      BD(3,2)= 1
      BD(4,1)= 1
      BD(4,2)= 1
      BD(5,1)= 1
      BD(5,2)= 1

      ORDMAX(1)=2
      ORDMAX(2)=0
      ORDMAX(3)=0
      ORDMAX(4)=0
      ORDMAX(5)=0

      RETURN
      END
```

A subrotina BOUND informa o código sobre as características gerais definidas para as fronteiras a que o modelo está sujeito. No caso de **LFLAG** = 1, a análise é apenas superficial, sendo apenas fornecidas informações sobre o comportamento temporal de cada fronteira. Neste caso, verifica-se que as fronteiras de cada variável evoluem no tempo de uma forma não especificada (todos os **LBE** e **LBD** são fixados em 1). Caso alguma das fronteiras fosse fixa (condição de *Dirichlet*) ou a sua variação fosse perfeitamente especificada, o valor do parâmetro (**LBE** para a fronteira esquerda e **LBD** para a direita) deveria ser nulo.

Na segunda chamada da subrotina a partir de RES (**LFLAG** = 0), são introduzidas informações mais pormenorizadas. Assim, definem-se os valores fixados para a primeira derivada espacial da primeira variável (u), em cada fronteira, referentes às condições de *Neumann* aí especificadas pelo modelo (**YZB(1,1)** para a fronteira direita e **YZB(1,2)** para a esquerda). A solução fronteira referente às restantes variáveis pode variar livremente, ou seja o vector **BD(I,J)** com I = 2, … , 5 e J = 1, 2 é fixado em 1. As componentes desse vector correspondentes à variável um (I = 1) são nulas, já que neste caso, existe uma expressão que condiciona a sua variação em cada fronteira, apesar de não o fazer de uma forma explícita. Por outro lado, o modelo somente define uma diferenciação espacial de segunda ordem em relação à primeira variável (u), enquanto que nenhuma das outras variáveis envolve qualquer derivada em ordem ao espaço. Desse modo, **ORDMAX(1)** = 2, sendo os restantes componentes nulos. É necessário frisar que, se qualquer parâmetro especificado nesta

subrotina apresente um valor nulo, essa atribuição não precisa de ser realizada explicitamente, porque todas as variáveis assumem esse valor por defeito.

```
      SUBROUTINE MODELO(Y,YZ,YZZ,J,EQ)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(5),YZ(5),YZZ(5)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA

      IF(J.EQ.1)THEN
      EQ = 1.D0/THETA(1)*YZZ(1)-YZ(1)-THETA(2)*(Y(1)-Y(4))
      ELSEIF(J.EQ.2)THEN
      EQ = THETA(3)*(Y(5)-Y(2))-THETA(4)*Y(2)*Y(3)
      ELSEIF(J.EQ.2)THEN
      EQ = -THETA(5)*THETA(4)*THETA(6)*Y(2)*Y(3)
      ELSEIF(J.EQ.4)THEN
      EQ = (Y(1)-Y(4))-THETA(3)/THETA(2)*THETA(7)*(Y(5)-Y(2))
      ELSEIF(J.EQ.5)THEN
      EQ = Y(5)-THETA(8)*Y(4)/(1.D0+(THETA(8)-1.D0)*Y(4))
      ENDIF

