



“Modelagem Automática de Escoamentos em Meios
Porosos via Método dos Elementos Finitos”

Por

Bruno Gustavo Borges Luna

Dissertação de Mestrado

Recife, Fevereiro de 2012



UNIVERSIDADE FEDERAL DE PERNAMBUCO

Bruno Gustavo Borges Luna

**“Modelagem Automática de Escoamentos em Meios
Porosos via Método dos Elementos Finitos”**

*Trabalho apresentado ao Programa de Pós-graduação em
Engenharia Mecânica do Centro de Tecnologia e Geociên-
cias da Universidade Federal de Pernambuco como requisito
parcial para obtenção do grau de Mestre em Engenharia
Mecânica.*

Orientador: *Paulo Roberto Maciel Lyra*
Co-Orientador: *Ramiro Brito Willmersdorf*

L961m Luna, Bruno Gustavo Borges.
“Modelagem Automática de Escoamentos em Meios Porosos via Método dos Elementos Finitos” / Bruno Gustavo Borges Luna – Recife: O Autor, 2012.
xv, 94f. il., figs., gráfs., tabs.

Orientador: Prof. Dr. Paulo Roberto Maciel Lyra.
Co-Orientador: Prof. Dr. Ramiro Brito Willmersdorf.

Dissertação (Mestrado) – Universidade Federal de Pernambuco. CTG.
Programa de Pós-Graduação em Engenharia Mecânica, 2012.
Inclui Referências e Apêndices.

1. Engenharia Mecânica. 2. Escoamentos em Meios Porosos. 3. Métodos dos Elementos Finitos. 4. Modelagem Automática. I. Lyra, Paulo Roberto Maciel (Orientador). II. Willmersdorf, Ramiro Brito (Co-Orientador). III. Título.

Modelagem Automática de Escoamentos em Meios Porosos via Método dos Elementos Finitos

Bruno Gustavo Borges Luna

Trabalho apresentado ao Programa de Pós-graduação em Engenharia Mecânica do Centro de Tecnologia e Geociências da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Engenharia Mecânica.

Aprovada por:

Paulo Roberto Maciel Lyra, Ph.D.
(Orientador)

Ramiro Brito Willmersdorf, Ph.D.
(Co-Orientador)

Darlan Karlo Elisário de Carvalho, D.Sc.
(Examinador Interno)

Alessandro Romário Echevarria Antunes, D.Sc.
(Examinador Externo: CAA-UFPE)

Ézio da Rocha Araújo, D.Sc.
(Examinador Externo: DECIV-UFPE)

Recife, Fevereiro de 2012

*Aos meus pais,
à minha irmã
e à minha esposa*

Agradecimentos

Ao Prof. Paulo Lyra, cuja mentoria foi fundamental para a minha formação como profissional, pela orientação, confiança e apoio inestimáveis desde os tempos de iniciação científica. Agradeço especialmente pela amizade e paciência demonstradas ao longo dos anos.

Ao Prof. Ramiro Willmersdorf, pelos comentários sempre sucintos e pertinentes, e também por sua memorável palestra plenária no CILAMCE 2008, cujas ideias influenciaram bastante o rumo tomado neste trabalho.

Aos professores Ézio Araújo e Alessandro Antunes, pela participação na banca examinadora e pelos comentários para melhoria da dissertação.

Ao Prof. Darlan Carvalho, pelas várias sugestões para este trabalho e principalmente por todos os ensinamentos transmitidos ao longo dos anos.

Aos colegas de LABCOM (Rafael, Adriano, Rodrigo, Hélder, Danilo, Ana Paula, etc.), pelos momentos de trabalho e de descontração.

À Petrobras, pelo apoio constante ao grupo PADMEC.

À minha mãe, Neuce, e ao meu pai, Cosme, aos quais serei eternamente grato pelo apoio incondicional em todos os momentos, pelo incentivo à minha formação acadêmica e principalmente por serem os meus maiores orientadores graças a todos os valores ensinados através de exemplos diários.

À minha irmã, Monique, pelo carinho, amizade e por ser sempre um exemplo de dedicação e vontade.

À minha esposa, Caroline, por seu amor e por ter estado sempre ao meu lado durante esta jornada. Este trabalho é tão seu quanto meu. *Ich liebe dich!*

Resumo

A simulação de escoamentos multifásicos em meios porosos impõe vários desafios de ordem numérica devido a uma série de fatores, como os meios altamente anisotrópicos e heterogêneos tratados neste tipo de análise, as Equações Diferenciais Parciais (EDP) de natureza acoplada elíptica-hiperbólica que descrevem o fenômeno, entre outros. Uma vez definidas as formulações matemáticas e numéricas a serem utilizadas para modelar adequadamente o escoamento, encontra-se outra dificuldade na codificação destes métodos, já que, usualmente, despende-se um tempo considerável para o desenvolvimento de programas de computador que implementem formulações para casos gerais ou complexos. Este trabalho apresenta a implementação de um software criado utilizando a linguagem Python e a ferramenta computacional FEniCS para a geração automática de código de baixo-nível em C++ aplicado na solução numérica de escoamentos mono- e bifásicos em meios porosos usando o Método dos Elementos Finitos (MEF). Foram testados o MEF de Galerkin e o Método dos Elementos Finitos Mistos (MEFM) para a solução da equação da pressão (pressão e velocidade no caso do MEFM) e o MEF com estabilização via *Streamline Upwind Petrov Galerkin* (SUPG) e operador de captura de choque para a equação da saturação. Para a solução do sistema de equações lineares provenientes do MEF de Galerkin foram utilizadas técnicas de aceleração de convergência via Método Multigrid Algébrico (AMG). Os métodos descritos neste trabalho são gerais o suficiente para lidar com problemas tridimensionais, heterogêneos e anisotrópicos. Exemplos são apresentados e resultados discutidos para problemas uni- e bidimensionais com domínios homogêneos e heterogêneos com tensores de permeabilidade iso- e anisotrópicos. A comparação favorável dos resultados obtidos com soluções analíticas e referências da literatura demonstra o potencial da ferramenta desenvolvida para a simulação de escoamentos em meios porosos.

Palavras-chave: Escoamentos em Meios Porosos, Métodos dos Elementos Finitos, Modelagem Automática

Abstract

The simulation of multiphase flows in porous media imposes many numerical challenges due to a series of factors such as the high anisotropic and heterogeneous media handled in this type of analysis, the Partial Differential Equations (PDEs) with coupled elliptic-hyperbolic mathematical nature, among others. Even after the mathematical and numerical formulations used to model the flow are defined, there is still another challenge regarding the coding of these methods, because it is usually a very time-consuming task to develop computer programs that implement formulations for general and/or complex cases. This work presents the implementation of a software written using the Python programming language and the FEniCS computational tool for the automatic generation of low-level code in C++ applied to the numerical solution of mono- and biphasic flows in porous media using the Finite Element Method (FEM). The classical Galerkin FEM and the Mixed Finite Element Method (MFEM) were tested for the solution of the pressure (pressure and velocity for MFEM) and the Streamline Upwind Petrov Galerkin (SUPG) stabilized FEM with shock capturing operator for the saturation equation. A convergence acceleration technique via Algebraic Multigrid (AMG) was used for the solution of the linear system of equations derived from the Galerkin FEM discretization. The methods described here are general enough to handle three-dimensional, heterogeneous and anisotropic problems. Examples are shown and results discussed for one- and two-dimensional problems in homogenous and heterogenous domains with iso- and anisotropic permeability tensors. The comparisons of the results obtained in this work with those from analytical solutions and literature references show the potential of the developed tool for the simulation of flows in porous media.

Keywords: Flow in Porous Media, Finite Element Method, Automatic Modelling

Sumário

Lista de Figuras	x
Lista de Tabelas	xii
Lista de Acrônimos	xiii
Lista de Símbolos	xiv
1 Introdução	1
1.1 Motivação	1
1.2 Objetivos	3
1.3 Organização da Dissertação	4
2 Formulação Matemática	6
2.1 Propriedades	7
2.1.1 Porosidade	8
2.1.2 Permeabilidade Absoluta	9
2.1.3 Permeabilidade Relativa	10
2.2 Equação da Conservação de Massa	13
2.3 Equação da Pressão	14
2.4 Equação da Saturação	16
2.4.1 Equação de Buckley-Leverett	18
2.4.2 Equação do Deslocamento de Fluidos Miscíveis	18
2.5 Condições Iniciais e de Contorno	19
3 Formulação Numérica	22
3.1 Método dos Elementos Finitos de Galerkin	23
3.2 Método dos Elementos Finitos Mistos	24

3.3	Estabilização do Método dos Elementos Finitos (MEF) via <i>Streamline Upwind Petrov-Galerkin</i> (SUPG)	28
3.3.1	Termo de Captura de Choque	32
3.4	Multigrid	35
4	Implementação Computacional	40
4.1	Estrutura Geral do Programa	41
4.2	Pré-Processamento	41
4.3	Processamento	44
4.3.1	FEniCS	46
4.3.2	PyAMG	53
4.4	Pós-Processamento	54
5	Resultados	56
5.1	Problemas Elípticos	56
5.1.1	Meio Homogêneo e Isotrópico	57
5.1.2	Meio Homogêneo e Anisotrópico	61
5.1.3	Meio Heterogêneo e Anisotrópico	64
5.2	Escoamentos Bifásicos em Meios Porosos	67
5.2.1	Buckley-Leverett 1-D	67
5.2.2	1/4 de Cinco Poços Heterogêneo	71
6	Conclusões e Trabalhos Futuros	74
6.1	Conclusões	74
6.2	Trabalhos Futuros	76
	Referências Bibliográficas	78
A	Delfine - Manual do usuário	83
A.1	Dependências e <i>Download</i> do Programa	83
A.2	Dados de Entrada	84
A.2.1	Arquivo de Dados	85
A.2.2	Arquivo de Malha	89
A.3	Exemplo Detalhado	90

Lista de Figuras

2.1	Exemplo de meio poroso com destaque para os volumes de matriz sólida (rocha) e de poros (vazio).	9
2.2	Exemplo de meio poroso saturado com escoamento bifásico.	10
2.3	Exemplo de correlação entre porosidade e permeabilidade absoluta (retirado de Schneider (2003)).	11
2.4	Saturação normalizada.	12
2.5	Exemplo de modelo de permeabilidade relativa para escoamentos bifásicos em meios porosos.	13
2.6	Exemplo de função de fluxo fracional e sua derivada relativa à saturação da fase água.	19
3.1	Exemplo de elementos triangulares para análise via MEF.	25
3.2	Exemplo de elementos triangulares para análise via MEFM.	29
3.3	Comparação entre diferentes definições da função de ponderação (retirado de Monajemi (2009)).	30
3.4	Comparação entre solução analítica e numérica considerando o uso do MEF de Galerkin para o caso de advecção pura.	34
3.5	Comparação entre solução analítica e numérica considerando o uso do MEF com estabilização via SUPG para o caso de advecção pura.	34
3.6	Comparação entre solução analítica e numérica considerando o uso do MEF com estabilização via SUPG e adição de viscosidade artificial para o caso de advecção pura.	35
3.7	Comparação dos métodos Multigrid Geométrico e Algébrico (retirado de Trottenberg <i>et al.</i> (2001)).	37
3.8	Ciclos V, W e F. S representa suavização e E é a solução no nível mais grosseiro.	39
4.1	Fluxograma da etapa de pré-processamento.	45

4.2	Fluxograma da resolução da parte elíptica.	47
4.3	Fluxograma da resolução da parte hiperbólica.	48
4.4	Interação entre os diversos componentes do projeto <i>FEniCS</i> para definição do problema, seguidos pela resolução no <i>Delfine</i> (adaptado de Rathgeber (2010)).	49
4.5	Estrutura modular do <i>DOLFIN</i> (retirado de Logg e Wells (2010)). . . .	50
4.6	Fluxograma da etapa de pós-processamento.	55
5.1	Caso homogêneo e isotrópico. Campo escalar para malha 64×64 . . .	58
5.2	Caso homogêneo e isotrópico. Comparação da evolução dos resíduos para malha 32×32	61
5.3	Caso homogêneo e anisotrópico. Campo escalar para malha 64×64 . .	62
5.4	Caso homogêneo e anisotrópico. Comparação da evolução dos resíduos para malha 64×64	64
5.5	Caso heterogêneo e anisotrópico. Malha não-estruturada e heterogeneidade	67
5.6	Caso heterogêneo e anisotrópico. Campo escalar para malha de 64×64 .	68
5.7	Caso heterogêneo e anisotrópico. Comparação da evolução dos resíduos para malha 64×64	69
5.8	Representação de escoamento unidimensional imiscível de óleo por água (retirado de Hurtado (2005)).	69
5.9	Perfil de saturação para a resolução da equação de Buckley-Leverett . .	70
5.10	Descrição do problema de 1/4 de cinco poços	72
5.11	Campo de velocidade e representação extrudada da saturação	72
5.12	Campo de saturação e representação do perfil ao longo da diagonal entre poços para problema de 1/4 de cinco poços heterogêneo.	73

Lista de Tabelas

5.1	Erro e taxa de convergência para problema homogêneo e isotrópico . . .	59
5.2	Erro e taxa de convergência para problema homogêneo e isotrópico - literatura	59
5.3	Erro e taxa de convergência para problema homogêneo e anisotrópico .	63
5.4	Erro e taxa de convergência obtidos na literatura para problema homogê- neo e anisotrópico	63
5.5	Erro e taxa de convergência para problema heterogêneo e anisotrópico .	66

Lista de Acrônimos

AMG Multigrid Algébrico

BDM *Brezzi-Douglas-Marini*

CG Método dos Gradientes Conjugados

EDP Equações Diferenciais Parciais

FFC *FEniCS Form Compiler*

GMG Multigrid Geométrico

GMRES *Generalized Minimal Residual Method*

MDF Método das Diferenças Finitas

MEF Método dos Elementos Finitos

MEFM Método dos Elementos Finitos Mistos

MVF Método dos Volumes Finitos

RNC *Relax NG Compact Syntax*

RNV *Relax NG Validator*

SUPG *Streamline Upwind Petrov-Galerkin*

UFC *Unified Form-assembly Code*

UFL *Unified Form Language*

XML *Extensible Markup Language*

Lista de Símbolos

Escalares

λ	Mobilidade
μ	Viscosidade
ν	Viscosidade artificial
ϕ	Porosidade efetiva
ψ	Fator de descontinuidade no domínio
ρ	Densidade
τ	Parâmetro de estabilização
C	Concentração
f	Fluxo fracional
g	Aceleração da gravidade
k_r	Permeabilidade relativa
n	Quantidade total de fases
p	Pressão
q	Termo fonte/sumidouro
r	Resíduo da solução aproximada
S	Saturação
t	Tempo

V Volume

w Função de ponderação do MEF

Vetores

f Vetor de carregamentos

n Vetor normal ao contorno

v Velocidade

Matrizes

A Matriz resultante da discretização via MEF

D Tensor de Dispersão-Difusão

I Operador de transferência entre níveis

K Permeabilidade absoluta

Subscritos

α Relativo à fase α ($\alpha=w$ para água e $\alpha=o$ para óleo)

c Relativo ao nível menos refinado

f Relativo ao nível mais refinado

o Relativo à fase óleo

w Relativo à fase água

Domínios, Espaços

Γ Contorno do domínio Ω

Ω Domínio espacial do problema

H^k Espaço de Sobolev com derivadas de ordem k quadrado-integráveis

P Espaço das funções tentativa para a formulação de Galerkin

W Espaço das funções de ponderação para a formulação de Galerkin

Capítulo 1

Introdução

1.1 Motivação

Atualmente, a simulação numérica do escoamento de fluidos em meios porosos é fundamental em várias áreas da engenharia, como na análise de transporte de contaminantes em aquíferos (Bear *et al.*, 1992), no gerenciamento de água e calor em membranas poliméricas para células de combustível (Matamoros e Brueggemann, 2006) ou para a simulação de escoamentos multifásicos em reservatórios de petróleo (Peaceman, 1977; Aziz e Settari, 1979; Ewing, 1983; Carvalho, 2005; Silva, 2008).

A modelagem científica do escoamento de fluidos em meios porosos para a indústria do petróleo vem sendo feita desde os anos 30 do século passado, inicialmente utilizando modelos de areia compactada para compreender o mecanismo da produção de água em reservatórios de petróleo e o porquê da proporção da água sobre o óleo aumentar ao longo do tempo. No final dos anos 30 e começo dos 40 alguns experimentos com modelos eletrolíticos foram usados para representar o escoamento em meios porosos, tendo sido nesta época já reconhecida a analogia entre as leis que regem o fluxo de corrente elétrica e as leis de Darcy (Peaceman e Nash, 1990). Todos esses modelos físicos análogos permitiram a obtenção de um conhecimento mais profundo sobre os fenômenos que governam este tipo de problema, porém apenas com o advento dos computadores eletrônicos é que foi possível para os engenheiros simularem problemas de forma mais realista, como o de reservatórios de petróleo em 2 ou 3 dimensões.

Paralelamente ao desenvolvimento de novos *hardwares*, houve também um trabalho extensivo em métodos computacionais aplicados ao escoamento multifásico em meios porosos. O primeiro método utilizado em larga escala para resolver Equações Diferenciais Parciais (EDP) numericamente, e ainda padrão na indústria do petróleo, foi o Método

das Diferenças Finitas (MDF) (Peaceman e Nash, 1990). Entretanto, este tipo de método apresenta certas desvantagens, como a dificuldade para lidar com domínios complexos, condições de contorno gerais e para incorporar adaptação local de malhas (Chen *et al.*, 2006), especialmente quando comparado com métodos mais adequados para lidar com malhas não-estruturadas que vêm sendo aplicados mais recentemente nesta área como é o caso do Método dos Volumes Finitos (MVF) (Carvalho, 2005; Cordazzo, 2006) ou o Método dos Elementos Finitos (MEF) (Chen *et al.*, 2006). Por estes motivos, três métodos da última classe citada foram usados ao longo deste trabalho.

Um aspecto comum a todos estes métodos computacionais é o tempo considerável necessário para o desenvolvimento de programas de computador que implementem formulações para casos gerais ou complexos. Muito desse tempo é consumido codificando tarefas que são comuns a praticamente qualquer software de simulação numérica, como, por exemplo, a montagem das matrizes, manipulação de dados de entrada e saída (I/O) ou a solução iterativa de sistemas de equações lineares. A abordagem adotada neste trabalho para superar esta dificuldade foi utilizar a ferramenta de código aberto *FEniCS/DOLFIN* (Logg e Wells, 2010), a qual permite efetuar a geração automática de código de baixo nível (em *C++*) baseada em informações fornecidas através de uma interface com um programa escrito em código de alto nível (*Python* ou *C++*), permitindo ao programador se concentrar no desenvolvimento e teste de diferentes formulações matemáticas e numéricas para o problema de interesse usando uma sintaxe muito similar àquela encontrada na descrição matemática do problema, ao invés de despendar tempo escrevendo códigos auxiliares para tarefas administrativas.