      RETURN
      END
```

Finalmente, na subrotina MODELO são introduzidas as expressões referentes às funções **f** e **g** do modelo (correspondentes ao modelo geral – vd. Equações (D.1) e (D.2)). O valor dessas funções é definido na variável **EQ**, referente a cada equação J. É essencial referir que, na construção desta subrotina, é obrigatório considerar inicialmente as funções relacionadas com as equações diferenciais (funções **f** para as variáveis u, v e M) e só depois se introduzem as expressões algébricas (funções **g** para as variáveis u^* e v^*). Deste modo, os primeiros três componentes dos vectores **Y**, **YZ** e **YZZ**, correspondem às variáveis que são diferenciadas temporalmente (u, v e M). É óbvio que se torna igualmente necessário conferir se a ordem de colocação dos parâmetros do modelo no vector **THETA**, coincide com a ordem em que estes são introduzidos a partir do ficheiro de dados DATA.

D.2 – Algoritmo de Malha Móvel Adaptativa - Programa MMOVEL

D.2.1 – Estrutura Geral.

O código MMOVEL apresenta uma estrutura bastante semelhante à do programa REFIN, analisado na secção anterior, procedendo à execução do algoritmo de malha móvel adaptativa em problemas diferenciais, explícitos em relação às derivadas temporais, com a seguinte forma geral:

$$u^i{}_t = f_i\left(u, u_z, u_{zz}\right), \qquad i = 1, \cdots, \mathrm{NPDE} \tag{D.7}$$

Condições fronteira:

$$u\left(z^L, t\right) = u^L \tag{D.8}$$

$$u\left(z^R, t\right) = u^R \tag{D.9}$$

Condição inicial: $$u(z,0) = u^0(z) \quad ; \quad z \in \left[z^L, z^R\right] \qquad \text{(D.10)}$$

em que u(z,t) é o vector solução definido da mesma forma como anteriormente.

Para o efeito, foi elaborado um programa em FORTRAN 77, cuja estrutura geral é resumida na Figura D.2, para uma melhor visualização das diferentes rotinas que constituem o código global.

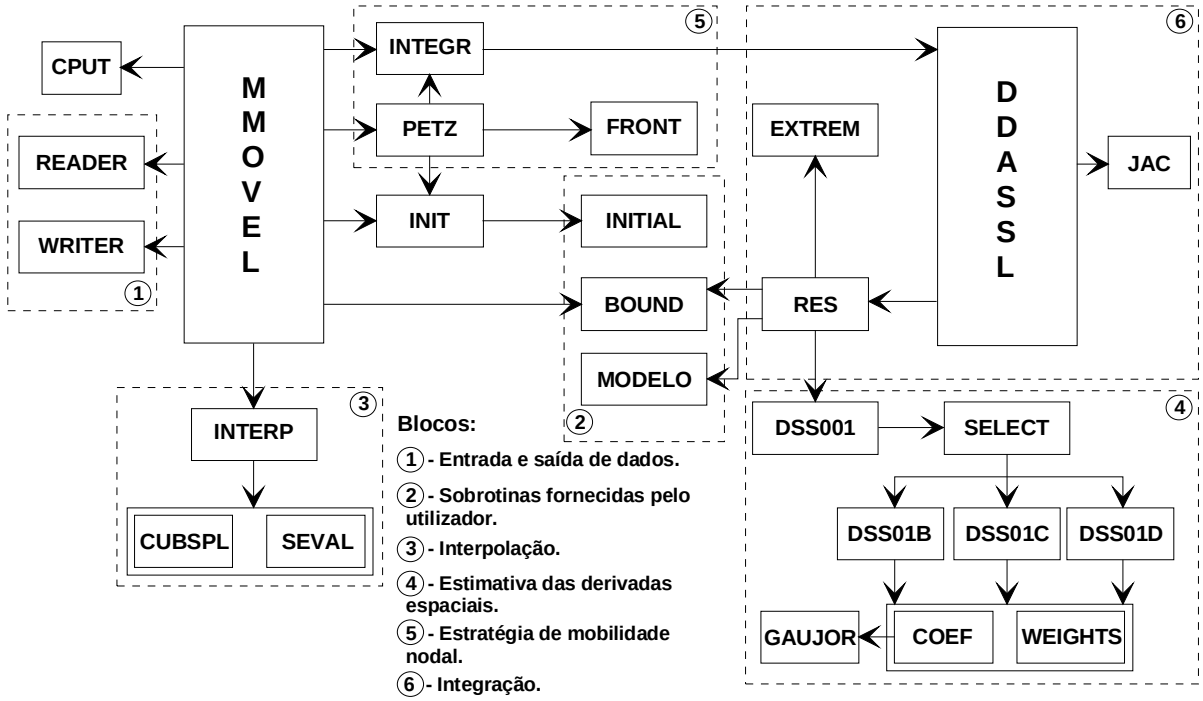

Figura D.2: Esquema simplificado da estrutura do código de malha móvel adaptativa.

O núcleo principal do código consiste no programa principal MMOVEL, concebido de forma muito semelhante à do programa REFIN, onde se executam as seguintes operações gerais:

❶ Determinação do tempo de computação, através da chamada da subrotina CPUT, exactamente da mesma forma como na execução do programa anterior.

❷ Introdução dos dados e listagem dos resultados, efectuados pelas chamadas das subrotinas READER e WRITER, respectivamente.

❸ Avaliação resumida do tipo de condições fronteira do problema, pela chamada da subrotina BOUND (com **LFLAG** = 1).

❹ Inicialização das integrações com malhas fixas, executadas com a chamada da subrotina INTEGR. Para o passo temporal inicial, a correspondente inicialização é efectuada pela chamada da subrotina INIT, equivalente à rotina referida no caso do código de refinamento.

❺ Cálculo e comparação dos erros espaciais e consequente selecção dos nodos "insatisfatórios" da malha base. A aplicação da fase de integração dos subproblemas dinâmicos, assim gerados, é executada com a chamada da subrotina que implementa o algoritmo PETZ.

❻ Ajuste do passo temporal, caso ocorram dificuldades de desempenho do integrador, ou cruzamentos nodais, através de um procedimento igual ao descrito na secção anterior.

❼ Redefinição estática das malhas (facultativa), caso seja exigida a transposição da solução nas malhas móveis, para a malha base inicial, em cada passo temporal.