Além disso, existe uma necessidade de reduzir o tempo de CPU de um programa, de modo a viabilizar o uso de simulações dentro do prazo de um projeto. Normalmente, a parte que mais consome tempo de processamento em uma simulação é a solução numérica dos sistemas de equações lineares resultantes da discretização das EDPs (Saad, 2003). Várias técnicas estão disponíveis para resolver de modo iterativo estes tipos de sistemas, cada uma delas com suas próprias vantagens e desvantagens em termos de velocidade e generalidade, sendo que estas características frequentemente apontam para direções distintas, isto é, um método extremamente rápido para uma classe de problemas muitas vezes não é geral e vice-versa. Neste trabalho foi utilizado um método conhecido por acoplar de modo quase ideal estas duas características: o método Multigrid (Trottenberg *et al.*, 2001; Saad, 2003; Briggs *et al.*, 2000). Todavia, esta particularidade não é obtida sem custo, neste caso, as principais dificuldades adicionais associadas aos métodos Multigrid são: a necessidade de uma sequência de malhas sucessivamente menos refinadas,

cujas geração pode se tornar um problema caso não seja abordada adequadamente e o uso de operadores de transferência de informações entre níveis consecutivos. Seguindo o princípio de economizar tempo de codificação com tarefas gerais e manter o foco no problema de interesse, neste trabalho foi utilizada a biblioteca de código aberto *PyAMG* (Bell *et al.*, 2008), a qual não apenas efetua de modo completamente automático a geração de uma sequência de matrizes, dada uma malha inicial, como também permite o uso de diferentes variantes do Método do Multigrid Algébrico (AMG), incluindo os algoritmos clássicos de Ruge-Stüben e o de Agregação Suavizada (*Smooth Aggregation*) (Trottenberg *et al.*, 2001).

1.2 Objetivos

Dada a relevância do problema proposto, como mencionado na seção 1.1, o objetivo principal deste trabalho é o desenvolvimento de um sistema computacional para a simulação de escoamentos bifásicos óleo-água em meios porosos, considerando a flexibilidade de lidar com geometrias bidimensionais e tridimensionais, domínios homogêneos e heterogêneos e tensores de permeabilidades isotrópicos e anisotrópicos. Além disso, como objetivos secundários temos o uso de técnicas de alto desempenho (Método do Multigrid Algébrico (AMG)) para diminuição do tempo necessário de processamento para as análises e a utilização de pacotes computacionais que permitem a automação de parte do processo de desenvolvimento do software.

Na literatura existem diversas opções disponíveis de métodos matemáticos, numéricos e computacionais para resolução dos problemas inerentes às metas pretendidas neste trabalho. Em relação à descrição matemática do problema, optou-se pela utilização da formulação clássica que considera a lei de Darcy para modelar o fenômeno de interesse. Esta formulação é razoável para uma certa classe de aplicações de interesse, porém possui limitações bastante discutidas na literatura já que a mesma foi originalmente desenvolvida para um contexto diferente (escoamento monofásico laminar em um tubo preenchido com areia). No capítulo 2 serão discutidas as hipóteses que devem ser consideradas para manter a validade da formulação utilizada. Em todo caso, o modelo de Darcy é utilizado largamente na indústria e academia (Peaceman e Nash, 1990; Aziz e Settari, 1979; Marle, 1981; Ewing, 1983), tendo se mostrado adequado para o objetivo ao qual este trabalho se destina, desde que respeitadas as limitações impostas pelo mesmo.

Analisando as alternativas disponíveis em termos de métodos numéricos, optou-se pelo uso do MEF pela sua combinação de sólida base teórica disponível na literatura (Cha-

vent e Jaffre, 1986; Chen *et al.*, 2006; Hughes, 2000) e grande flexibilidade para lidar com casos que apresentem geometrias ou condições de contorno complexas. Como objetivos específicos dentro deste quesito, temos a discretização das EDPs usando a formulação de Galerkin para a equação de pressão (Luna *et al.*, 2011) e o método SUPG para a equação de saturação, estando as duas variáveis acopladas de modo sequencial implícito via velocidade total. Também foi testada a discretização via Método dos Elementos Finitos Mistos (MEFM) para a pressão e velocidade simultaneamente, calculando a saturação posteriormente usando também o método SUPG. Em relação à resolução do sistema de equações provenientes de tal discretização, foi definida como meta específica o uso de alguma técnica de aceleração de convergência que permitisse uma grande escalabilidade de modo a ser possível a simulação de problemas de grande porte. Para este fim, foi escolhido, dentre as opções disponíveis, o uso do método AMG devido a sua robustez e eficiência.

Por fim, foi estabelecida a meta de elaborar um sistema computacional de modo prático e utilizando técnicas de geração de código automática, tendo assim um programa mais fácil de ser gerenciado e expandido conforme surjam novas necessidades. No capítulo 4 serão discutidas as opções disponíveis na literatura e o porquê da escolha, dentre elas, da ferramenta *FEniCS/DOLFIN*.

1.3 Organização da Dissertação

Esta dissertação está dividida em seis capítulos, incluindo introdução e conclusão, organizados de modo a apresentar sequencialmente os passos necessários para se alcançar os objetivos definidos na seção 1.2. Além disso, um apêndice que lida com aspectos práticos da implementação e uso do código desenvolvido foi anexado ao final do trabalho.

- *Capítulo 1: Introdução.* O presente capítulo contém a motivação para a realização deste trabalho, um curto histórico das principais áreas relacionadas com uma revisão bibliográfica de alguns dos principais trabalhos de outros autores e suas contribuições, e por fim uma apresentação clara dos objetivos do trabalho e a descrição da organização desta dissertação.
- *Capítulo 2: Formulação Matemática.* No segundo capítulo são apresentadas as equações governantes do problema que se deseja modelar. As propriedades de rocha, fluido e de interação rocha-fluido são descritas e a notação utilizada ao longo do trabalho é definida. Além disso, são discutidas as principais hipóteses

simplificadoras utilizadas.

- *Capítulo 3: Formulação Numérica.* Lida com a transformação de um problema contínuo definido pelas equações apresentadas no capítulo 2 em um problema discreto que pode ser resolvido de modo aproximado utilizando métodos numéricos, os quais também são discutidos em detalhes.
- *Capítulo 4: Implementação Computacional.* Neste capítulo são abordados os aspectos práticos relativos à implementação de todo o arcabouço teórico desenvolvido. Os principais programas e bibliotecas utilizados são apresentados e as razões para a escolha de cada um é discutida. Por fim, um algoritmo em diagramas de blocos da estrutura geral do programa é exposto, de onde se pode observar as interligações entre os diferentes pacotes adotados e os pontos de interface com o código escrito especificamente para este trabalho.
- *Capítulo 5: Resultados.* A flexibilidade e acurácia do programa desenvolvido é comprovada através do uso de diversos exemplos que variam desde casos simples homogêneos e isotrópicos com solução analítica, até casos mais complexos considerando heterogeneidades no domínio e o uso de tensores de permeabilidade anisotrópicos. Sempre que possível, as soluções obtidas foram comparadas com *benchmarks* da literatura ou com soluções analíticas.
- *Capítulo 6: Conclusões e Trabalho Futuros.* Neste último capítulo são feitas as observações principais a respeito dos objetivos atingidos neste trabalho quando comparado com as metas estabelecidas. Finalmente, são apresentadas algumas possibilidades de trabalhos futuros baseados no que foi discutido ao longo desta dissertação.
- *Apêndice 1: Delfine - Manual do Usuário.* Este apêndice tem um enfoque bastante prático e seu objetivo é servir como um tutorial para o leitor interessado em utilizar as ferramentas desenvolvidas neste trabalho. Todos os passos, desde a obtenção do código através da internet até o uso do mesmo, passando pela instalação e configuração de pacotes adicionais, são discutidos com base em um exemplo simples usado como orientação.

Capítulo 2

Formulação Matemática

Neste trabalho, foi adotada a formulação matemática clássica proposta em Peaceman (1977) para o escoamento simultâneo de duas fases imiscíveis em um meio poroso saturado. Esta abordagem vem sendo utilizada por muitos pesquisadores (Ewing, 1983; Chavent e Jaffre, 1986; Carvalho, 2005; Silva, 2008) e tem com uma de suas principais características a manipulação da equação de conservação da massa usando a lei de Darcy de modo a formar um sistema com uma equação de pressão parabólica-elíptica e uma equação de saturação parabólica-hiperbólica (Carvalho, 2005), em oposição a outras formulações nas quais os campos de pressão e saturação são resolvidos simultaneamente em um sistema de EDPs parabólicas (Aziz e Settari, 1979).

A formulação segregada de Peaceman permite o uso de métodos especializados capazes de explorar as particularidades matemáticas de cada equação do sistema resultante. O acoplamento entre estes dois campos é obtido através do uso de um termo de velocidade total.

De modo a utilizar as equações apresentadas ao longo deste capítulo, é necessária a adoção de algumas hipóteses simplificadoras (Carvalho, 2005; Peaceman, 1977):

- Meio poroso saturado;
- Rochas e fluidos incompressíveis;
- Fluidos imiscíveis;
- Escoamento isotérmico;
- Lei de Darcy válida para as velocidades consideradas.

Para o correto entendimento da formulação adotada é necessário introduzir alguns conceitos, os quais serão discutidos na seção seguinte, usados para a descrição macroscópica das propriedades de rochas, fluidos e da interação entre os mesmos.

2.1 Propriedades

Antes de iniciar a descrição das diversas propriedades necessárias para a elaboração do modelo matemático utilizado, é importante fazer um comentário a respeito do modo como essas propriedades se apresentam na realidade e como são de fato representadas na simulação do fenômeno no domínio de interesse.

Para uma perfeita descrição do escoamento de fluidos em meios porosos, seria necessário adotar uma abordagem microscópica, a qual necessitaria do conhecimento das características geométricas de cada um dos poros para a definição das fronteiras e das condições de contorno. De posse dessas informações, seria possível utilizar uma formulação matemática, como as equações de Navier-Stokes (Fortuna, 2000), acrescida de modelos geomecânicos e geoquímicos para simular o escoamento de cada um dos fluidos existentes no meio e a interação entre eles. Desnecessário comentar que tal alternativa é impraticável tanto do ponto de vista computacional, já que seria possível analisar domínios apenas em escalas cuja ordem de grandeza esteja na escala dos poros, quanto do ponto de vista da aquisição de dados de entrada, pois é praticamente impossível se obter toda a descrição morfológica de um meio como um reservatório de petróleo, do qual se obtém normalmente apenas algumas amostras, chamadas de testemunhos, os quais são retiradas de pontos que estão várias centenas de metros ou até quilômetros de distância entre si (Thomas, 2001).

Sendo assim, para todos os efeitos práticos, na análise numérica de reservatórios de petróleo se utiliza uma abordagem macroscópica, onde as propriedades são normalmente consideradas variáveis contínuas definidas em todo o domínio ocupado pelo meio poroso. Tais valores das grandezas físicas representam para cada ponto específico uma média volumétrica dos valores da região circunvizinha a este ponto. Ou seja, não é mais necessário uma descrição da morfologia exata do reservatório, já que ao invés disso serão utilizadas propriedades não mensuráveis no nível microscópico, mas que representam no nível macroscópico o efeito equivalente ao da estrutura do meio poroso (Hurtado, 2005).

Entretanto, é válido comentar que esta descrição macroscópica do meio poroso apresenta também dificuldades. As formações rochosas que formam os reservatórios de petróleo são frequentemente descritas através de modelos geológicos e geofísicos

na escala de metros ou até menor (Aarnes *et al.*, 2007). Tal escala tornaria em muitos casos inviável a simulação de escoamentos de fluidos em tais reservatórios, já que eles têm ordem de grandeza de quilômetros, o que implicaria, para casos tridimensionais, em malhas com vários milhões de elementos, passíveis de serem analisadas apenas em *clusters* de grande porte.

Apesar do grande aumento na capacidade computacional do *hardware* disponível e também do uso de técnicas avançadas para extrair o máximo de desempenho de tais computadores, os modelos de reservatório a serem analisados na prática frequentemente têm que contar com o uso de técnicas de *Upscaling* (Menezes, 2009), ou seja, uma restrição dos parâmetros geofísicos obtidos em modelos geológicos para malhas tipicamente dezenas ou centenas de vezes menos refinadas que serão então analisadas via algum método numérico (Aarnes *et al.*, 2007).

2.1.1 Porosidade

Uma rocha, para ter a capacidade de armazenar um fluido, por exemplo petróleo, deve possuir vazios no seu interior, os quais são chamados de poros. Além disso, para que haja fluxo dentro desta rocha-reservatório é necessário que tais poros estejam interconectados para que existam caminhos que os fluidos possam percorrer. Para efeitos de cálculo de porosidade, existe uma diferença entre a porosidade *absoluta*, a qual abrange todos os poros, estejam eles interconectados ou não, e a porosidade *efetiva*, onde apenas os poros ligados entre si são considerados (Thomas, 2001; Hyne, 2001). Apenas esse segundo tipo de porosidade será usado neste trabalho, já que é o de maior interesse para a análise de escoamentos. Portanto, sempre que houver referência à porosidade, pode-se entender como a variante efetiva da mesma.

A porosidade de uma rocha-reservatório é definida matematicamente como a razão entre o volume dos espaços vazios existentes na rocha e o volume total da amostra (Thomas, 2001). Logo, podemos descrever a porosidade ϕ como:

$$\phi = \frac{V_p}{V_t} \quad (2.1)$$

onde o volume total é $V_t = V_p + V_s$, com V_p o volume poroso e V_s o volume da matriz sólida. Segundo a abordagem macroscópica adotada, a porosidade será uma função contínua no espaço. Logo, se está assumindo que o valor calculado pela Eq. (2.1) representará a média de um volume representativo com dimensão pequena quando comparada ao meio poroso como um todo, porém grande em comparação com as dimensões características

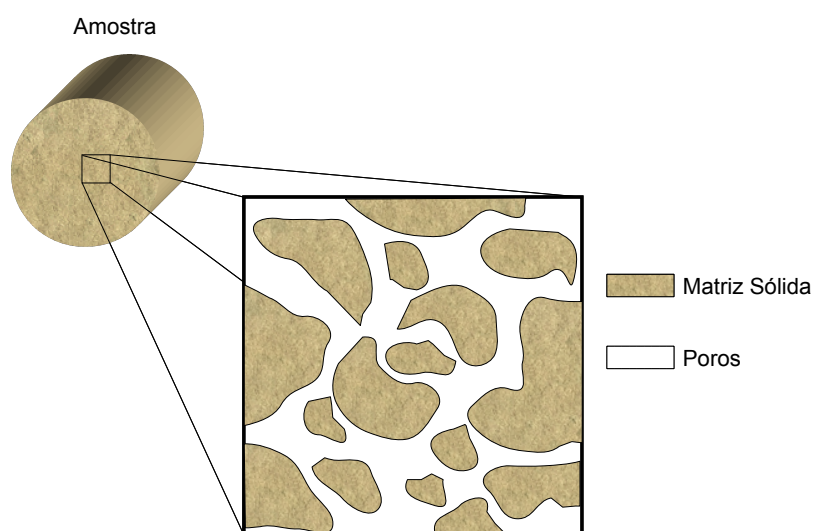


Figura 2.1 Exemplo de meio poroso com destaque para os volumes de matriz sólida (rocha) e de poros (vazio).

dos poros.

A Fig. 2.1 apresenta esquematicamente uma amostra de meio poroso, com uma aproximação em uma parte dela onde se pode observar claramente os conceitos de volume poroso e de matriz sólida representados. A caracterização das fases em um escoamento bifásico é vista na Fig. 2.2.

2.1.2 Permeabilidade Absoluta

A permeabilidade absoluta é uma medida da capacidade de uma única fase escoar em um meio poroso sob certas condições. Esta é uma propriedade mais difícil de ser medida do que a porosidade, porém mais importante no que tange ao escoamento de fluidos. Não necessariamente irá existir uma proporcionalidade direta entre estes dois parâmetros, já que para a permeabilidade importa não apenas a quantidade de poros interconectados, mas também o modo como eles estão conectados e a tortuosidade do caminho formado por eles. Existe também a possibilidade de se ter rochas com baixíssimas porosidades, como muitas vezes é o caso de algumas rochas carbonáticas, que porém apresentam uma alta permeabilidade devido à presença de fraturas ou falhas (Hyne, 2001). Todavia, normalmente são geradas correlações entre as duas propriedades, como pode ser visto na

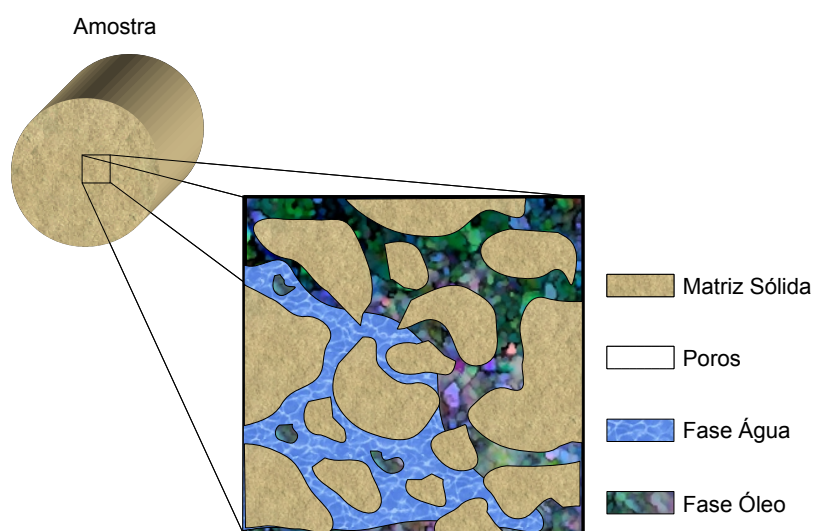


Figura 2.2 Exemplo de meio poroso saturado com escoamento bifásico.

Fig. 2.3 (Schneider, 2003).

Normalmente, a permeabilidade é representada matematicamente através de um tensor de 2ª ordem, denominado de $\mathbf{K}$ ao longo deste trabalho. Isto significa que não apenas os valores da permeabilidade absoluta variam de acordo com a orientação da rocha, como também que, devido aos termos cruzados do tensor, os valores em uma direção influenciam a permeabilidade nas outras direções. Neste caso o tensor é chamado de anisotrópico. Se $\mathbf{K}$ puder ser representado por uma função escalar, ou seja, a permeabilidade em todas as direções for a mesma, este tensor degenerado em um escalar é chamado de isotrópico. Além disso, devido à grande heterogeneidade entre as diferentes formações rochosas, a permeabilidade pode variar espacialmente em ordens de grandeza, sendo que variações locais no intervalo entre 1mD e 10D não são incomuns (Aarnes *et al.*, 2007).

O sistema elaborado neste trabalho pode lidar de modo efetivo tanto com tensores anisotrópicos quanto com variações da permeabilidade ao longo do domínio (heterogeneidade).

2.1.3 Permeabilidade Relativa

Apesar de termos assumido total imiscibilidade entre as fases, consideramos para efeitos de simulação, segundo a abordagem macroscópica adotada neste trabalho, a presença de

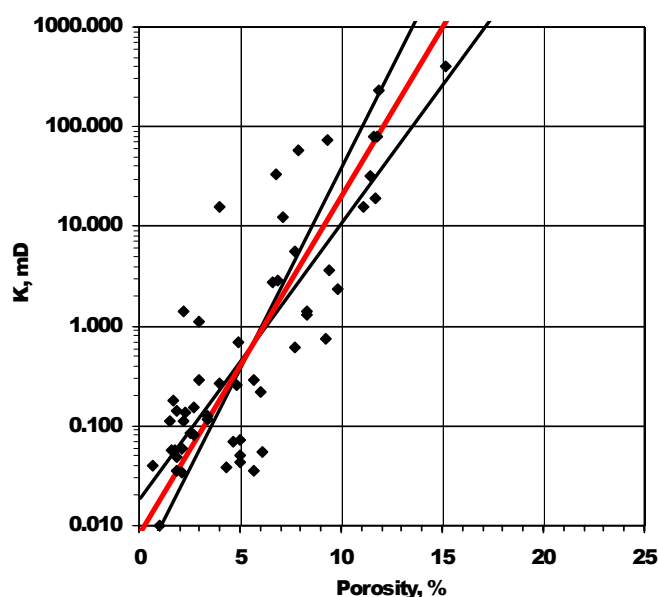


Figura 2.3 Exemplo de correlação entre porosidade e permeabilidade absoluta (retirado de Schneider (2003)).

todas as fases simultaneamente no mesmo ponto, já que, como mencionado anteriormente na seção 2.1, o comportamento do escoamento na escala dos poros não é de interesse prático para a análise de reservatórios. Isto significa que todas as fases terão influência umas sobre as outras.

Portanto, a permeabilidade de fato percebida por um fluido em um meio poroso será reduzida pela presença das outras fases, resultando, portanto, na necessidade da definição de um termo de permeabilidade relativa da fase α , representado neste trabalho por $k_{r\alpha}$, que irá descrever como uma fase α escoar na presença simultânea de outras. As permeabilidades absoluta K , efetiva K_α e relativa $k_{r\alpha}$ estão relacionadas através da seguinte expressão:

$$K_\alpha = k_{r\alpha} K \quad (2.2)$$

Dada a complexidade envolvida na interação entre a rocha e as diferentes fases na escala dos poros durante o escoamento, os valores para a permeabilidade relativa em função das propriedades dos fluidos e dos meios porosos são difíceis de serem obtidos. De um modo geral, depende-se de modelos semi-empíricos para uso em simulações de reservatórios de petróleo.

Um exemplo típico de curva de permeabilidade relativa tanto da fase água quanto da fase óleo em função da saturação da água normalizada (S_{wn}) é obtido usando o modelo

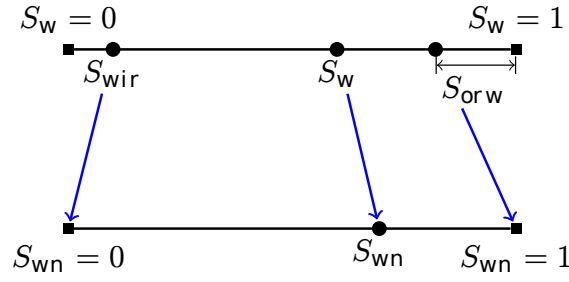


Figura 2.4 Saturação normalizada.

de Corey (Chen *et al.*, 2006; Carvalho, 2005; Silva, 2008), o qual foi utilizado neste trabalho. A saturação normalizada pode ser definida utilizando os valores das saturações irreduzíveis de água (S_{wir}) e residual de óleo (S_{orw}). S_{wir} representa a menor saturação possível para a fase água em um processo mecânico, já S_{orw} representa a menor saturação da fase óleo depois do varrido pela fase água. Uma representação gráfica da saturação normalizada pode ser vista na Fig. 2.4 e a expressão para o cálculo da mesma é mostrada a seguir:

$$S_{wn}(S_w) = \frac{S_w - S_{wir}}{1 - S_{wir} - S_{orw}} \quad (2.3)$$

Uma vez obtido o valor para a saturação normalizada, as permeabilidades relativas para as fases são calculadas de acordo com a expressão a seguir. Um exemplo do uso do modelo de Corey para o cálculo das permeabilidades relativas é apresentado na Fig. 2.5, onde k_{rw}^o representa o valor máximo da permeabilidade relativa da água.

$$k_{rw}(S_{wn}) = k_{rw}^o S_{wn}^{n_w} \quad (2.4a)$$

$$k_{row}(S_{wn}) = (1 - S_{wn})^{n_w} \quad (2.4b)$$

Neste modelo, k_{rw} , k_{row} e n_w representam a permeabilidade relativa da água, a permeabilidade relativa do óleo e um expoente de forma da curva, respectivamente. O expoente n_w é usualmente determinado através do método dos mínimos quadrados para ajuste da curva à resultados experimentais. Como exemplo, podemos citar o caso do deslocamento de óleo por gás em rochas homogêneas, para o qual o valor de $n_w = 4.0$ se apresenta adequado. Porém, dependendo do grau de consolidação da rocha, outros valores podem resultar em melhores ajustes aos dados empíricos (Ahmed, 2006).

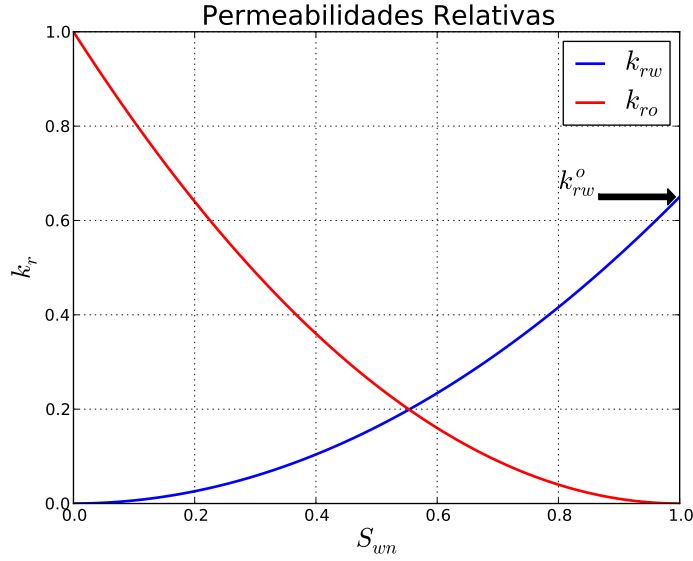


Figura 2.5 Exemplo de modelo de permeabilidade relativa para escoamentos bifásicos em meios porosos.

2.2 Equação da Conservação de Massa

Considerando a hipótese de meio poroso saturado e a existência de n fases, a equação constitutiva para a saturação pode ser descrita por:

$$\sum_{\alpha=1}^n S_{\alpha} = 1 \quad (2.5)$$

onde α representa cada fase (neste trabalho, $\alpha = w$ e o para água e óleo, respectivamente) e S_{α} representa a saturação da fase α .

A lei de Darcy generalizada para a velocidade de cada fase é expressa através da seguinte equação:

$$\mathbf{v}_{\alpha} = -\frac{K_{\alpha}}{\mu_{\alpha}} (\nabla p_{\alpha} - \rho_{\alpha} g \nabla z) \quad (2.6)$$

com K_{α} , μ_{α} , p_{α} e ρ_{α} representando a permeabilidade efetiva, viscosidade, pressão e densidade da fase α , ao passo que g e z representam a aceleração da gravidade e o deslocamento na sua direção, respectivamente.

A equação de conservação para cada fase é descrita como em Peaceman (1977):

$$-\nabla \cdot (\rho_{\alpha} \mathbf{v}_{\alpha}) + q_{\alpha} = \frac{\partial (\phi \rho_{\alpha} S_{\alpha})}{\partial t} \quad (2.7)$$

Os termos q_α , ϕ e t representam termos fontes/sumidores da fase α (por exemplo, poços), porosidade efetiva da rocha e o tempo, respectivamente. Além disso, se forem consideradas duas fases, uma molhante (aquosa) e uma não-molhante (oleosa), combinando as Eqs. (2.6) e (2.7), e usando a definição dada na Eq. (2.2), é obtido o seguinte sistema de equações diferenciais parciais que resolve o problema do escoamento bifásico dadas as hipóteses já mencionadas:

$$\nabla \cdot \left(\frac{\rho_w \mathbf{K} k_{rw}}{\mu_w} \nabla (p_w - \rho_w g \nabla z) \right) + q_w = \frac{\partial (\phi \rho_w S_w)}{\partial t} \quad (2.8a)$$

$$\nabla \cdot \left(\frac{\rho_o \mathbf{K} k_{ro}}{\mu_o} \nabla (p_o - \rho_o g \nabla z) \right) + q_o = \frac{\partial (\phi \rho_o S_o)}{\partial t} \quad (2.8b)$$

Nas Eqs. (2.8a) e (2.8b), os campos de pressão e saturação são acoplados em ambas as equações. Estas possuem uma descrição matemática semelhante à da equação de condução de calor e por isso espera-se que tenham um comportamento essencialmente parabólico. Esta assertiva não é necessariamente verdadeira e pode ser avaliada através da obtenção de um par de equações que seja dependente ou da pressão ou da saturação. A dedução de tais equações é apresentada na seção seguinte.

2.3 Equação da Pressão

A abordagem utilizada para se obter a equação da pressão é eliminar a derivada temporal da saturação apresentada na Eq. (2.7). Primeiramente, as derivadas temporais são expandidas para se obter:

$$-\nabla \cdot (\rho_w \mathbf{v}_w) + q_w = \rho_w S_w \frac{\partial \phi}{\partial t} + \phi S_w \frac{d\rho_w}{dp_w} \frac{\partial p_w}{\partial t} + \phi \rho_w \frac{\partial S_w}{\partial t} \quad (2.9a)$$

$$-\nabla \cdot (\rho_o \mathbf{v}_o) + q_o = \rho_o S_o \frac{\partial \phi}{\partial t} + \phi S_o \frac{d\rho_o}{dp_o} \frac{\partial p_o}{\partial t} + \phi \rho_o \frac{\partial S_o}{\partial t} \quad (2.9b)$$

Então, dividindo a primeira equação por ρ_w , a segunda por ρ_o e levando ainda em consideração as hipóteses feitas anteriormente de que fluidos e rochas são incompressíveis e finalmente adicionando as equações resultantes, obtêm-se:

$$-\nabla \cdot \mathbf{v}_t + Q_t = \phi \frac{\partial (S_w + S_o)}{\partial t} \quad (2.10)$$

onde $\mathbf{v}_t = \mathbf{v}_w + \mathbf{v}_o$ é denominada de velocidade total e $Q_t = (q_w/\rho_w) + (q_o/\rho_o)$ é a taxa

volumétrica total. Além disso, considerando a Eq. (2.5) e rearranjando os termos, nós obtemos a equação de pressão para o escoamento bifásico em meios porosos:

$$\nabla \cdot \mathbf{v}_t = Q_t \quad (2.11)$$

De modo a apresentar a Eq. (2.11) em relação a apenas uma única variável de pressão, pode-se definir uma pressão média por:

$$p_{avg} = \frac{p_w + p_o}{2} \quad (2.12)$$

Considerando a definição de pressão capilar como $p_c = p_o - p_w$, pode-se expressar as pressões das fases individuais como:

$$p_w = p_{avg} - \frac{p_c}{2} \quad (2.13a)$$

$$p_o = p_{avg} + \frac{p_c}{2} \quad (2.13b)$$

As mobilidades das fases são definidas como a relação entre a permeabilidade relativa e a viscosidade do fluido:

$$\lambda_\alpha = \frac{k_{r\alpha}}{\mu_\alpha} \quad (2.14)$$

Finalmente, reescrevendo a Eq. (2.11) utilizando pressões médias e capilares pode-se obter, depois de um rearranjo dos termos, a seguinte expressão:

$$\nabla \cdot \left(-\mathbf{K} \left((\lambda_w + \lambda_o) \nabla p_{avg} + \frac{\lambda_w - \lambda_o}{2} \nabla p_c - (\lambda_w \rho_w + \lambda_o \rho_o) g \nabla z \right) \right) = Q_t \quad (2.15)$$

Logo, pode ser observado que a Eq. (2.15), considerando as hipóteses feitas, tem uma natureza elíptica. Outrossim, foi considerada neste trabalho a hipótese de fluxo horizontal (sem influência da gravidade) e com pressão capilar negligenciável. Deste modo foi possível se concentrar nas características elípticas da equação de pressão. Definindo-se a mobilidade total como $\lambda_t = \lambda_w + \lambda_o$, a forma simplificada da equação de pressão pode ser apresentada como:

$$-\nabla \cdot (\mathbf{K} \lambda_t \nabla p_{avg}) = Q_t \quad (2.16)$$

2.4 Equação da Saturação

A equação de saturação pode ser deduzida usando uma manipulação algébrica similar à utilizada na seção anterior de modo a completar o modelo de pressão-saturação para um escoamento bifásico água-óleo em meios porosos. Para a dedução desta equação faz-se necessário definir qual das fases (molhante ou não-molhante) servirá de referência. Neste trabalho foi escolhida a fase aquosa seguindo a prática usual da literatura (Peaceman, 1977; Ewing, 1983; Carvalho, 2005; Silva, 2008).

Consideraremos a princípio que tenha sido obtida uma solução para um caso mais geral representado pela Eq. (2.15), sendo portanto conhecida p_{avg} . Logo, as pressões das fases água (p_w) e óleo (p_o) podem ser calculadas a partir da Eq. (2.13) e utilizando a Eq. (2.6) podemos calcular as velocidades das duas fases, obtendo:

$$\mathbf{v}_w = -K\lambda_w(\nabla p_w - \rho_w g \nabla z) \quad (2.17a)$$

$$\mathbf{v}_o = -K\lambda_o(\nabla p_o - \rho_o g \nabla z) \quad (2.17b)$$

Multiplicando a Eq. (2.17a) por λ_o , a Eq. (2.17b) por λ_w , subtraindo uma da outra e utilizando a definição de pressão capilar, pode-se obter:

$$-\lambda_w \mathbf{v}_o + \lambda_o \mathbf{v}_w = K\lambda_o \lambda_w \nabla p_c - K\lambda_o \lambda_w (\rho_o - \rho_w) g \nabla z \quad (2.18)$$

Considerando-se ainda as definições de velocidade e mobilidade total, podemos reescrever a equação anterior como:

$$\lambda_t \mathbf{v}_w = \lambda_w \mathbf{v}_t + K\lambda_o \lambda_w (\nabla p_c + (\rho_w - \rho_o) g \nabla z) \quad (2.19)$$

Definindo ainda o fluxo fracional da fase α como:

$$f_\alpha = \frac{\lambda_\alpha}{\lambda_t} \quad (2.20)$$

e utilizando-se a seguinte função da saturação para simplificar a notação:

$$h_w = -\frac{\lambda_o \lambda_w}{\lambda_t} \frac{dp_c}{dS_w} \quad (2.21)$$

podemos obter uma expressão na qual a velocidade da fase água é expressa em função da velocidade total. Substituindo as Eqs. (2.20) e (2.21) na Eq. (2.19) e inserindo esta na

Eq. (2.7), podemos escrever a seguinte expressão para a saturação da fase água:

$$\frac{\partial(\phi \rho_w S_w)}{\partial t} = -\nabla \cdot (\rho_w (f_w \mathbf{v}_t - \mathbf{K} h_w \nabla S_w + \mathbf{K} \lambda_o f_w (\rho_w - \rho_o) g \nabla z)) + q_w \quad (2.22)$$

Assumindo as hipóteses já mencionadas de rochas e fluidos incompressíveis podemos representar a expressão anterior por:

$$\phi \rho_w \frac{\partial S_w}{\partial t} = -\rho_w \nabla \cdot ((f_w \mathbf{v}_t - \mathbf{K} h_w \nabla S_w + \mathbf{K} \lambda_o f_w (\rho_w - \rho_o) g \nabla z)) + q_w \quad (2.23)$$

Além disso, dividindo a Eq. 2.23 por ρ_w e considerando a definição do termo de taxa volumétrica de injeção/produção da fase água $Q_w = q_w / \rho_w$, obtemos:

$$\phi \frac{\partial S_w}{\partial t} = -\nabla \cdot ((f_w \mathbf{v}_t - \mathbf{K} h_w \nabla S_w + \mathbf{K} \lambda_o f_w (\rho_w - \rho_o) g \nabla z)) + Q_w \quad (2.24)$$

A Eq. (2.24) permite obter a saturação da fase água em um meio poroso considerando as hipóteses básicas descritas no início deste capítulo. Esta equação apresenta um comportamento essencialmente parabólico-hiperbólico, a não ser que o termo de capilaridade seja desprezível comparado com os termos advectivos, o que é verdade especialmente para casos como na vizinhança de poços ou em áreas do reservatório onde a velocidade total seja alta (Carvalho, 2005). Considerando esta última condição como válida e supondo um escoamento horizontal, obtemos a forma simplificada para a saturação da fase água em um meio poroso:

$$\phi \frac{\partial S_w}{\partial t} = -\nabla \cdot (f_w \mathbf{v}_t) + Q_w \quad (2.25)$$

A Eq. (2.25) apresenta características essencialmente hiperbólicas e não-lineares devido ao termo de fluxo fracional. Um modo de estudar os fenômenos principais associados a esta equação é utilizar modelos que isolem apenas algumas das características de interesse. Sendo assim, serão apresentadas a seguir duas variações das equações apresentadas para a saturação. A primeira delas considera um escoamento unidimensional e sem termo de fonte, onde o principal objetivo é analisar o efeito da não-linearidade decorrente do fluxo fracional. A segunda variação considera uma versão linear da Eq. (2.24), a qual nada mais é do que a conhecida equação de difusão-convecção e que neste caso governa o escoamento multidimensional de fluidos miscíveis (Peaceman, 1977; Barbosa *et al.*, 2009; Loula *et al.*, 1995).

2.4.1 Equação de Buckley-Leverett

Buckley e Leverett (1942) propuseram um modelo para representar o deslocamento imiscível de óleo por água em meios porosos rígidos unidimensionais, considerando os termos de gravidade e capilaridade desprezíveis. Partindo da Eq. (2.25), eliminamos o termo de fonte, já que neste modelo injeção e produção são manuseadas nas condições de contorno. Assim, para o caso unidimensional podemos obter:

$$\phi \frac{\partial S_w}{\partial t} + \frac{\partial (f_w v_t)}{\partial x} = 0 \quad (2.26)$$

Expandindo o segundo termo da Eq. (2.26) utilizando a regra de derivada do produto, chegamos a:

$$\phi \frac{\partial S_w}{\partial t} + v_t \frac{\partial f_w}{\partial x} + f_w \frac{\partial v_t}{\partial x} = 0 \quad (2.27)$$

Sendo que a derivada parcial do terceiro termo desta equação necessariamente tem que ser igual a zero se consideramos a imposição de continuidade dada pela Eq. (2.11) e utilizando um termo de fonte nulo. Além disso, podemos expandir o segundo termo usando a regra da cadeia, obtendo assim a equação de Buckley-Leverett:

$$\phi \frac{\partial S_w}{\partial t} + v_t \frac{df_w}{dS_w} \frac{\partial S_w}{\partial x} = 0 \quad (2.28)$$

Esta é uma equação hiperbólica de transporte não-linear, pois o coeficiente que multiplica o termo $\partial S_w / \partial x$ é uma função não-linear de S_w (Carvalho, 2005). A análise do comportamento desta função permite inferir um série de propriedades comuns aos escoamentos mais complexos. Um exemplo de representação da função de fluxo fracional e de sua derivada em relação à saturação pode ser observado na Fig. 2.6. No Capítulo 5 serão apresentados e discutidos resultados para esta equação modelo.