❽ Redefinição dinâmica das malhas, caso a opção anterior não esteja activada, por introdução de nodos adicionais, entre pontos demasiado afastados e o afastamento compulsivo de nodos consecutivos excessivamente próximos. As interpolações necessárias na implementação destas redefinições, são efectuadas através da chamada da subrotina INTERP.

❾ Mecanismo de comparação das posições nodais de modo a identificar o instante temporal em que ocorra a separação de malhas correspondentes a diferentes variáveis (controlado pelo parâmetro interno **ISTATUS**).

As restantes subrotinas que constituem o código são igualmente, agrupadas em seis blocos (vd. Figura D.2), que executam funções equivalentes às descritas na secção anterior, para o algoritmo de refinamento, verificando-se, mesmo que, alguns dos blocos são perfeitamente coincidentes. Assim, para cada bloco, tem-se que:

Bloco ①: Entrada e saída de dados.

O Bloco ① é constituído por duas subrotinas, exactamente como no caso do código REFIN: a subrotina READER que procede à leitura dos dados; e a subrotina WRITER, que executa a escrita dos resultados.

Como anteriormente, o ficheiro de dados, designado por DATA, é construído pelo utilizador e define todas as características necessárias para a execução. Os parâmetros possíveis de especificar pelo utilizador no ficheiro DATA são, no entanto, algo diferentes dos correspondentes ao programa de refinamento. Tal seria de esperar, já que as estratégias de avaliação das malha óptimas para o avanço da integração são completamente diferentes em cada um dos algoritmos. No entanto, as estruturas de ambos os ficheiros são muito parecidas, sendo os parâmetros a definir, para este caso, resumidos na Tabela D.3, juntamente com a sua influência na evolução de cada execução.

Tabela D.3: Resumo das variáveis de entrada do programa MMOVEL fornecidas pelo ficheiro DATA.

Nome da Variável	*Tipo de Variável*	*Valores que Assume*	*Significado*	*Observações*
NPDE	Escalar Inteira		Nº de equações total do modelo	
ZE	Escalar Dupla Precisão		Posição da fronteira esquerda	
ZD			Posição da fronteira direita	
TINI			Instante de tempo inicial	Tempo inicial da integração
TFIN			Instante de tempo final	Tempo final da integração
ATOL			Tolerância absoluta	Tolerâncias referentes ao integrador
RTOL			Tolerância relativa	
NPI	Escalar Inteira		Nº de pontos de discretização	Referente ao tipo de diferenças finitas escolhido
TIPO(I)	Vector Caracter	CENTRAIS	Tipo de diferenças finitas	Diferenças centradas
		NCENTRAL		Diferenças descentradas; I = 1, … , **NPDE**
		NCBIASED		Diferenças descentradas *biased*
TIPO1(I)		UP	Orientação predominante das diferenças finitas	Diferenças *Upwind* ; I = 1, … , **NPDE**
		DOWN		Diferenças *Downwind*
NP	Escalar Inteira		Nº de pontos de interpolação	Apenas necessário se **NINTPL** = 1
NPR			Nº de parâmetros do modelo	
NINTPL		0	Parâmetro que controla o tipo de interpolação	Interpolações lineares
		1		Interpolações de *Splines* cúbicas com **NP** pontos
NGRINI		0	Parâmetro que controla o modo como é lida a malha base	Malha de nível 1 uniforme com **NPDINI** pontos
		1		Malha de nível 1 introduzida como dado
NINIT		0	Parâmetro que controla a forma como são lidas as condições iniciais	Através da sua forma analítica (subr. INITIAL)
		1		Na forma discreta, por um ficheiro de dados
NDERV2		0	Parâmetro que define a forma como é calculada a 2ª derivada	Por utilização directa da solução
		1		Por diferenciação da primeira derivada
NRDEF		0	Parâmetro que controla a forma de redefinição da malha	Liberdade para o movimento dos nodos
		1		Solução interpolada para a malha base inicial