2.4.2 Equação do Deslocamento de Fluidos Miscíveis

A análise das equações de escoamentos miscíveis em meios porosos permite analisar o comportamento de diversos fenômenos de interesse como, por exemplo, o transporte de contaminantes em aquíferos (Carvalho, 2005), a injeção de traçadores em reservatórios de petróleo para recuperação avançada (Loula *et al.*, 1995), entre outros (Bear *et al.*, 1992; Barbosa *et al.*, 2009). Na área de simulação de reservatórios tais equações têm sido tradicionalmente utilizadas para testar e desenvolver formulações numéricas devido às diversas similaridades existentes quando comparadas às equações que descrevem os

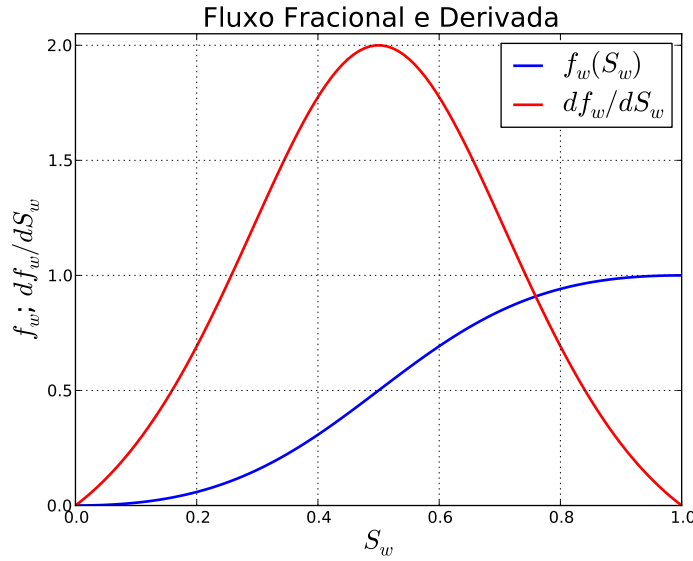


Figura 2.6 Exemplo de função de fluxo fracional e sua derivada relativa à saturação da fase água.

escoamentos imiscíveis (Ewing, 1983).

A equação para a concentração C do soluto (traçador) dissolvido em um fluido solvente é dada por:

$$\phi \frac{\partial C}{\partial t} = -\nabla \cdot (\mathbf{v}C - \mathbf{D}\nabla C) + \hat{C}Q \quad (2.29)$$

onde $\mathbf{D}$ representa o tensor de dispersão-difusividade, Q é o termo fonte/sumidouro nos poços e $\hat{C}$ é a concentração neles.

O tensor $\mathbf{D}$ combina os efeitos de dispersão mecânica e difusão molecular, onde o primeiro é normalmente proporcional à velocidade e possui componentes tanto na direção longitudinal quanto transversal do escoamento. Já o termo de difusão pode ser calculado pela Lei de Fick, a qual permite obter um fluxo difusivo a partir do gradiente de concentração do soluto C . Uma excelente discussão a respeito dos detalhes sobre este tensor e a interpretação física de seus componentes pode ser encontrada em Ewing (1983).

No Capítulo 3 a versão unidimensional em meio homogêneo da Eq. (2.29) será utilizada para a apresentação de alguns dos métodos numéricos utilizados neste trabalho.

2.5 Condições Iniciais e de Contorno

Antes de especificar a formulação numérica utilizada, algumas definições a respeito das condições iniciais, de contorno e hipóteses simplificadoras têm que ser discutidas para

2.5. CONDIÇÕES INICIAIS E DE CONTORNO

tornar o problema abordado completamente determinado.

Os contornos são descritos como $\Gamma = \partial\Omega = \Gamma_I \cup \Gamma_P \cup \Gamma_D \cup \Gamma_N$, onde (Carvalho, 2005):

- Γ_I = Poços injetores;
- Γ_P = Poços produtores;
- Γ_D = Condição de contorno de Dirichlet (Variável prescrita);
- Γ_N = Condição de contorno de Neumann (Fluxo prescrito).

Usualmente, os poços são tratados como condições de contorno internas, sendo modelados através de métodos especiais para lidar com a velocidade alta nas adjacências de um poço (Peaceman, 1977) comparada com o resto do domínio. Apesar disso, foi adotada neste trabalho uma hipótese simplificadora considerando que os poços são termos fonte/sumidouros de produção ou injeção concentrados em um único nó da malha, isto é, uma função delta de Dirac neste ponto.

Neste trabalho, foi feita a consideração usual adotada em problemas de reservatório de petróleo (Peaceman, 1977; Ewing, 1983) de condição de fluxo zero em todas as fronteiras exteriores do domínio, isto é:

$$K\lambda p \cdot \mathbf{n} = 0 \quad \text{em } \Gamma_N \quad (2.30)$$

onde λ é a mobilidade total, ou seja, $\lambda = \lambda_w + \lambda_o$, p é a pressão média ($p = p_{avg}$) e $\mathbf{n}$ é o vetor unitário normal aos contornos externos.

Logo, assumindo que a transferência de massa entre o reservatório e o meio externo se dará apenas pelos poços injetores e produtores, temos:

$$\begin{aligned} \mathbf{v}_\alpha \cdot \mathbf{n} &= q_\alpha^I \quad \text{ou} \quad p(\mathbf{x}, t) = p^I \quad \text{em } \Gamma_I \\ \mathbf{v}_\alpha \cdot \mathbf{n} &= q_\alpha^P \quad \text{ou} \quad p(\mathbf{x}, t) = p^P \quad \text{em } \Gamma_P \end{aligned} \quad (2.31)$$

onde $q_{I\alpha}$ e $q_{P\alpha}$ representam as vazões volumétricas da fase α nos poços injetores e produtores, respectivamente.

As condições inicial e de contorno para a equação de saturação são:

$$\begin{aligned} S_w(\mathbf{x}, 0) &= S_w^{t=0} \quad \text{em } \Omega \\ S_w(\mathbf{x}, t) &= S_w^I \quad \text{em } \Gamma_I \\ K\nabla S_w(\mathbf{x}, t) \cdot \mathbf{n} &= 0 \quad \text{em } \Gamma_N \end{aligned} \quad (2.32)$$

2.5. CONDIÇÕES INICIAIS E DE CONTORNO

Para a equação da concentração de um soluto em um fluido solvente, ver Eq. (2.29), as condições iniciais e de contorno são definidas de modo similar:

$$\begin{aligned}C(\mathbf{x}, 0) &= C^{t=0} \quad \text{em } \Omega \\C(\mathbf{x}, t) &= \hat{C} \quad \text{em } \Gamma_I \\D\nabla C(\mathbf{x}, t) \cdot \mathbf{n} &= 0 \quad \text{em } \Gamma_N\end{aligned}\tag{2.33}$$

Capítulo 3

Formulação Numérica

A ideia básica de praticamente todos os métodos numéricos para a solução de EDPs é a discretização de um problema contínuo com infinitos graus de liberdade (DOFs) e desse modo a redução para um número finito de DOFs. Este problema discreto resulta em um sistema de equações com um número finito de variáveis, o qual normalmente pode ser resolvido utilizando um método adequado (ver seção 3.4 para detalhes) (Chen *et al.*, 2006).

No Método das Diferenças Finitas (MDF), o qual ainda é o padrão na simulação numérica de reservatórios, as derivadas das equações originais são simplesmente substituídas por quocientes de diferenças. Os valores das variáveis são definidos apenas por um número específico de pontos, os quais podem estar localizados nos vértices ou no interior das células (Fortuna, 2000).

O processo de discretização no caso do MEF é diferente na medida em que ele usualmente requer a reformulação das EDPs em uma forma fraca equivalente. As variáveis desconhecidas para o problema abordado são a pressão e saturação (além da velocidade, para formulações mistas), as quais podem ser descritas em qualquer ponto da região usando uma função de interpolação (também conhecida como função de forma do elemento). Os pontos de suporte para estas funções são os nós da malha, os quais são conectados para formar os elementos.

Neste trabalho, foram experimentados tanto o clássico Método de Elementos Finitos de Galerkin quanto uma formulação dos Métodos dos Elementos Finitos Mistos para resolver a equação da pressão descrita na seção 2.3. Para a resolução da equação da saturação (ver seção 2.4) foi utilizado o MEF acrescido de um termo de estabilização do tipo SUPG. Os resultados destas equações foram acoplados através de um método Sequencial Implícito. Estas metodologias são discutidas nas seções subsequentes e

referências são citadas para informações adicionais.

3.1 Método dos Elementos Finitos de Galerkin

A fim de resolver o problema da equação da pressão utilizando o MEF, é necessário inicialmente expressar o mesmo na chamada *forma fraca*. O primeiro passo para obter esta nova formulação (também chamada de problema variacional) é multiplicar a EDP por uma função w chamada de *função de ponderação*, em seguida o resultado é integrado no domínio Ω e finalmente uma integração por partes é efetuada nos termos com derivadas de segunda ordem (Hughes, 2000). A função desconhecida (pressão p , por exemplo) é chamada de *função tentativa* (ou de aproximação). Espaços de funções apropriados devem ser definidos tanto para função de ponderação quanto para a função tentativa.

O problema variacional para a Eq. (2.16) considerando todas as hipóteses mencionadas tem a seguinte forma:

Problema 1 (Contínuo). *Encontre $p \in P$ tal que:*

$$A(p, w) = f(w) \quad \forall w \in W \quad (3.1)$$

com

$$P = \{p \in H^1(\Omega); p = p_D \text{ em } \Gamma_D\} \quad (3.2)$$

$$W = \{w \in H^1(\Omega); w = 0 \text{ em } \Gamma_D\} \quad (3.3)$$

$$A(p, w) = \int_{\Omega} \lambda K \nabla p \cdot \nabla w d\Omega \quad \forall p, w \in P, W \quad (3.4)$$

$$f(w) = \int_{\Omega} q_t w d\Omega \quad \forall w \in W \quad (3.5)$$

onde p_D é o valor prescrito de p nos contornos de Dirichlet e H^1 é o Espaço de Sobolev de funções com derivadas generalizadas de primeira ordem quadrado-integráveis, podendo este espaço ser definido de modo mais geral do seguinte modo (Hughes, 2000):

$$H^k(\Omega) = \{w \in L_2(\Omega); w_{,x} \in L_2(\Omega); \dots; w_{\underbrace{x \dots x}_{k \text{ vezes}}} \in L_2(\Omega)\} \quad (3.6)$$

onde

$$L_2(\Omega) = \{w | \int_{\Omega} w^2 d\Omega < \infty\} \quad (3.7)$$

Isto significa que ao passo que a solução da EDP tem que existir em um espaço de funções com derivadas contínuas (*forma forte*), o Espaço de Sobolev de grau 1

exigido pela forma fraca permite funções com derivadas descontínuas (Langtangen, 2011). A existência e unicidade da solução deste problema é dada pelo lema de Lax e seu detalhamento completo pode ser encontrado em Garcia (1997).

Para a análise usando elementos finitos deste problema variacional, é necessário transformar esta formulação contínua definida pela Eq. (3.1) em um problema discreto. Considere o domínio Ω , o qual é discretizado por N elementos não sobrepostos E_i de modo que $\Omega_h = \bigcup_{i=1}^N E_i$ e $E_i \cap E_j = \emptyset, i \neq j$, onde o subíndice h representa um parâmetro de tamanho característico de malha que representa o problema discretizado.

O problema variacional discreto para a equação de pressão utilizando a abordagem clássica de Galerkin descrita em Hughes (2000) e adotada em vários outros trabalhos (Wells *et al.*, 2008; Barbosa *et al.*, 2009) é:

Problema 2 (Discreto). *Encontre $p_h \in P_h$ tal que:*

$$A(p_h, w_h) = f(w_h) \quad \forall w_h \in W_h \quad (3.8)$$

com

$$P_h = \{p_h \in H^1(\Omega_h); p_h \in \mathbf{P}^k(E_i); p_h = p_D \text{ em } \Gamma_D\} \quad (3.9)$$

$$W_h = \{w_h \in H^1(\Omega_h); w_h \in \mathbf{P}^k(E_i); w_h = 0 \text{ em } \Gamma_D\} \quad (3.10)$$

$$A(p_h, w_h) = \int_{\Omega_h} \lambda K \nabla p_h \cdot \nabla w_h d\Omega \quad \forall p_h, w_h \in P_h, W_h \quad (3.11)$$

$$f(w_h) = \int_{\Omega_h} Q_t w_h d\Omega \quad \forall w_h \in W_h \quad (3.12)$$

onde $\mathbf{P}^k(E_i)$ define funções de forma com ordem k para o elemento finito E_i . Na Fig. 3.1 são apresentados elementos triangulares para a formulação clássica de Galerkin com diferentes ordens para o polinômio da função de forma, onde pode ser observado que um aumento da ordem de aproximação implica em uma maior quantidade de graus de liberdade por elemento, o que melhora a precisão da interpolação, porém ao custo de um maior tempo computacional. A Eq. (3.8) aplicada a uma malha computacional resulta em um conjunto de equações discretas na variável p , cuja solução pode ser obtida usando o métodos descrito na seção 3.4.

3.2 Método dos Elementos Finitos Mistos

Na simulação de escoamentos em meios porosos é fundamental se obter uma boa aproximação para a variável de velocidade, pois esta será utilizada para o acoplamento entre

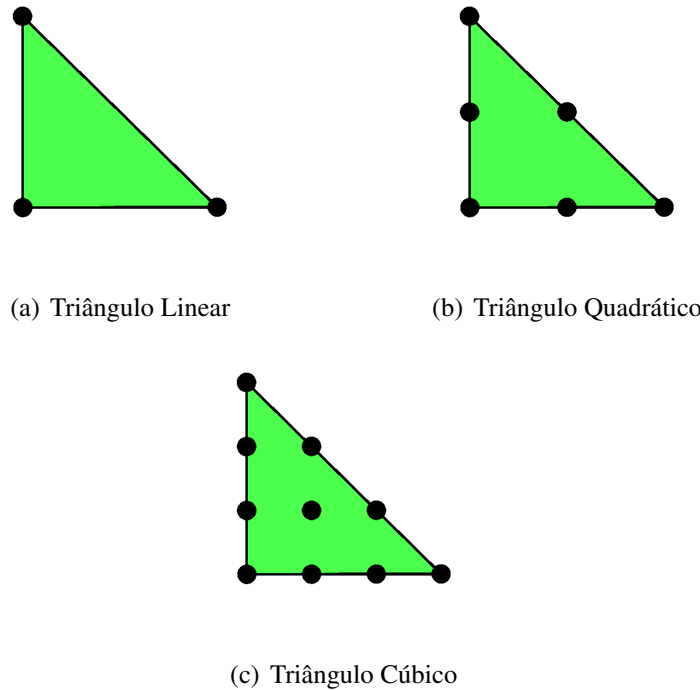


Figura 3.1 Exemplo de elementos triangulares para análise via MEF.

as equações de pressão e saturação. Após a utilização de um método numérico como o descrito na seção anterior apenas a variável de pressão estará definida, sendo que esta aproximação será de segunda ordem para elementos triangulares lineares como os que foram usados na maior parte deste trabalho. Portanto, seria necessário calcular em uma etapa seguinte a velocidade a partir da pressão calculada.

O modo mais direto de fazer isto seria usar diretamente a equação de Darcy (Eq. (2.6)), a qual permite obter o campo de velocidades a partir do gradiente de pressão. Porém, tal alternativa apresenta as grandes desvantagens de requerer a diferenciação do campo de pressão, diminuindo a ordem de convergência da aproximação e de não garantir a continuidade da velocidade nas fronteiras, já que um campo contínuo linear de pressão terá o seu gradiente representado como descontínuo entre elementos vizinhos. Uma alternativa utilizada por alguns autores (Loula *et al.*, 1995; Barbosa *et al.*, 2009) é realizar um pós-processamento ao final do cálculo do campo de pressão para se obter um novo problema variacional na variável de velocidade, o qual ao ser resolvido garante uma melhor ordem de aproximação e a continuidade do campo entre os elementos.

Uma outra alternativa, a qual é apresentada nesta seção e utilizada neste trabalho, é o uso de uma formulação mista. A maior motivação para o desenvolvimento e uso do Método dos Elementos Finitos Mistos (MEFM) é que em algumas aplicações, como

no caso da simulação de escoamentos em meios porosos, uma variável vetorial (ex: velocidade de um fluido) também é de interesse, sendo que os métodos do tipo MEFM foram desenvolvidos para aproximar tanto esta variável quanto a escalar (ex: pressão) simultaneamente e fornecer aproximações de alta ordem para ambas as variáveis.

Entretanto, esta vantagem é obtida ao custo de uma maior complexidade da formulação, já que ao invés de um único espaço de elementos finitos o MEFM requer dois espaços, os quais *não* podem ser escolhidos arbitrariamente pois têm que satisfazer uma condição do tipo *inf-sup* para serem estáveis (Chen *et al.*, 2006).

Em Raviart e Thomas (1977) foi proposta a primeira família de espaços de elementos finitos mistos que satisfazem a condição de estabilidade conhecida como *Ladyshenskaja-Babuska-Brezzi Condition* (LBB) (ver Brezzi e Fortin (1991) para uma discussão detalhada sobre esta condição) para problemas elípticos de 2ª ordem como o representado pela equação da pressão (Eq. (2.16)), sendo que, depois desta, várias outras combinações de espaços já foram sugeridos na literatura (Brezzi *et al.*, 1985; Chen *et al.*, 2006).

De modo a ilustrar o procedimento para obtenção da forma fraca, consideremos inicialmente a equação da pressão desenvolvida na seção 2.3, a qual é reproduzida a seguir, onde os subíndices foram omitidos por questão de clareza:

$$-\nabla \cdot (K\lambda \nabla p) = Q \quad (3.13)$$

sendo que ela está sujeita à condição de fluxo zero em todas as fronteiras exteriores do domínio, conforme explicado na seção 2.5:

$$K\lambda \nabla p \cdot \mathbf{n} = 0 \quad \text{em } \Gamma_N \quad (3.14)$$

De modo a garantir que a variável vetorial (velocidade $\mathbf{v}$) tenha componentes contínuas em Ω , podemos utilizar um Espaço de Sobolev para funções vetoriais definido por:

$$\mathbf{H}(\text{div}, \Omega) = \{ \boldsymbol{\zeta} = (\zeta_1, \zeta_2) \in (L_2(\Omega) \times L_2(\Omega)); \nabla \cdot \boldsymbol{\zeta} \in L_2(\Omega) \} \quad (3.15)$$

Assim, podemos definir o espaço $\mathbf{V} = \{ \boldsymbol{\zeta} \in \mathbf{H}(\text{div}, \Omega); \boldsymbol{\zeta} \cdot \mathbf{n} = 0 \text{ em } \Gamma_N \}$ para garantir a condição de fluxo zero na fronteira e o espaço $W = L_2(\Omega)$, onde é importante observar que funções em W não são necessariamente contínuas.

Além disso, definimos a seguinte notação para o produto interno em $L_2(\Omega)$:

$$(v, w) = \int_{\Omega} v(x)w(x)dx \quad (3.16)$$

Utilizando a definição da velocidade, podemos escrever:

$$\mathbf{v} = -\mathbf{K}\lambda \nabla p \quad (3.17)$$

Logo, a Eq. (3.13) se torna:

$$\nabla \cdot \mathbf{v} = Q \quad (3.18)$$

Multiplicando-se a Eq. (3.17) por $\zeta \in \mathbf{V}$, passando os termos de permeabilidade e mobilidade para o lado esquerdo da equação e integrando o resultado no domínio, usamos a definição de produto interno da Eq. (3.16) para obter:

$$((\mathbf{K}\lambda)^{-1}\mathbf{v}, \zeta) = -(\zeta, \nabla p) \quad (3.19)$$

Aplicando o teorema da divergência no lado direito desta equação, obtemos:

$$((\mathbf{K}\lambda)^{-1}\mathbf{v}, \zeta) = (\nabla \cdot \zeta, p) \quad (3.20)$$

E multiplicando a Eq.(3.18) por $w \in W$, temos:

$$(\nabla \cdot \mathbf{v}, w) = (Q, w) \quad (3.21)$$

De posse de todas estas definições, podemos descrever o problema variacional (*forma fraca*) para as equações de pressão e velocidade do seguinte modo (Chen *et al.*, 2006):

Problema 3 (Contínuo). *Encontre $p \in W$ e $\mathbf{v} \in \mathbf{V}$ tal que:*

$$A(p, w, \mathbf{v}, \zeta) = f(w) \quad \forall \zeta \in \mathbf{V}, \quad \forall w \in W \quad (3.22)$$

com

$$\mathbf{V} = \{\zeta \in \mathbf{H}(\text{div}, \Omega); \zeta \cdot n = 0 \text{ em } \Gamma_N\} \quad (3.23)$$

$$W = L_2(\Omega) \quad (3.24)$$

$$A(p, w, \mathbf{v}, \zeta) = ((\mathbf{K}\lambda)^{-1}\mathbf{v}, \zeta) - (\nabla \cdot \zeta, p) - (\nabla \cdot \mathbf{v}, w) \quad (3.25)$$

$$f(w) = -(Q, w) \quad (3.26)$$

De modo a se possibilitar a análise deste problema variacional via MEFM, é necessário representar o mesmo na forma discreta. Para isto, precisamos definir espaços finitos que substituam os espaços $\mathbf{V}$ e W determinados conforme mostrado nas Eqs. (3.23)

e (3.24). Neste trabalho foi utilizada a combinação conhecida como espaços *Brezzi-Douglas-Marini* (BDM) (Brezzi *et al.*, 1985), os quais têm a característica de permitir obter a mesma ordem de convergência do erro para a variável vetorial que os espaços de *Raviart-Thomas* (Raviart e Thomas, 1977), porém com a vantagem de terem uma dimensão menor. Os espaços BDM podem ser definidos para cada elemento finito da malha do modo a seguir, considerando um $k \geq 1$ (Chen *et al.*, 2006):

$$\mathbf{V}_h(E_i) = \mathbf{P}^k(E_i) \times \mathbf{P}^k(E_i) \quad (3.27)$$

$$\mathbf{W}_h(E_i) = \mathbf{P}^{k-1}(E_i) \quad (3.28)$$

Para o caso mais simples, com $k = 1$, obtemos a seguinte definição para $\mathbf{V}_h$ em um triângulo:

$$\mathbf{V}_h(E_i) = \{\zeta_h = (a_{E_i}^1 + a_{E_i}^2 x_1 + a_{E_i}^3 x_2, a_{E_i}^4 + a_{E_i}^5 x_1 + a_{E_i}^6 x_2)\} \quad (3.29)$$

Este triângulo é mostrado na Fig. 3.2(a), onde percebe-se que os graus de liberdade em $\mathbf{V}_h$ são os valores das componentes normais das funções em dois pontos diferentes para cada aresta do elemento, tendo este espaço portanto uma dimensão igual a seis. Para $k = 1$, o espaço $\mathbf{W}_h$ é constante em cada elemento, tendo portanto uma dimensão unitária. O problema discreto para o MEFM pode então ser descrito por:

Problema 4 (Discreto). *Encontre $p_h \in W_h$ e $\mathbf{v}_h \in \mathbf{V}_h$ tal que:*

$$A(p_h, w_h, \mathbf{v}_h, \zeta_h) = f(w_h) \quad \forall \zeta_h \in \mathbf{V}_h, \quad \forall w_h \in W_h \quad (3.30)$$

com

$$A(p_h, w_h, \mathbf{v}_h, \zeta_h) = ((\mathbf{K}\lambda)^{-1} \mathbf{v}_h, \zeta_h) - (\nabla \cdot \zeta_h, p_h) - (\nabla \cdot \mathbf{v}_h, w_h) \quad (3.31)$$

$$f(w_h) = -(Q_h, w_h) \quad (3.32)$$

3.3 Estabilização do MEF via SUPG

O MEF de Galerkin permite obter respostas bastantes acuradas para problemas com soluções suaves, como normalmente é o caso para equações essencialmente elípticas como a equação de pressão. Entretanto, ele não se mostra adequado para a simulação de fenômenos dominados por termos advectivos, caso frequente de equações com características hiperbólicas como a equação de saturação descrita em 2.4. Nestes casos a solução

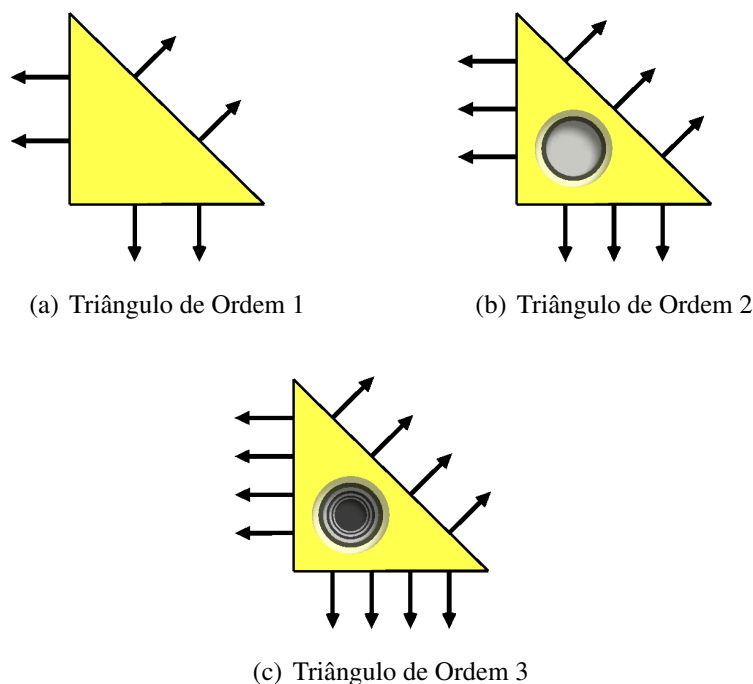


Figura 3.2 Exemplo de elementos triangulares para análise via MEFM.

pode se tornar instável, sendo que tal instabilidade é transportada para todo o domínio, deteriorando rapidamente a acurácia global.

Mesmo antes do aparecimento deste problema no contexto do MEF, ele já era bastante conhecido do estudo do MDF, podendo ser explicado em ambos os casos pelo uso de aproximações que se baseiam no uso de discretizações centradas, ou seja, quando as contribuições vindo de todas as direções possuem o mesmo peso na solução. Inicialmente, uma das técnicas para se contornar este tipo de problema era adicionar um termo artificial de difusão diretamente nas EDPs que se desejava aproximar (Papastavrou, 1998). Todavia, esta técnica apresenta a desvantagem fundamental de alterar a descrição matemática do problema, ocasionando assim uma perda de consistência, já que mesmo que se utilize uma discretização tendendo ao contínuo continuaria-se obtendo uma resposta que não tenderia à analítica. Para alguns problemas tal alternativa é aceitável, pois os termos adicionados são de alta ordem e portanto afetam pouco a consistência. Porém, uma solução mais apropriada seria o uso de termos baseados no resíduo, o que garantiria a consistência para qualquer caso.

Uma solução proposta no contexto do MDF consiste na ponderação da solução com um peso maior para a informação que segue o sentido do fluxo (esta técnica é comumente conhecida como *Upwinding*). No caso do MEF, Brooks e Hughes (1982) propuseram um

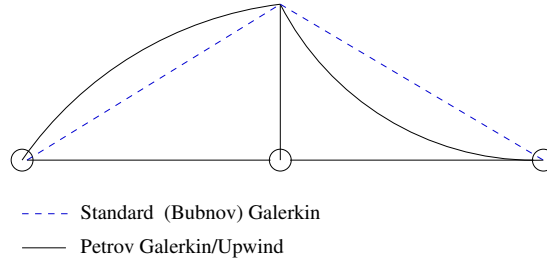


Figura 3.3 Comparação entre diferentes definições da função de ponderação (retirado de Monajemi (2009)).

método no qual o espaço da função de ponderação não mais corresponde ao espaço da função tentativa utilizada (ou seja, um método do tipo Petrov-Galerkin), sendo adicionado um termo de perturbação com caráter difusivo que atua apenas na direção da linha de fluxo no nível do elemento. Este método, o qual se tornou conhecido pelo nome de *Streamline Upwind Petrov-Galerkin* (SUPG), foi extensamente utilizado na literatura para várias aplicações, inclusive na resolução da equação de saturação em escoamentos multifásicos (Barbosa *et al.*, 2009; Garcia, 1997; Loula *et al.*, 1995). Um exemplo desta função de ponderação modificada tem sua representação gráfica mostrada na Fig. 3.3, onde se considera um fluxo da esquerda para a direita. Pode ser observado claramente na figura o maior peso considerado para o elemento que está a montante do escoamento quando comparado ao que se encontra a jusante. Isto implica que a modificação a ser efetuada na função de ponderação deve de alguma forma ser dependente da direção e sentido do escoamento.

Esta nova função de ponderação para a formulação estabilizada, aqui representada por w_s , pode ser descrita matematicamente conforme a equação seguinte:

$$w_s = w + \tau_s \mathbf{v} \cdot \nabla w \quad (3.33)$$

onde $\mathbf{v}$ representa o vetor velocidade, w a função de ponderação original e τ_s um parâmetro de estabilização que depende do tamanho do elemento e do módulo da velocidade.

Uma vez definida tal função, aplica-se um procedimento semelhante ao utilizado para a discretização pelo Método de Galerkin onde se integra a Eq. (2.25), desprezando-se aqui o termo de fonte por simplicidade, porém a multiplicação é pela função de ponderação modificada (Eq. (3.33)), para se obter depois de algum algebrismo e de modo consistente a seguinte forma variacional discreta para o problema da equação de saturação (Wells *et al.*, 2008):

Problema 5 (Discreto). Dados $S_h^n, f_{,S}$ e $\mathbf{v}_h$, encontre $S_h^{n+1} \in X_h$ tal que:

$$F(S_h^{n+1}, w_h) = 0 \quad \forall w_h \in W_h \quad (3.34)$$

com

$$X_h = \{S_h \in H^1(\Omega_h); S_h \in \mathbf{P}^1(E_i); S_h = S_D \text{ em } \Gamma_D\} \quad (3.35)$$

$$W_h = \{w_h \in H^1(\Omega_h); w_h \in \mathbf{P}^k(E_i); w_h = 0 \text{ em } \Gamma_D\} \quad (3.36)$$

$$F(S_h^{n+1}, w_h) = \int_{\Omega} \left(w \phi \frac{S_h^{n+1} - S_h^n}{\Delta t} \right) d\Omega + \int_{\Omega} w f_{,S} \mathbf{v}^{n+1} \cdot \nabla S_h^{n+1} d\Omega + \sum_{E_i} \int_{E_i} (\mathbf{v}^{n+1} \cdot \nabla w) \tau_s r^{n+1} d\Omega \quad (3.37)$$

onde o resíduo r é descrito por:

$$r^{n+1} = \phi \frac{S_h^{n+1} - S_h^n}{\Delta t} + f_{,S} \mathbf{v}^{n+1} \cdot \nabla S_h^{n+1} \quad (3.38)$$

Ao se observar o problema acima, percebe-se que os dois primeiros termos do lado direito da Eq. (3.37) correspondem ao que seria obtido através de uma discretização via MEF de Galerkin, sendo o terceiro termo o responsável pela estabilização do método através de um somatório das contribuições calculadas elemento por elemento. A consistência da formulação utilizada é garantida pelo fato do termo adicionado ser diretamente proporcional ao resíduo da solução, o qual tende a zero para a resposta exata, cancelando portanto qualquer difusão não-física que poderia ser adicionada neste caso.

O termo $f_{,S}$, que representa a derivada do fluxo fracional em relação à saturação, torna este problema não-linear. Esta não-linearidade pode ser tratada através de diversos métodos, entre eles os métodos de Newton-Raphson e de iteração de Picard. Todavia, neste trabalho foi adotada a mesma estratégia usada por outros autores (Carvalho, 2005; Silva, 2008) de linearizar a equação de saturação através do uso do termo $f_{,S}$ calculado no passo de tempo anterior. Esta estratégia tem como vantagem a sua simplicidade, por outro lado requer o uso de passos de tempo pequenos o suficiente para evitar que a linearização do problema degrade a solução.

Existem diversas sugestões na literatura acerca do modo mais adequado de definir o parâmetro de estabilização τ_s (Brooks e Hughes, 1982; Barbosa *et al.*, 2009; Papastavrou, 1998; Wells *et al.*, 2008). De um modo geral, este termo está relacionado a uma dimensão característica de malha e à velocidade de transporte, tendo portanto que ser calculado para cada elemento. Neste trabalho optou-se pelo uso da expressão apresentada a seguir (Wells *et al.*, 2008), a qual, apesar de bastante simples comparada a outras encontradas na

literatura, normalmente se mostra adequada para problemas altamente hiperbólicos como é o caso da equação da saturação ao se desprezar os termos de capilaridade e gravidade:

$$\tau_s = \frac{h}{2||\mathbf{v}||} \quad (3.39)$$

3.3.1 Termo de Captura de Choque

Apesar do SUPG permitir a obtenção de soluções numéricas estáveis em comparação ao MEF de Galerkin, o mesmo não garante a inexistência de oscilações em regiões próximas às discontinuidades da solução exata. Isto se explica pelo fato deste método não ser do tipo monótono ou preservador de monotonicidade. Métodos monótonos são aqueles nos quais a solução numérica mantém ao longo do tempo o seu sinal para todos os nós da malha (Codina, 1992; Papastavrou, 1998).

Um meio para garantir uma alta precisão na região de solução suave e ao mesmo tempo evitar o aparecimento de oscilações espúrias na região de choque é utilizar métodos não-lineares, os quais dependem portanto da própria solução do problema. Isto se deve à condição estabelecida pelo teorema de Godunov de que métodos lineares e que preservem a monotonicidade podem ter no máximo uma convergência de primeira ordem (Leveque, 1992).

A ideia básica do uso de um termo de captura de choque é adicionar uma dissipação numérica (também chamada de viscosidade artificial) extra na região de discontinuidade. Neste trabalho foi utilizado um termo de dissipação isotrópico, ou seja, que acrescenta a mesma viscosidade em todas as direções (Wells *et al.*, 2008). Todavia, existem na literatura exemplos de usos de termos anisotrópicos (Codina, 1992; Papastavrou, 1998), já que o uso do SUPG implica na adição de uma certa viscosidade artificial na direção das linhas de corrente, sendo em princípio necessário acrescentar dissipação apenas na componente perpendicular ao fluxo em cada ponto. Em todo caso, é importante mencionar que assim como na estabilização via SUPG a viscosidade artificial adicionada é proporcional ao resíduo da solução, garantindo a consistência do método. Sendo assim, podemos somar o termo seguinte à Eq. (3.37) para obter uma nova forma variacional discreta:

$$\begin{aligned} F(S_h^{n+1}, w_h) = & \int_{\Omega} \left(w \phi \frac{S^{n+1} - S^n}{\Delta t} \right) d\Omega + \int_{\Omega} w f_{,S} \mathbf{v}^{n+1} \cdot \nabla S^{n+1} d\Omega + \\ & \sum_{E_i} \int_{E_i} (\mathbf{v}^{n+1} \cdot \nabla w) \tau_s r^{n+1} d\Omega + \sum_{E_i} \int_{E_i} \nu_{shock} \nabla w \cdot \nabla S_h^{n+1} d\Omega \end{aligned} \quad (3.40)$$

onde ν_{shock} é a viscosidade artificial definida por:

$$v_{shock} = \begin{cases} \frac{\beta h |r^{n+1}|}{2 \|\nabla S_h^n\|} & \text{se } \|\nabla S_h^n\| \neq 0 \\ 0 & \text{caso contrário} \end{cases} \quad (3.41)$$

onde β é um parâmetro dependente do problema. Neste trabalho foi adotado o valor de $\beta = 2.0$, conforme sugestão encontrada na literatura (Wells *et al.*, 2008) para problemas de escoamentos em meios porosos.

De modo a exemplificar a influência dos termos tanto de estabilização como de captura de choque, são apresentados a seguir os resultados obtidos para a equação linear de advecção, a qual pode ser vista com uma versão simplificada da Eq. (2.29) com o tensor de difusão e o termo de reação nulos, sendo representada no caso unidimensional pela expressão a seguir:

$$\frac{\partial C(x,t)}{\partial t} = -v \frac{\partial C(x,t)}{\partial x} \quad (3.42)$$

onde para este exemplo consideramos um vetor velocidade v unitário na direção x e as seguintes condições iniciais e de contorno:

$$\begin{aligned} C(x,0) &= 0.0 & \text{em } \Omega \\ C(x,t) &= 1.0 & \text{em } x = 0 \end{aligned} \quad (3.43)$$

A solução analítica do problema apresentado é uma função degrau com altura 1 que se desloca na direção positiva do eixo x . A Fig. 3.4 apresenta a solução numérica obtida para o instante de tempo $t = 0.5s$ em uma malha unidimensional com 500 elementos lineares utilizando exclusivamente o MEF de Galerkin conforme descrito na seção 3.1. Como esperado, a localização da descontinuidade foi resolvida com precisão para este exemplo simples, porém verifica-se o surgimento de oscilações espúrias na região do choque que tendem a crescer indefinidamente, tornando a solução instável.

De modo a estabilizar a solução aplicamos o método SUPG descrito na seção 3.3 e representado pela Eq. (3.37), obtendo o resultado mostrado na Fig. 3.5, onde pode-se perceber uma eliminação de grande parte da oscilação presente na aproximação numérica inicial. Entretanto, apesar de o choque ter permanecido com uma definição quase tão boa quanto na análise com o MEF de Galerkin, ainda se percebe a existência de *overshoots* e *undershoots* na região de descontinuidade.