NTTOL		0	Parâmetro que controla o tipo de tolerâncias usada para o algoritmo	Tolerâncias absolutas
		1		Tolerâncias relativas
NITER			Nº de subintervalos temporais	No passo base, na integração das malhas fixas
NMETH		1	Parâmetro que define o tipo de equação de mobilidade nodal usada	Aplicação da equação geral
		2		Equação geral com termo de penalidade
		3		Introdução dos pesos para o ajuste da escala
DZMIN	Escalar D. P.		Espaçamento internodal mínimo	Relevantes apenas se **NRDEF** = 0
DZMAX			Espaçamento internodal máximo	
ALFA			Constante de mobilidade nodal	Aplicada em todas as equações de mobilidade
RLAMBDA			Factor de viscosidade internodal	Apenas necessária se **NMETH** > 1
THETA(I)	Vector D. P.		Valores dos parâmetros do modelo	I = 1, … , **NPR**
NPDINI	Escalar Inteira		Nº de pontos da malha base inicial	Malha de nível 1
NSDT			Nº de subzonas temporais	Nº de partições do domínio temporal
NZONA			Subzona temporal de arranque	Partição em que a integração arranca
DELTAT(I)	Vector D. P.		Passos de tempo base	I = 1, … , **NSDT**
NPRT(I)	Vector Inteiro		Intervalos de impressão	I = 1, … , **NSDT**
TOLER(I,J)	Matriz D. P.		Tolerâncias para o algoritmo	I = 1, … , **NPDE** ; J = 1, … , **NSDT**
TCRIT(I)	Vector D. P.		Tempos de transição entre zonas	I = 1, … , **NSDT**-1 ; Para **NSDT** > 1
Z1(J)			Posições na malha base inicial	J = 1, … , **NPDINI** ; Apenas para **NGRINI** = 1
NAME(I)	Vector Car.		Nomes dos ficheiros de leitura	I = 1, … , **NPDE** ; Apenas para **NINIT** = 1
FLOOR(I)	Vector D. P.		Valores mínimos para os pesos	I = 1, … , **NPDE** + 1; Apenas para **NMETH** = 3

No que diz respeito à escrita de resultados, o programa imprime os diversos resultados disponíveis em ficheiros específicos, denominados por "fort.**n**", onde **n** corresponde a:

n = 50 + i ⇨ Perfil inicial (t = **TINI**) da variável **i**;
n = 200 + 10 × i ⇨ Solução nas grelhas já redefinidas, para a variável **i**;
n = 100 + 10 × i ⇨ Solução nas ainda não redefinidas, para a variável **i**;
n = 500 + 10 × i ⇨ Evolução temporal das posições nodais nas grelhas base de nível 2, para a variável **i**;
n = 50 ⇨ Tempo de computação da execução.

Neste programa não é utilizado nenhum parâmetro que controle o tipo de resultados que são imprimidos em cada execução. Assim, todos os perfis referidos acima são postos à disposição do utilizador, nos ficheiros correspondentes, para os instantes temporais por ele especificados.

Bloco ②: Subrotinas fornecidas pelo utilizador.

As subrotinas que constituem o bloco ② são fornecidas pelo utilizador, sendo exactamente equivalentes às aplicadas no código de refinamento. Ambos os programas foram elaborados de forma a que as subrotinas adicionais, introduzidas do exterior, fossem precisamente as mesmas. Assim, um bloco de rotinas que caracterize completamente um determinado modelo pode ser utilizado em cada um dos códigos, sem que haja necessidade de se efectuar qualquer adaptação.

Bloco ③: Subrotinas que implementam as operações de interpolação.

Devido à necessidade de se recorrer à interpolação de uma forma mais frequente, na implementação do código MMOVEL, optou-se por se utilizar uma subrotina específica para a execução desta operação, nas situações onde se revela necessária. Depois de seleccionada a dimensão do vector de pontos de interpolação (definido pelo parâmetro **NP**), introduz-se, na subrotina INTERP, o vector base dos pontos de interpolação e a abcissa onde se pretende estimar a solução. Esta rotina utiliza a subrotina CUBSPL para o cálculo dos pesos correspondentes a cada ponto, através da utilização de *Splines* cúbicas (no caso de **NINTPL** = 1) e, seguidamente, recorre a esses pesos e à função SEVAL, para a estimativa da solução interpolada na abcissa pretendida. No caso de **NINTPL** = 0, todas as interpolações são efectuadas por aproximação linear.