Por fim, acrescentamos à formulação o termo de captura de choque com adição de viscosidade artificial conforme descrito pelas Eqs. (3.40) e (3.41). Como pode ser visto na Fig. 3.6, a solução numérica se encontra praticamente livre de qualquer oscilação sem que isto tenha afetado a solução nas regiões distante da descontinuidade. Apenas próximo

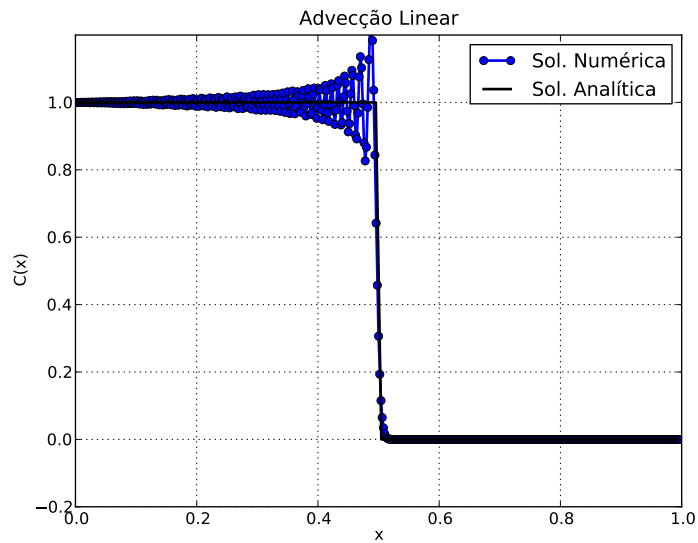


Figura 3.4 Comparação entre solução analítica e numérica considerando o uso do MEF de Galerkin para o caso de advecção pura.

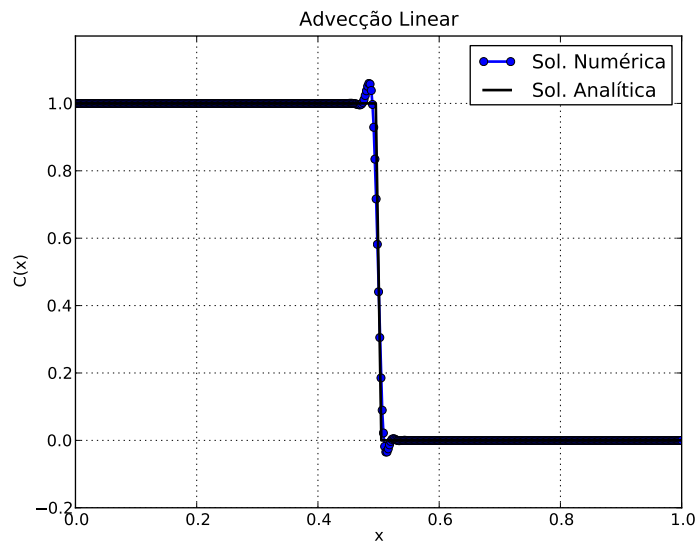


Figura 3.5 Comparação entre solução analítica e numérica considerando o uso do MEF com estabilização via SUPG para o caso de advecção pura.

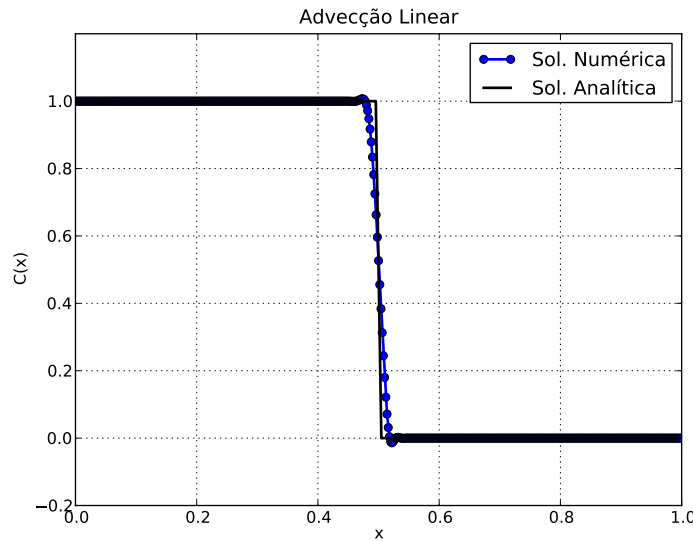


Figura 3.6 Comparação entre solução analítica e numérica considerando o uso do MEF com estabilização via SUPG e adição de viscosidade artificial para o caso de advecção pura.

a esta região de maior variação é que pode ser percebida uma maior difusão da solução quando comparada com a analítica, sendo tal consequência esperada devido ao fato de a solução poder ser no máximo de primeira ordem na região de choque conforme previsto pelo teorema de Godunov (Leveque, 1992).

3.4 Multigrid

A ideia básica dos métodos Multigrid é acelerar a solução de sistemas de equações em uma malha fina usando correções calculadas em uma malha menos refinada. A motivação para esta abordagem vem da observação do comportamento do erro da solução numérica no domínio da frequência. Erros de alta frequência, os quais estão associados à variações locais da solução, são bem resolvidos por métodos iterativos convencionais (Gauss-Seidel, Gradientes Conjugados, etc.). Entretanto, erros de baixa frequência, associados à variação global das soluções, são menos sensíveis a esses métodos.

Um método Multigrid inicia amortecendo os erros de alta frequência associados à solução inicial na malha fina usando algum tipo de método de relaxação (usualmente um método iterativo que seria usado isoladamente). Uma vez atingido este objetivo inicial, a realização de mais iterações não seria mais efetiva na redução do erro associado a este nível de refinamento. Sendo assim, o resíduo é transferido para uma malha menos refinada, onde os modos de baixa frequência da malha fina se comportam como de alta

frequência, sendo, portanto, facilmente eliminados usando algum tipo de método direto, no caso da malha possuir uma quantidade suficientemente pequena de nós, ou mesmo usando o mesmo método de relaxação aplicado na malha mais fina. As correções são então calculadas na malha menos refinada e transferidos de volta para a malha fina de modo a atualizar a solução. Este procedimento pode ser aplicado recursivamente em uma sequência de malhas cada vez menos refinadas, a fim de que cada nível nesta hierarquia seja responsável pela eliminação de uma faixa de frequência de erro (Briggs *et al.*, 2000).

Existem dois tipos principais de método Multigrid: o Multigrid Geométrico (GMG), o qual trabalha diretamente nas malhas que representam o domínio discreto, e o Multigrid Algébrico (AMG), o qual opera apenas no sistema de equações lineares resultantes da aplicação de uma formulação numérica, não tendo portanto uma interpretação geométrica direta.

A maior motivação para o desenvolvimento do AMG foi a necessidade de métodos robustos e flexíveis para acelerar a convergência sem a exigência de calibração fina para cada problema, como é frequentemente o caso para métodos GMG, o qual requer atenção especial na geração da sequência de malhas menos refinadas, principalmente na presença de geometrias complexas (Trottenberg *et al.*, 2001). Esta robustez é alcançada porque o AMG conta com um processo totalmente automático para gerar os “subproblemas” menos refinados, o qual pode agir apenas nas direções nas quais a relaxação irá efetivamente suavizar o erro para o problema dado, enquanto o GMG requer uma hierarquia fixa pré-determinada antes de começar o ciclo. A Fig. 3.7 apresenta uma comparação gráfica entre estes dois métodos.

Matematicamente, ambas as abordagens podem ser descritas aproximadamente do mesmo modo, é apenas necessário considerar que malhas e nós no AMG são sistemas de equações lineares e variáveis destes sistemas, respectivamente. Portanto, o método Multigrid pode ser brevemente descrito como segue:

Considere o sistema de equações no nível mais refinado:

$$\mathbf{A}_f \mathbf{p}_f = \mathbf{f}_f \quad (3.44)$$

onde $\mathbf{A}_f$ é a matriz resultante dos procedimentos descritos na seções 3.1 a 3.3 e o subíndice f representa valores no nível refinado. Depois de efetuadas algumas iterações com um método de relaxação, o erro estará suavizado e seu resíduo r_f será:

$$\mathbf{A}_f \hat{\mathbf{p}}_f - \mathbf{f}_f = r_f \quad (3.45)$$

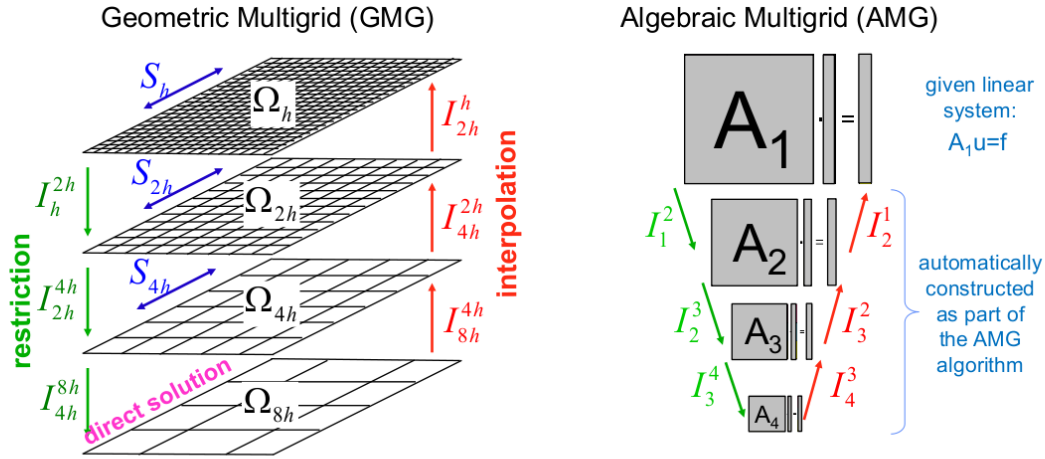


Figura 3.7 Comparação dos métodos Multigrid Geométrico e Algébrico (retirado de Trottenberg *et al.* (2001)).

onde $\hat{p}_f$ é a solução estimada para a variável de interesse. Subtraindo a Eq. (3.45) da Eq. (3.44) resulta:

$$A_f p_f - A_f \hat{p}_f = -r_f \quad (3.46)$$

e se o operador A for linear, esta equação pode ser apresentada como a equação de correção:

$$A_f \Delta p_f = -r_f \quad (3.47)$$

Assumindo, como já mencionado, que os modos de alta frequência do erro foram amortecidos depois da relaxação no nível mais refinado, o erro remanescente irá conter basicamente modos de baixa frequência que devem ser resolvidos, mas de modo a alcançar tal objetivo é necessário restringir tais modos ao nível menos refinado, onde os mesmos irão se comportar como modos de alta frequência, ou seja:

$$A_c \Delta p_c = -I_f^c r_f \quad (3.48)$$

Na Eq. (3.48), I_f^c representa o operador de restrição, o qual transfere valores do resíduo do nível refinado f para o nível menos refinado c . Uma vez que a Eq. (3.48) é resolvida adequadamente (frequentemente, se o nível for pouco refinado, um resolvidor direto será suficiente, já que este nível apresenta menores exigência de processamento e armazenamento), sua solução pode ser interpolada de volta para o nível refinado usando um operador de prolongamento I_c^f e a aproximação original no nível refinado pode então ser

corrigida:

$$\hat{p}_f^* = \hat{p}_f + \mathbf{I}_c^f \Delta p_c \quad (3.49)$$

onde $\hat{p}_f^*$ é a solução atualizada no nível refinado. É importante observar que este procedimento é intrinsecamente recursivo, podendo o sistema no nível menos refinado ser visto como um novo nível refinado, sendo suavizado novamente até que seja atingido um nível no qual as contribuições das correções nos níveis menos refinados sejam trazidas de volta para os seus respectivos níveis refinados. Existem diversos tipos de esquemas para lidar com esta estratégia, entre eles estão os ciclos V , W e F .

O Algoritmo 1 apresenta de modo recursivo e em pseudo-código o ciclo V , onde Ω_F e Ω_C representam as malhas mais e menos refinadas da sequência de modo absoluto. Já Ω_f e Ω_c representam as malhas mais e menos refinadas de modo relativo, ou seja, apenas em relação ao par e não à sequência.

Algoritmo 1 Ciclo V : $p_F \leftarrow V_F(p_F, f_F)$

Iterar n_1 vezes em $\mathbf{A}_f p_f = f_f$

while $\Omega_f \neq \Omega_C$ **do**

$r_f \leftarrow f_f - \mathbf{A}_f p_f$

$f_c \leftarrow \mathbf{I}_f^c r_f$

$p_c \leftarrow 0$

$p_c \leftarrow V_c(p_c, f_c)$

end while

$p_f \leftarrow p_f + \mathbf{I}_c^f p_c$

Iterar n_2 vezes em $\mathbf{A}_f p_f = f_f$

O Algoritmo 2 apresenta de modo recursivo e em pseudo-código o ciclo W , onde a principal diferença é o parâmetro m , o qual indica quantas vezes a recursão deve ser aplicada para cada nível. Na prática, se utilizam os valores de $m = 1$ (que equivale ao uso do ciclo V) e $m = 2$ (Briggs *et al.*, 2000).

A Fig. 3.8 mostra uma representação gráfica destes ciclos em um esquema Multigrid de 4 níveis, ver Briggs *et al.* (2000) ou Trottenberg *et al.* (2001) para uma explicação mais detalhada de cada um destes ciclos.

Algoritmo 2 Ciclo W: $p_F \leftarrow W_F(p_F, f_F)$

Iterar n_1 vezes em $A_f p_f = f_f$ **while** $\Omega_f \neq \Omega_C$ **do** $r_f \leftarrow f_f - A_f p_f$ $f_c \leftarrow I_f^c r_f$ $p_c \leftarrow 0$ $p_c \leftarrow W_c(p_c, f_c)$ m vezes**end while** $p_f \leftarrow p_f + I_c^f p_c$ Iterar n_2 vezes em $A_f p_f = f_f$

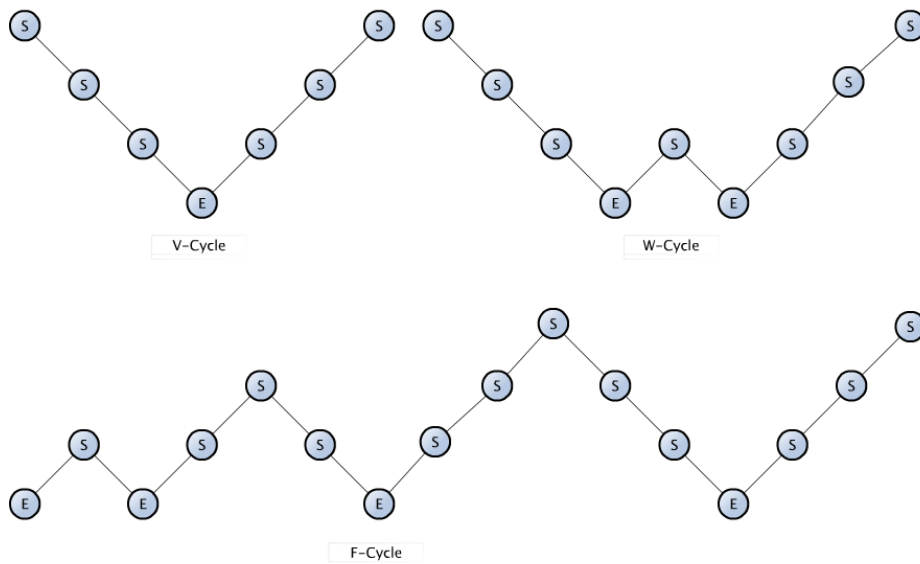


Figura 3.8 Ciclos V, W e F. S representa suavização e E é a solução no nível mais grosseiro.

Capítulo 4

Implementação Computacional

As formulações utilizadas neste trabalho, tanto a matemática quanto as numéricas, apresentam grandes desafios não apenas no que tange à compreensão teórica, mas também em relação ao desenvolvimento propriamente dito do *software*, já que é de fundamental importância uma implementação adequada destes métodos a fim de se atingir um compromisso vantajoso entre generalidade e desempenho computacional, pois estes dois objetivos frequentemente tendem a conduzir as diretivas de codificação do programa para direções diferentes.

Este capítulo se propõe a apresentar o modo como o programa de computador resultante deste trabalho foi elaborado, sendo que para isto foi adotada uma descrição *top-down*, ou seja, primeiramente são expostas as características gerais do programa e em seguida as divisões do mesmo em diferentes blocos que permitem uma melhor abordagem do problema. Cada um destes blocos é então subdividido em partes menores, as quais correspondem a tarefas específicas, que podem ter suas soluções diretamente codificadas utilizando uma linguagem de programação ou então serem resolvidas através de alguma biblioteca de software disponível na internet.

Outro aspecto a ser comentado a respeito da filosofia de desenvolvimento do programa é justamente que se buscou sempre utilizar soluções já prontas disponíveis publicamente, de modo a evitar o fenômeno popularmente conhecido como “reinventar a roda”. Além disso, foi dada preferência explícita a pacotes publicados segundo licenças livres (e.g., GNU GPL, BSD License, etc.), as quais permitem o uso, estudo, adaptação e redistribuição de programas de modo bastante transparente. Logicamente, o resultado deste trabalho também está disponível através de uma licença livre.

4.1 Estrutura Geral do Programa

A resolução de um problema através de um método numérico pode normalmente ser dividida em três etapas distintas:

- *Pré-processamento*: Definição de uma geometria que aproxime o domínio real e geração de uma malha de pontos discretos para ela. Além disso, nesta etapa são fornecidas as condições iniciais, de contorno e propriedades que buscam representar o problema real. Também nesta fase são definidos os parâmetros da análise a ser efetuada no passo a seguir.
- *Processamento*: Cálculo propriamente dito dos valores para as variáveis de interesse. Para isto, é utilizada alguma formulação numérica específica que trate adequadamente a descrição matemática do problema. Os resultados obtidos nesta etapa são analisados criticamente na fase seguinte.
- *Pós-processamento*: Visualização, cálculo de variáveis secundárias e interpretação dos resultados fornecidos pela etapa de processamento. Caso a resposta obtida seja satisfatória, a resolução do problema é encerrada neste ponto. Caso contrário, alterações são feitas na etapa de pré-processamento e o ciclo é reiniciado até que se atinja o objetivo.

Esta divisão também foi considerada no desenvolvimento do *Delfine*, sendo este o nome do programa desenvolvido neste trabalho. Os blocos responsáveis por executar cada uma das tarefas descritas anteriormente serão apresentados na forma de fluxograma e detalhados nas seções seguintes.

4.2 Pré-Processamento

Na Fig. 4.1 pode ser visto um fluxograma representando os vários passos necessários para a obtenção, a partir de um problema real, de um problema discreto passível de ser analisado numericamente.

Inicialmente, é necessário por parte do usuário ter uma descrição o mais precisa possível do problema de interesse, envolvendo a etapa da modelagem geométrica. De posse desta, parte-se para as etapas de discretização do domínio (i.e., geração da malha) e de definição de um arquivo de entrada de dados.

O *Delfine* oferece três caminhos para a obtenção da malha. A primeira é através de um gerador interno do *FEniCS/Dolfin*, o qual é utilizado principalmente na parte

de processamento. Este disponibiliza algumas rotinas básicas de geração de malhas, com a vantagem de tornar o programa independente de qualquer programa externo para este fim, já que os parâmetros para definição de malha são definidos no próprio arquivo de entrada de dados da simulação. Porém, como grande desvantagem podemos citar o fato da limitação quanto às geometrias disponíveis, pois apenas formas primitivas como linhas, retângulos, círculos, paralelepípedos e esferas podem ser descritos usando esta ferramenta.

Uma segunda alternativa é o uso da ferramenta *Gmsh* (Geuzaine e Remacle, 2009), a qual apresenta como vantagem uma maior flexibilidade na definição da geometria, pois este gerador disponibiliza várias operações que podem ser executadas em formas primitivas para a obtenção de outras mais complexas. Entre estas operações podemos citar adição, subtração, extrusão, escalonamento, divisão, entre outras. Além disso, o *Gmsh* dispõe de uma meta-linguagem própria que possibilita a escrita de scripts para a automatização e parametrização da geração de malhas. Como desvantagem, temos a necessidade de conversão do arquivo no formato *.msh* gerado pelo *Gmsh* para o padrão utilizado pelo *Delfine*, o qual é derivado diretamente do formato utilizado pelo *FEniCS/Dolfin*. Esta conversão é feita utilizando o script *delfine-convert*.

Como terceira e última alternativa temos o uso de outros geradores de malhas disponíveis publicamente ou não, como por exemplo o *Triangle*, o *Medit*, o *ExodusII* ou o pré-processador do *Abaqus*. Como vantagem desta alternativa podemos citar a liberdade em relação ao tipo de gerador de malhas a ser utilizado, já que o usuário pode escolher aquele com o qual tem mais familiaridade. Porém, os arquivos nos formatos de saída de qualquer um dos programas citados terá que ser convertido para o formato padrão do *Delfine* utilizando o script *dolfin-convert* (Logg e Wells, 2010).

A diferença básica entre a segunda e a terceira alternativa reside exatamente no tipo de script utilizado para a conversão dos arquivos de malha. O *dolfin-convert* é disponibilizado como parte da família de pacotes *FEniCS/Dolfin*, porém o mesmo faz apenas uma conversão das informações geométricas e topológicas da malha, ignorando as informações extras eventualmente presentes nos arquivos. Logo, não é possível importar *flags* de condições de contorno definidos no *Triangle* diretamente no *Delfine*, por exemplo. Sendo assim, tais informações têm que ser adicionadas manualmente aos arquivos de entrada do *Delfine*.

Já o *delfine-convert* é uma adaptação do *dolfin-convert* realizada durante este trabalho com o objetivo de importar todas as indicações de condições de contorno definidas no *Gmsh* e exportá-las no formato lido pelo *Delfine*. Esta funcionalidade é de fundamental

importância para problemas de maior complexidade, pois permite agilizar bastante a etapa de pré-processamento. Por isto, dentre toda as citadas, a segunda alternativa foi a mais utilizada ao longo deste trabalho.

Uma vez obtida a geometria discretizada, é necessário ler o arquivo de entrada de dados, o qual contém informações a respeito das condições iniciais, de contorno, propriedades de rochas e fluidos, parâmetros numéricos, etc. De um modo geral, tais informações podem ser fornecidas através de um arquivo de texto comum, desde que elas estejam ordenadas de modo estruturado para serem processadas pelo programa.

Entretanto, tal abordagem não apresenta uma robustez adequada, pois permite que pequenos erros do usuário na confecção do arquivo de dados passem despercebidos, o que pode acarretar tanto em demora para executar uma análise inicial, pois se torna necessário uma checagem manual de todos os parâmetros fornecidos até se encontrar a fonte de erro, como também se permite executar análises com valores não consistentes, os quais podem vir a gerar resultados totalmente não-físicos, podendo inclusive levar o usuário a interpretar o fenômeno de interesse de maneira errônea.

Sendo assim, de modo a aumentar a robustez da entrada de dados se optou pelo uso de arquivos estruturados no formato *Extensible Markup Language* (XML) (Bray *et al.*, 2000) com a utilização da linguagem de especificação de esquemas *Relax NG Compact Syntax* (RNC) (Clark, 2001). Esta linguagem permite definir um padrão lógico a ser seguido por qualquer arquivo XML gerado, caso contrário o mesmo não é considerado válido e o ponto exato onde o erro na entrada de dados foi encontrado é apresentado ao usuário antes mesmo de qualquer análise ter início. Este padrão é definido através de um arquivo chamado de *Schema Grammar* e todo arquivo de entrada será checado contra esta gramática através do programa *Relax NG Validator* (RNV) (Sheen, 2007). Este programa é sempre chamado automaticamente pelo *Delfine* antes do início da análise para verificar a entrada de dados do usuário, garantindo assim que a execução só será realizada caso exista uma consistência mínima nos dados fornecidos. Na Listagem 1 é apresentado um trecho de um arquivo de entrada típico. Nele pode ser observada a estrutura hierárquica utilizada para o armazenamento dos dados, os quais são tratados de modo completamente modular, ou seja, caso uma análise não precise de determinada informação, o bloco de dados referente a ela pode simplesmente ser deixado de fora do arquivo de dados, sem prejuízo na etapa de pré-processamento.

XML

```

<delfine>
  <geometry>
    <mesh dimension="2" order="1" type="gmsht">
      <filename>HomoIsoBCStruct.msh</filename>
    </mesh>
    <boundary-conditions>
      <well function="inject" id="301">.250</well>
    </boundary-conditions>
  </geometry>
  <physical>
    <rock-properties>
      <rock-type id="1">
        <porosity>1.0</porosity>
        <permeability type="per-domain">
          <Kxx>0.50</Kxx>
          <Kxy>0.0</Kxy>
          <Kxz>0.0</Kxz>
        </permeability>
      </rock-type>
    </rock-properties>
  </physical>
</delfine>

```

Listagem 1: Trecho de arquivo de entrada no formato *.xml.

4.3 Processamento

Uma vez lidos na etapa de pré-processamento os arquivos necessários para a execução da análise, tem início a etapa de processamento, a qual foi implementada neste trabalho de acordo com os fluxogramas apresentados nas Figs. 4.2 e 4.3.

Considerando uma formulação não-monolítica (i.e. segregada), conforme descrito no capítulo 2, o fluxograma da Fig. 4.2 representa os passos necessário para resolver a parte elíptica do problema de escoamentos multifásicos em meios porosos. As formulações matemática e numérica deste problema adotadas neste trabalho podem ser consultadas

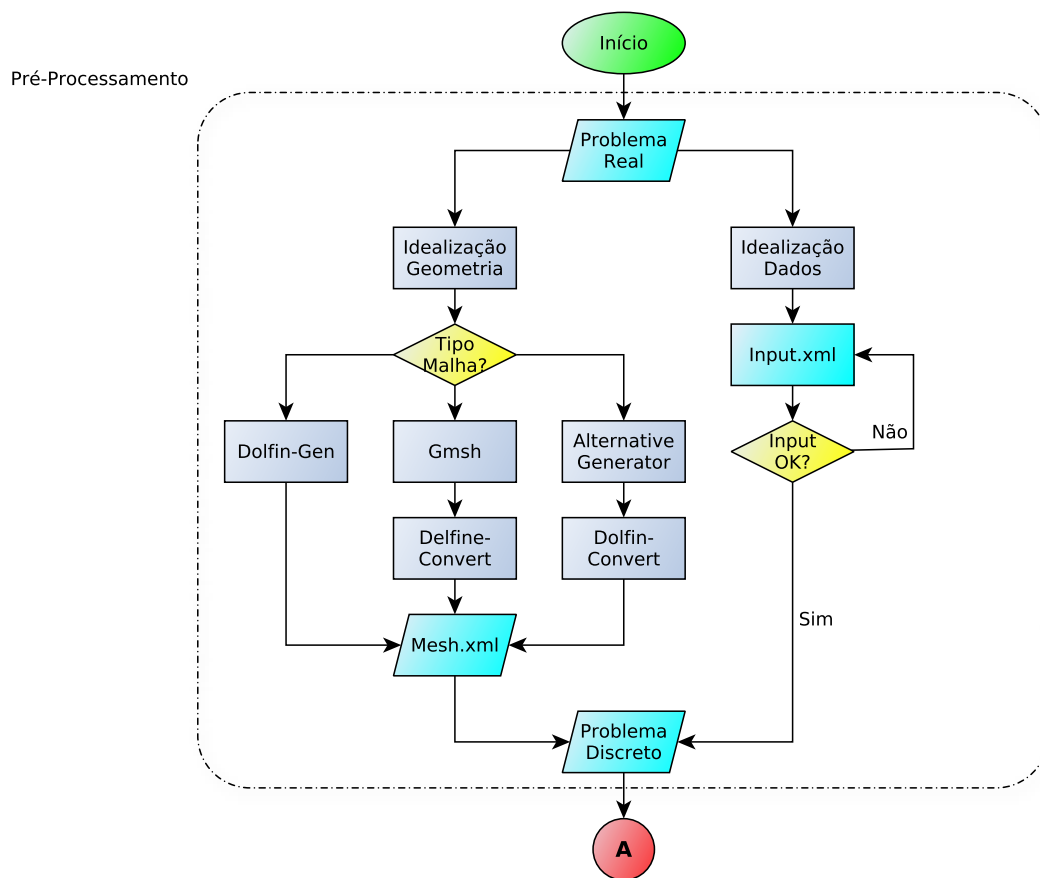


Figura 4.1 Fluxograma da etapa de pré-processamento.

nas seções 2.3 e 3.1, respectivamente.

A parte parabólica/hiperbólica do problema é descrita pelo fluxograma apresentado na Fig. 4.3. As formulações matemática e numérica deste problema adotadas neste trabalho podem ser consultadas nas seções 2.4 e 3.3, respectivamente.

O usuário tem a sua disposição duas alternativas de métodos numéricos para resolução do problema elíptico: o Método dos Elementos Finitos (MEF) e o Método dos Elementos Finitos Mistos (MEFM). As diferenças entre os dois métodos do ponto de vista da formulação numérica são discutidas no capítulo 3, sendo nesta seção discutidos apenas os aspectos de implementação. Já para o problema hiperbólico foi adotado o MEF com estabilização via *Streamline Upwind Petrov-Galerkin* (SUPG). Todos os métodos foram codificados utilizando a interface na linguagem *Python* da ferramenta *FEniCS/DOLFIN*. A seguir faremos uma descrição geral deste pacote computacional e em seguida mostraremos como o mesmo foi utilizado neste trabalho.

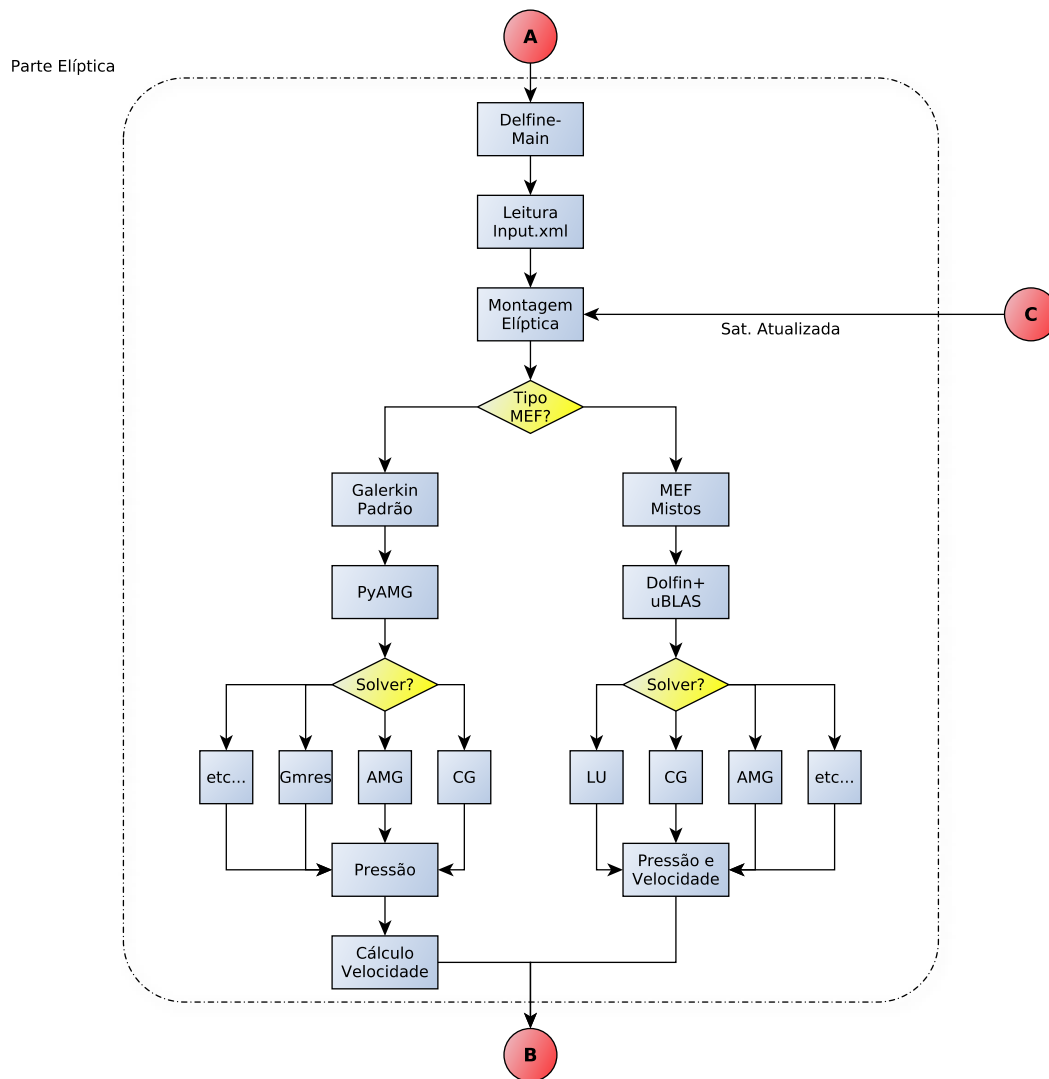
4.3.1 FEniCS

O *FEniCS* é um projeto colaborativo em código aberto iniciado em 2003 com o objetivo de automatizar a solução de modelos matemáticos baseados em equações diferenciais (Logg *et al.*, 2011), tendo todos os seus componentes sido desenvolvidos buscando generalidade, eficiência e simplicidade.

De um modo geral, o desenvolvedor tem acesso direto principalmente ao *DOLFIN*, o qual é uma biblioteca que permite a interface com o usuário através de diversas classes acessíveis via programas em *C++* ou em *Python*. Para facilitar a compreensão, a Fig. 4.4 apresenta de modo esquemático a sequência na qual os diversos componentes do projeto *FEniCS* são executados, e como eles interagem para permitir a resolução do problema.

Inicialmente, o problema tem que ser descrito matematicamente na sua forma variacional (ou fraca). Em seguida, esta deve ser implementada utilizando a *Unified Form Language* (UFL) (Logg *et al.*, 2011), a qual é uma linguagem específica de domínio para declaração da discretização via MEF de formas variacionais e funcionais. Para o caso de programas escritos em *Python*, a descrição via UFL é embutida dentro do próprio script, já no caso do *C++* é necessário criar um arquivo externo para definição da forma fraca do problema, sendo ele importado para o programa principal.

Em seguida, tais formas são compiladas utilizando o *FEniCS Form Compiler* (FFC) (Logg *et al.*, 2011), o qual é o responsável de fato pela geração automática do código otimizado em linguagem de baixo-nível (quando comparada à utilizada para a descrição do problema). Este código estará automaticamente conforme o padrão do *Unified Form-*

**Figura 4.2** Fluxograma da resolução da parte elíptica.

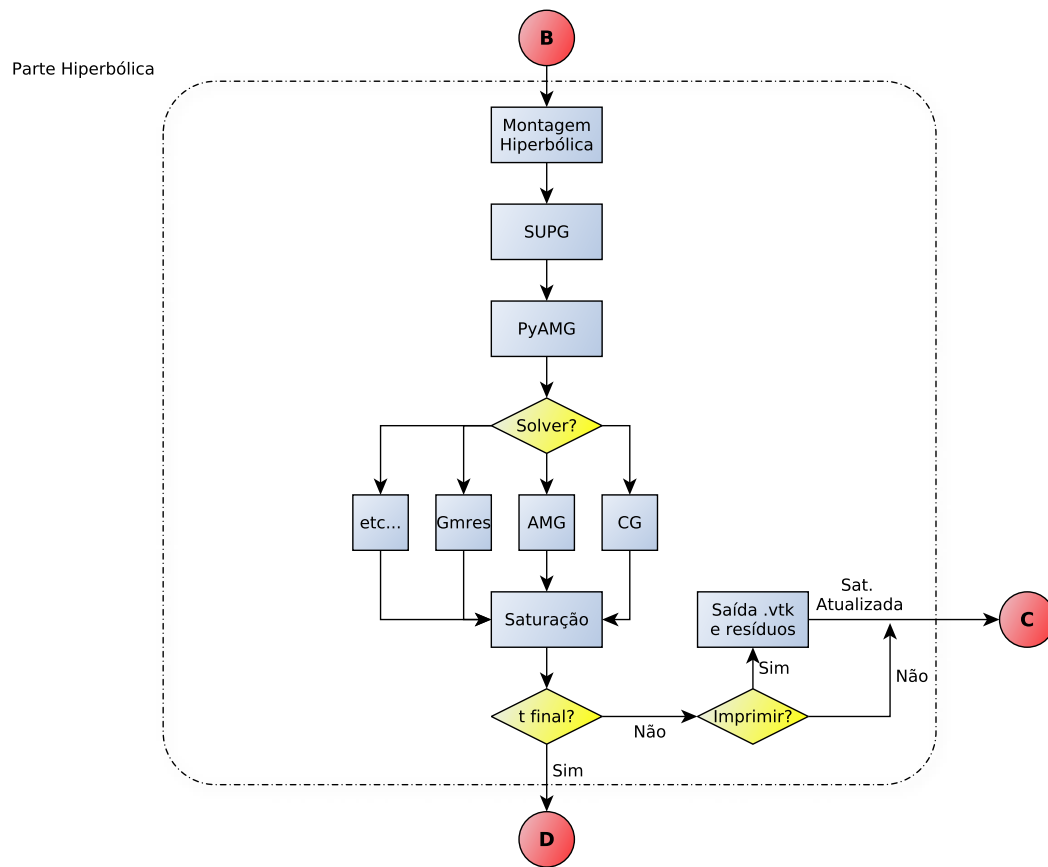


Figura 4.3 Fluxograma da resolução da parte hiperbólica.

assembly Code (UFC) e pode ser acessado de modo transparente através de classes da biblioteca *DOLFIN*, a qual será responsável pela montagem de todos os tensores necessários para a resolução numérica do problema dentro do programa definido pelo usuário (*Delfine* no caso deste trabalho, como representado na Fig. 4.4).



Figura 4.4 Interação entre os diversos componentes do projeto *FEniCS* para definição do problema, seguidos pela resolução no *Delfine* (adaptado de Rathgeber (2010)).

O *DOLFIN* (Logg e Wells, 2010) automatiza a montagem dos sistemas lineares ou não-lineares provenientes da discretização via MEF de EDPs expressas na forma variacional. Na Fig. 4.5 é apresentada a estrutura modular da biblioteca, onde os dados de entrada fornecidos pelo usuário para um problema específico são a malha, a forma variacional e os tipos de elementos finitos adotados. De posse destes dados, o *DOLFIN* gera as matrizes e vetores necessários para resolução do sistema de equações provenientes da discretização. Como mostrado na figura, o *DOLFIN* contém interfaces para as bibliotecas de álgebra linear computacional *PETSc*, *Epetra*, *uBLAS* e *MTL4*, além de permitir que o usuário utilize um outro programa qualquer, desde que o mesmo tenha suporte para matrizes no formato do *SciPy* (Jones *et al.*, 2001), como é o caso do *PyAMG* usado neste trabalho.