Bloco ④: Subrotinas para a estimativa das derivadas espaciais.

O bloco ④ engloba todas as subrotinas utilizadas na estimativa do valor das derivadas espaciais através de aproximações de diferenças finitas, sendo exactamente igual ao aplicado no código de refinamento, e já descrito com algum pormenor, na secção anterior.

Bloco ⑤: Subrotinas que implementam a estratégia de movimentação nodal.

O bloco ⑤ é constituído por todas as subrotinas relacionadas com a implementação do algoritmo de movimentação nodal. Para tal, a subrotina PETZ procede à construção dos subdomínios iniciais, referentes às malhas correspondentes a cada variável, e à consequente inicialização dos subproblemas dinâmicos assim gerados, através da inclusão do vector das posições nodais na integração. A integração temporal destes subproblemas é efectuada pela chamada da subrotina INTEGR que condiciona todos os parâmetros de execução do integrador implícito DDASSL. No caso de se tratar do primeiro passo de integração, a rotina PETZ recorre à subrotina INIT para a inicialização destes problemas.

Implementa-se, igualmente, o esquema iterativo para a integração de cada subproblema de forma a assegurar a convergência das interfaces entre as secções dos perfis calculadas por integração dinâmica e as obtidas pela resolução do problema estático geral, de forma a impedir a ocorrência de descontinuidades nos perfis globais. Quando os valores da solução, calculadas nos pontos fronteira de cada subdomínio, nas duas fases do algoritmo, diferem de um valor superior a uma determinada tolerância, o subdomínio em questão é alargado em um nodo na direcção dessa fronteira. Nesse caso, o subproblema dinâmico é

integrado novamente, no subdomínio redefinido, até que se obtenha convergência em ambas as fronteiras.

A subrotina FRONT é utilizada para a identificação dos nodos adicionais, referentes a cada subdomínio, dependentes do tipo de discretização considerado e da posição relativa desse subdomínio no domínio espacial global. Estas posições, e as respectivas soluções, são apenas aplicadas na avaliação da derivada espacial nos pontos fronteira, ao longo da integração temporal, não intervindo directamente na mesma.

Bloco ⑥: Subrotinas referentes à integração temporal.

Como no código anterior, o núcleo central do bloco ⑥, referente à integração temporal, consiste na *package* DDASSL que engloba as subrotinas que asseguram a integração temporal do problema original (integração estática) e dos subproblemas gerados (integrações dinâmicas). A subrotina JAC, onde se processa o cálculo analítico dos componentes da matriz jacobiana é uma rotina *dummy*, já que, como anteriormente, essa opção não se encontra activada. Por outro lado, a subrotina RES, que processa o cálculo dos resíduos relacionados com as equações do modelo é bastante semelhante à utilizada no código de refinamento. Deste modo esta rotina, além de calcular os resíduos (através da chamada da subrotina MODELO), define as características fronteira de cada problema ou subproblema integrado (pela chamada da subrotina BOUND) e procede ao cálculo das estimativas para as derivadas espaciais (recorrendo à rotina DSS001). Efectua, igualmente a selecção do problema a integrar (estático ou dinâmico, especificado pelo valor do parâmetro interno **IMETH**) e, dentro dos modelos dinâmicos, procede à selecção das equações de mobilidade nodal a aplicar (processo que é controlado pelo parâmetro de entrada **NMETH**). No caso de **NMETH** = 3, a utilização da expressão respectiva implica a avaliação do valor de pesos associados a cada variável. Estes pesos dependem do valor dos extremos verificados nos conjuntos das soluções, associados a cada variável (incluindo o vector das posições nodais). Os valores destes extremos (máximos e mínimos) são obtidos pela chamada de uma subrotina auxiliar designada por EXTREM. A evolução temporal da solução nos pontos vizinhos das fronteiras é estimada por interpolação linear dos valores disponíveis, obtidos na integração das malhas fixas.

A forma como o código é, de momento, definido, obriga à verificação de algumas limitações, resumidas da seguinte maneira:

Número máximo de equações do modelo (**NPDE**)	⇨	5
Número máximo de parâmetros do modelo (**NPR**)	⇨	10
Número máximo de subdomínios gerados em cada passo (Variável Interna)	⇨	10
Número máximo de subintervalos no domínio temporal (**NSDT**)	⇨	8
Número máximo de nodos para a malha base de nível 1 (**NPDINI**)	⇨	201
Dimensão máxima do vector de nodos de interpolação (**NP**)	⇨	10
Número máximo de passos temporais intermédios em cada passo base (**NITER**)	⇨	10
Dimensão máxima do vector de nodos de discretização (**NPI**)	⇨	15

Da mesma forma do caso anterior, estes valores máximos podem ser alterados por manipulação das dimensões dos vectores ou matrizes adequados. No entanto, há de ter em conta que estas alterações podem originar problemas como os já anteriormente referidos, relacionados principalmente com limitações de memória.

Durante a execução do algoritmo, um dos parâmetros internos do código (**ISTATUS**) controla a utilização de uma estratégia mono ou multi-malha, na integração das malhas fixas. A monitorização deste parâmetro apenas faz sentido no caso da resolução de sistemas de equações diferenciais. Se **ISTATUS** = 0 (valor de arranque), todas as variáveis estão ainda a ser integradas em malhas coincidentes, e assim, a fase de estimativa do erro apenas é executada uma única vez. Pelo contrário, no caso de **ISTATUS** tomar o valor de 1, verifica-se a ocorrência de uma separação de malhas, e todo o procedimento terá de ser executado para cada uma das variáveis do modelo.

D.2.2 – Apresentação de um Exemplo.

Na presente secção, procede-se à apresentação de todos dados que o utilizador necessita de fornecer ao código para a execução de um exemplo estudado neste trabalho. Optou-se por considerar aqui, um problema diferente do discutido na Secção D.1.2. de forma se ilustrar a variedade de opções postas à disposição por cada um dos códigos, e o tratamento a dar no caso de um modelo substancialmente distinto do anterior. O problema considerado refere-se a um sistema de duas equações diferenciais parciais que descreve o comportamento de propagação de uma chama (**Exemplo 6**). As condições da execução não correspondem às utilizadas em nenhum dos *runs* estudados para este exemplo, pretendendo, apenas, exemplificar a aplicação de um vasto leque de opções disponíveis.

Deste modo, considera-se o seguinte ficheiro DATA, que define um conjunto possível de parâmetros de execução do algoritmo de malha móvel adaptativa, para o modelo em questão:

```
2                                                   NPDE
0.D0,1.D0                                           ZE,ZD
0.D0,6.0D-3                                         TINI,TFIN
1.D-6,1.D-6                                         ATOL,RTOL
5                                                   NPI
CENTRAIS,DOWN                                       TIPO(I),TIPO1(I)
CENTRAIS,DOWN                                       I=1, ... , NPDE
3,2                                                 NP,NPR
0,1,0                                               NINTPL,NGRINI,NINIT
0,0,0,1                                             NDERV2,NRDEF,NTTOL,NITER
2,1.D-4,1.D-2                                       NMETH,DZMIN,DZMAX
1.D0,0.60D0                                         ALFA,RLAMBDA
3.52D+6,4.D0                                        THETA(I), I=1, ... , NPR
20                                                  NPDINI
2,1                                                 NSDT,NZONA
1.0D-4,6.D-4                                        DELTAT(I), I=1, ... , NSDT
3,2                                                 NPRT(I), I=1, ... , NSDT
5.D-3,5.D-3                                         TOLER(I,J),    I=1, ... , NPDE
5.D-3,5.D-3                                                        J=1, ... , NSDT
6.D-4                                               TCRIT(I), I=1, ... , NSDT-1
0.D0                                                Z1(J)
0.2D0                                               J=1, ... , NPDINI
0.3D0
0.4D0
0.45D0
0.5D0
0.55D0
0.6D0
0.65D0
0.7D0
0.75D0
0.8D0
0.825D0
0.85D0
0.875D0
0.9D0
0.925D0
0.95D0
0.975D0
1.D0
```

Assim, o problema é constituído por duas equações diferenciais (**NPDE** = 2). O domínio espacial coincide com o intervalo [0 , 1] (**ZE** = 0.0 e **ZD** = 1.0), e pretende-se que a integração temporal se realize entre os instantes t = 0 e t = 0.0060 (**TINI** = 0.0 e **TFIN** = 0.0060). As tolerâncias absoluta e relativa do esquema de integração temporal são fixadas em 1×10^{-6} (**ATOL** = **RTOL** = 1×10^{-6}). Aplica-se uma discretização espacial com fórmulas de diferenças finitas de cinco pontos (**NPI** = 5), do tipo centrado para as duas variáveis (**TIPO** = CENTRAIS, sendo os parâmetros **TIPO1** irrelevantes). A execução utiliza interpolações lineares (**NINTPL** = 0, enquanto que **NP** não interessa). O modelo é definido por dois parâmetros (**NPR** = 2), introduzidos no vector **THETA**. A execução é efectuada com uma grelha inicial base de nível um não uniforme de 20 nodos (**NPDINI** = 20), cujas posições são lidas a partir do ficheiro de dados (**NGRINI** = 1), sendo armazenadas no vector **Z1**. A solução inicial é introduzida, na sua forma analítica (**NINIT** = 0), a partir da subrotina INITIAL. As derivadas de segunda ordem são calculadas directamente, por manipulação dos valores da solução (**NDERV2** = 0), e a opção de redefinição da malha dinâmica para a sua correspondente inicial encontra-se desactivada (**NRDEF** = 0). As tolerâncias especificadas para o algoritmo são do tipo absoluto (**NTTOL** = 0), enquanto que cada passo temporal não é subdividido em qualquer das integrações estáticas (**NITER** = 1). Como se permite a livre movimentação dos nodos, fixa-se os seus espaçamentos máximo e mínimo admissíveis nos valores **DZMIN** = 1×10^{-4} e **DZMAX** = 0.01, respectivamente. Na integração dos subproblemas dinâmicos aplica-se a equação de mobilidade nodal com um termo de penalização (**NMETH** = 2), sendo para tal especificado o valor do correspondente coeficiente de viscosidade (**RLAMBDA** = 0.6, com **ALFA** = 1.0). O domínio temporal é subdividido em dois intervalos (**NSDT** = 2), iniciando-se a integração no primeiro daqueles (**NZONA** = 1). O instante de tempo onde se dá a transição situa-se em t = 0.0006 (**TCRIT(1)** = 0.0006). O passo temporal base para o primeiro intervalo é fixado em 0.0001 (**DELTAT(1)** = 0.0001), enquanto que no segundo se utiliza um passo de 0.0006 (**DELTAT(2)** = 0.0006). O intervalo de impressão é de três (**NPRT(1)** = 3), no caso do primeiro intervalo temporal, e assim, todos os perfis são imprimidos entre espaçamentos temporais de $3\times0.0001 = 0.0003$. No que diz respeito ao segundo intervalo, o período de impressão é de dois, o que corresponde à escrita dos resultados entre espaçamentos de $2\times0.0006 = 0.0012$ (**NPRT(2)** = 2), a partir do instante de tempo inicial (isto porque, neste caso, **NPRT(2)**×**DELTAT(2)** > **TCRIT(1)**). Finalmente, o algoritmo é obrigado a verificar uma tolerância absoluta de 5×10^{-3} (**TOLER(I,J)** = 5×10^{-3}) para todas as variáveis, em ambos os subintervalos temporais.

Como anteriormente, a construção do ficheiro implica que todas as variáveis têm de ser especificadas, mesmo que não estejam a ser utilizadas, com a excepção do vector **Z1**, que neste caso é necessário, porque **NGRINI** = 1, e do vector **NAME**, que no presente exemplo é irrelevante, já que **NINIT** = 0.

Deste modo, todas as informações referentes às condições pretendidas para a execução são introduzidas no código. Para definir as características específicas do modelo em si associam-se ao programa três subrotinas adicionais É importante salientar que estas subrotinas podem perfeitamente ser aplicadas no código de refinamento, apresentado na secção anterior, sem que seja necessário proceder a qualquer alteração. As rotinas correspondentes à descrição do modelo em estudo são as seguintes.

```
      SUBROUTINE INITIAL(J,Z,Y)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA

      IF(J.EQ.1)THEN
      Y=1.D0
      ELSEIF(J.EQ.2)THEN
      Y=0.2D0
      ENDIF

      RETURN
      END
```

Com a definição da subrotina INITIAL introduzem-se no sistema, as expressões analíticas para as soluções iniciais referentes a cada variável. Neste caso, constata-se que, ambas as variáveis apresentam perfis iniciais constantes com os valores: **Y** = 1.0 para a primeira variável (u) e **Y** = 0.2 para a segunda (v).

```
      SUBROUTINE BOUND(T,LBE,LBD,LFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION LBE(5),LBD(5),YZB(5,2),Y(5,2),DYBDT(5,2)
      DIMENSION ORDMAX(5),BD(5,2)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA
      COMMON /BODR1/ Y,YZB,DYBDT,BD
      COMMON /BODR2/ ORDMAX

      LBE(1)=1
      LBD(1)=1
      LBE(2)=1
      LBD(2)=0

      IF(LFLAG.EQ.1)RETURN

      YZB(1,1)= 0.0
      YZB(1,2)= 0.0
      YZB(2,1)= 0.0

      IF(T.LE.2.D-4)THEN
      YZB(2,2)=0.2+T/2.D-4
      DYBDT(2,2)=1.D0/2.D-4
      ELSE
      YZB(2,2)=1.2
      DYBDT(2,2)=0.0
      ENDIF

      ORDMAX(1)=2
      ORDMAX(2)=2

      RETURN
      END
```

Na subrotina BOUND introduzem-se as características referentes à condições fronteira especificadas para o modelo em estudo. Neste caso, verifica-se que as fronteiras de cada variável evoluem no tempo de uma forma não especificada (todos os **LBE** e **LBD** são fixados em 1), há excepção de uma: a fronteira da direita para a segunda variável, onde a evolução da solução é perfeitamente definida (**LBD(2)** = 0). Nesta fronteira, aplica-se uma perturbação em rampa na variável v, de declive 1/0.0002 (**DYBDT(2,2**) = 1/0.0002), partindo do ponto 0.2 (**YZB(2,2)** = 0.2 + **T**/0.0002) até ao instante **T** = 1.2, a partir do qual a condição

fronteira se torna constante (**YZB(2,2)** = 1.2 e **DYBDT(2,2)** = 0.0). As restantes fronteiras são regidas por condições de *Neumann*, fixando-se valores nulos para as primeiras derivadas correspondentes (**YZB** = 0.0 nas três fronteiras restantes). O modelo define diferenciações espaciais de ordem máxima dois, em relação às duas variáveis (u e v). Desse modo, especifica-se que **ORDMAX(1)** = **ORDMAX(2)** = 2.

```
      SUBROUTINE MODELO(Y,YZ,YZZ,J,EQ)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(5),YZ(5),YZZ(5)
      DIMENSION THETA(10)
      COMMON /MODEL/ THETA

      IF(J.EQ.1)THEN
      EQ = YZZ(1)-THETA(1)*Y(1)*DEXP(-THETA(2)/Y(2))
      ELSEIF(J.EQ.2)THEN
      EQ = YZZ(2)+THETA(1)*Y(1)*DEXP(-THETA(2)/Y(2))
      ENDIF

      RETURN
      END
```

Finalmente, na subrotina MODELO são introduzidas as expressões referentes às funções **f** do modelo diferencial (correspondentes ao modelo geral – vd. Equação (D.7)). O valor de cada uma das funções é definido na variável **EQ**. Torna-se igualmente necessário conferir se a ordem dos parâmetros do modelo no vector **THETA**, coincide com a ordem em que estes são lidos a partir do ficheiro de dados DATA.