O *DOLFIN* permite a utilização de alguns operadores básicos na definição da forma variacional, como os de adição ($v+w$), multiplicação ($v*w$), diferenciação ($v \cdot dx(i)$) e integração ($v*dx$). Além disso, e neste ponto reside o grande diferencial desta biblioteca em relação as outras existentes, é possível definir de modo bastante direto operações complexas que nada mais são do que composições dos operadores básicos citados. Entre estas operações podemos citar o produto interno (`dot()`), produto vetorial (`cross()`), divergente (`div()`), gradiente (`grad()`), rotacional (`curl()`), entre outros.

Portanto, um dos grandes atrativos do *DOLFIN* é a possibilidade de utilizar uma notação muito próxima da matemática para a escrita do código. De modo a exemplificar isto, serão apresentados a seguir os trechos de código correspondentes às formulações do MEF de Galerkin, do MEFM e do MEF com estabilização via SUPG discutidos nas seções 3.1, 3.2 e 3.3, respectivamente.

A formulação variacional da equação da pressão utilizando o MEF de Galerkin é dada por:

$$A(p, w) = f(w) \quad \forall w \in W \quad (4.1)$$

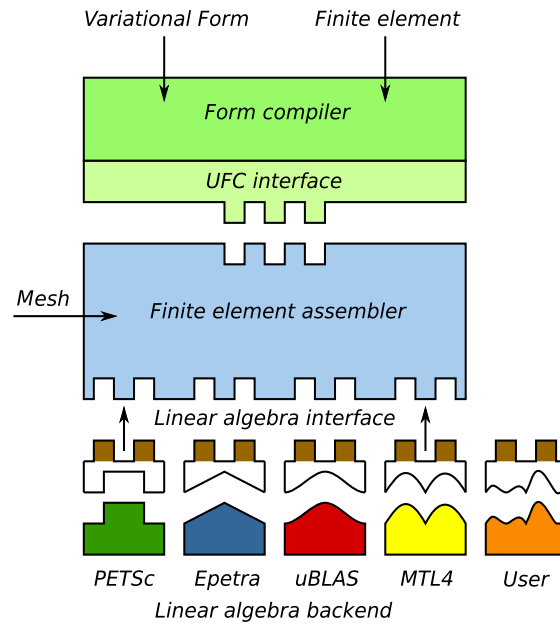


Figura 4.5 Estrutura modular do *DOLFIN* (retirado de Logg e Wells (2010)).

onde as formas bilinear e linear, respectivamente, são:

$$A(p, w) = \int_{\Omega} \nabla w \cdot \lambda K \nabla p d\Omega \quad (4.2)$$

$$f(w) = \int_{\Omega} w f d\Omega - \int_{\Gamma} g w d\Gamma \quad (4.3)$$

que representam a notação matemática cuja codificação em *Python* pode ser vista na Listagem 2.

```

Python
def Galerkin(self, defineVar, parameter):
    .
    .
    # Define variational form
    a = inner(grad(w), K*mob*grad(p))*dx
    L = w*f*dx - g*w*ds

```

Listagem 2: Codificação em *Python* da montagem do operador elíptico via MEF de Galerkin usando o *FEniCS/DOLFIN*.

Já a formulação variacional para a equação que resolve simultaneamente a pressão e a velocidade utilizando o MEFM é representada pelas seguintes formas bilinear e linear,

respectivamente:

$$A(p, w, \mathbf{v}, \zeta) = \int_{\Omega} (K^{-1} \lambda^{-1} \mathbf{v} \cdot \zeta - \nabla \cdot \zeta p - \nabla \cdot \mathbf{v} w) d\Omega \quad (4.4)$$

$$f(w) = - \int_{\Omega} f w d\Omega \quad (4.5)$$

que representam a notação matemática cuja codificação em *Python* pode ser vista na Listagem 3.

```

Python
def MixedFEM(self, defineVar, parameter):
    .
    .
    # Define variational form
    a = (dot((invK/mob)*v, zeta)
        - div(zeta)*p - div(v)*w)*dx
    L = - f*w*dx

```

Listagem 3: Codificação em *Python* da montagem do operador elíptico via MEFM usando o *FEniCS/DOLFIN*.

A formulação variacional para a equação da saturação utilizando o MEF com estabilização via SUPG e termo de captura de choque para adição de difusão artificial é representada matematicamente da seguinte forma:

$$F(S, w) = \int_{\Omega} \left(w \phi \frac{S^{n+1} - S^n}{\Delta t} \right) d\Omega + \int_{\Omega} w f_{,S} \mathbf{v}^{n+1} \cdot \nabla S^{n+1} d\Omega + \sum_{E_i} \int_{E_i} (\mathbf{v}^{n+1} \cdot \nabla w) \tau_s r^{n+1} d\Omega + \sum_{E_i} \int_{E_i} v_{shock} \nabla w \cdot \nabla S^{n+1} d\Omega \quad (4.6)$$

onde

$$r^{n+1} = \phi \frac{S^{n+1} - S^n}{\Delta t} + f_{,S} \mathbf{v}^{n+1} \cdot \nabla S^{n+1} \quad (4.7)$$

com

$$\tau_s = \frac{h}{2 \|\mathbf{v}^{n+1}\|} \quad (4.8)$$

e

$$v_{shock} = \begin{cases} \frac{\beta h |r^{n+1}|}{2 \|\nabla S^n\|} & \text{se } \|\nabla S^n\| \neq 0 \\ 0 & \text{caso contrário} \end{cases} \quad (4.9)$$

A codificação em *Python* deste método pode ser vista na Listagem 4.

```

Python
def SUPG(self, delfineVar, parameter):
    .
    .
    # Galerkin variational problem
    F = w*phi*((S-S0)/dt)*dx +
        (w*fs*dot(v, grad(S))*dx
    # Residual
    r = phi*((S-S0)/dt) + fs*(dot(v, grad(S)))
    # SUPG stabilization term
    tau = h/(2.0*sqrt(dot(v, v)))
    F += tau*dot(v, grad(w))*r*dx
    # Add shock capturing term
    beta = 2.0
    snorm = sqrt(dot(grad(S0), grad(S0)))
    tol = 1E-15
    if (abs(snorm) > tol):
        vshock = (beta*h*abs(r))/(2*snorm)
    else:
        vshock = 0.0
    F += vshock*dot(grad(w), grad(S))*dx
    # Create bilinear and linear forms
    a = lhs(F)
    L = rhs(F)

```

Listagem 4: Codificação em *Python* da montagem do operador hiperbólico via MEF com estabilização via SUPG usando o *FEniCS/DOLFIN*.

Como pôde ser visto, o *DOLFIN* permite tanto a definição direta das formas bilineares e lineares, quanto a definição de um funcional e posterior subdivisão dele através dos comandos `rhs()` e `lhs()`. Uma vez obtidas as formas que descrevem o problema, o sistema de equações é montado através de um simples comando, como exemplificado na Listagem 5.

```

Python
# Assemble system
A, rhs = assemble_system(a, L)
bc.apply(A, rhs)

```

Listagem 5: Montagem do sistema de equações no formato matricial e aplicação das condições de contorno usando o *FEniCS/DOLFIN*.

Para a resolução do sistema de equações proveniente das matrizes e vetores resultantes da discretização foram utilizadas as bibliotecas de álgebra linear *PyAMG* (prioritariamente) e *uBLAS* (para matrizes provenientes do MEFM). Como já mencionado, a interface desta última biblioteca com o *DOLFIN* é implementada por default, logo o foco neste capítulo será na interface com o *PyAMG*, o qual é descrito com mais detalhes na seção a seguir.

4.3.2 PyAMG

O *PyAMG* (Bell *et al.*, 2008) é uma coleção de resolvedores de equações lineares com uma interface em *Python*. Diversas variações do método AMG estão implementadas no *PyAMG*, como o AMG clássico, o *smoothed aggregation* (SA) e o *adaptive smoothed aggregation*. Por ter sido escrito de maneira modular, o *PyAMG* se apresenta como uma excelente ferramenta para prototipagem rápida de métodos multigrid. Além disso, por ter suas operações de maior custo computacional compiladas em C++ é possível a resolução eficiente de problemas de grande escala.

A listagem de código 6 apresenta um exemplo típico de utilização do *PyAMG* em conjunto com o *DOLFIN*. A comunicação das matrizes e vetores provenientes da montagem no *DOLFIN* é feita através do formato *SciPy*. Em seguida, tais dados são repassados em conjunto com alguns parâmetros numéricos para a função do *PyAMG* responsável pela resolução do sistema de equações. No exemplo mostrado, se utilizou um resolvidor do tipo Método dos Gradientes Conjugados (CG) com condicionamento via AMG. Os parâmetros passados foram o número máximo de níveis menos refinados (ver seção 3.4 para mais detalhes) e a tolerância considerada. A grande vantagem do uso desta biblioteca é justamente a possibilidade de testar diversas possibilidades apenas com a mudança destes parâmetros de entrada, já que seria perfeitamente possível trocar o resolvidor do tipo CG para um *Generalized Minimal Residual Method* (GMRES), caso se esteja trabalhando com matrizes não-simétricas. Outra flexibilidade permitida é a de alterar o tipo de ciclo multigrid utilizado, sendo o tipo *V* considerado default caso nenhuma

informação seja fornecida, como no caso do exemplo utilizado.

```

Python
def solve_withPyAMG(self, delfineVar, parameter):
    # Getting data from elliptic eq. assembler
    A = delfineVar.A
    rhs = delfineVar.rhs
    # Get sparse matrix data
    (r,c,data) = A.data()
    n = A.size(0)
    # Matrix in scipy/numpy format
    As = csr_matrix((data,c.view(),r.view()),shape=(n,n))
    # Get right-hand side vector(rhs) data
    b = rhs.data()
    res = []
    # Solve with AMG as preconditioner for the CG Method
    ml = smoothed_aggregation_solver(As,max_coarse=10)
    x = ml.solve(b,tol=1e-10,accel='cg',residuals=res)

```

Listagem 6: Leitura da matriz e vetor que representam o sistema de equações e resolução do mesmo utilizando o *PyAMG*.

4.4 Pós-Processamento

Uma vez finalizada a etapa de processamento, é necessário analisar os resultados obtidos. Em termos de implementação computacional esta é a fase que envolve menos componentes, como pode ser visto na Fig. 4.6.

O *DOLFIN* dispõe de funções para impressão em formato XML dos dados armazenados (como malhas, propriedades, etc.) e resultados gerados durante a análise. Esta funcionalidade pode ser utilizada de modo bastante direto no programa devido ao uso de operadores sobrecarregados. A Listagem 7 apresenta um exemplo de impressão do campo de saturação S em um passo de tempo t . O formato utilizado (`.vtk`) nada mais é do que um arquivo XML formatado para seguir um padrão definido que pode ser lido por vários programas de visualização. Neste trabalho foi utilizado prioritariamente o *Paraview* (Henderson, 2007), porém o arquivo também poderia ter sido visualizado em outros programas de visualização, como o *VisIt* ou o *TecPlot*. Além disso, o *DOLFIN*

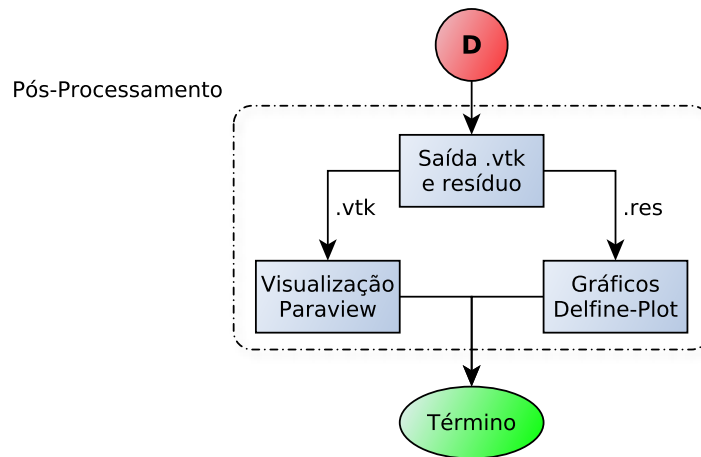


Figura 4.6 Fluxograma da etapa de pós-processamento.

permite a visualização rápida de resultados durante a própria simulação utilizando uma ferramenta própria chamada *Viper*.

Python

```

# Output file
out_file = File("Results/supg_saturation.vtk")
# Save the saturation solution to file
out_file << (S, t)
  
```

Listagem 7: Impressão dos resultados em arquivos do tipo XML usando o *FEniCS/DOL-FIN*.

Além da impressão dos arquivos com os resultados associados aos campos de pressão, velocidade e saturação, foi implementado neste trabalho um script em *Python* para comparação da performance de diversos métodos de resolução do sistema de equações (AMG, CG, GMRES, AMG+CG e AMG+GMRES). Este script lê automaticamente todos os resíduos gerados pelos métodos desejados e cria gráficos semi-logarítmicos para visualização da evolução do resíduo em comparação ao número de iterações.

Capítulo 5

Resultados

Neste capítulo serão apresentados exemplos e resultados obtidos utilizando a metodologia e o programa computacional descritos nos capítulos anteriores.

5.1 Problemas Elípticos

Esta seção tem por objetivo avaliar os métodos utilizados para a resolução da equação da pressão em meios porosos, a qual para o estado estacionário pode ser considerada uma típica equação elíptica. Devido a essa característica, a metodologia utilizada pode ser analisada utilizando-se de problemas modelos que compartilham das mesmas propriedades matemáticas. Uma particularidade comum aos problemas regidos por equações elípticas é que todo o domínio de interesse Ω é afetado por qualquer mudança no valor da variável em um ponto qualquer no interior de Ω , ou em sua fronteira Γ (Fortuna, 2000). Isto significa que uma perturbação em um ponto irá ter influência sobre todo o domínio, sendo que a mesma diminui com o aumento da distância em relação ao ponto originador de tal perturbação. Sendo assim, tais problemas tendem, em geral, a apresentar soluções suaves ao longo do domínio. Entretanto, é importante mencionar que na área da simulação de escoamentos em meios poroso é comum o aparecimento de singularidades ou gradientes elevados no campo de pressão, devido, por exemplo, a existência de poços nodalmente aplicados, de grandes discontinuidades na permeabilidade ou de mudanças nas condições de contorno. Além da equação de pressão, outros exemplos de aplicação deste tipo de equação são o cálculo do potencial elétrico, da difusão de calor em uma chapa metálica e de escoamentos incompressíveis, invíscidos e irrotacionais (também conhecidos como escoamentos potenciais).

A equação normalmente utilizada como modelo para testar metodologias de resolução

de equações elípticas pode ser descrita como:

$$\nabla(\mathbf{K}\nabla u) = f \quad (5.1)$$

Em coordenadas cartesianas e considerando um coeficiente anisotrópico e variável no espaço $\mathbf{K}(x, y, z)$, esta equação pode ser representada por:

$$\frac{\partial}{\partial x} \left(\mathbf{K} \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left(\mathbf{K} \frac{\partial u}{\partial y} \right) + \frac{\partial}{\partial z} \left(\mathbf{K} \frac{\partial u}{\partial z} \right) = f(x, y, z) \quad (5.2)$$

Nas seções seguintes, iremos resolver equações do tipo (5.2) considerando diversas possibilidades para o coeficiente $\mathbf{K}$ de modo a demonstrar a flexibilidade da metodologia utilizada para lidar com meios homogêneos, heterogêneos, iso- e anisotrópicos. Vários parâmetros de interesse serão avaliados a partir dos resultados obtidos, entre eles:

- Erro da solução numérica quando comparada com a solução analítica, a qual para problemas simples é facilmente encontrada.
- Taxas de convergências para uma sequência de malhas sucessivamente refinadas.
- Evolução do resíduo para diferentes estratégias de resolução do sistema de equações lineares.

5.1.1 Meio Homogêneo e Isotrópico

Este primeiro exemplo foi originalmente proposto em (Chen *et al.*, 2006), sendo resolvido utilizando dois métodos diferentes, o CVFA (*Control Volume Function Approximation*) e o CVFE (*Control Volume Finite Element*). Este problema também foi explorado em (Silva, 2008) utilizando duas variações do Método dos Volumes Finitos baseado em Arestas (*Edge-Based Finite Volume* - EBFV). Este exemplo é um problema de valor de contorno que pode ser representado do modo a seguir:

$$\nabla(\mathbf{K}\nabla u) = 2\pi^2 \cos(\pi x) \cos(\pi y) \quad \text{em} \quad \Omega = \{(x, y) \mid 0 < x < 1 \text{ e } 0 < y < 1\} \quad (5.3)$$

onde $\mathbf{K}$ é uma matriz simétrica e diagonal representada por:

$$\mathbf{K} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad (5.4)$$

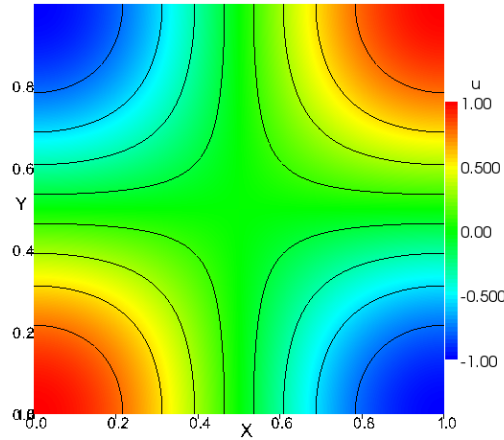


Figura 5.1 Campo escalar obtido utilizando o MEF com malha de 64×64 discretizada por triângulos lineares.

Este problema apresenta condições de contorno periódicas nas fronteiras inferior e superior, já as fronteiras laterais estão sujeitas a uma condição de fluxo zero. Matematicamente tais condições de contorno podem ser definidas do seguinte modo:

$$\begin{aligned} u &= \cos(\pi x) && \text{para } 0 < x < 1 \text{ e } y = 0 \\ u &= -\cos(\pi x) && \text{para } 0 < x < 1 \text{ e } y = 1 \\ \nabla u \cdot \vec{n} &= 0 && \text{para } 0 < y < 1 \text{ e } x = 0, 1 \end{aligned} \quad (5.5)$$

Resolvemos o problema elíptico descrito utilizando a formulação detalhada no capítulo 3, considerando diferentes malhas estruturadas sucessivamente refinadas e formadas por elementos triangulares de 1ª e 2ª ordem. Foram utilizadas malhas estruturadas apenas para fins de comparação de erros e obtenção de ordens de convergência, já que a metodologia utilizada é perfeitamente capaz de lidar com malhas não-estruturadas bi- e tridimensionais. Para avaliação da acurácia do método utilizado, os resultados foram comparados com a solução analítica deste problema, a qual é representada pela função $u(x, y) = \cos(\pi x) \cos(\pi y)$. A Fig. 5.1 mostra o campo escalar u obtido para a malha de 64×64 elementos triangulares lineares, a qual já permite obter uma excelente concordância com a solução analítica.

Para uma sequência de malhas formadas por triângulos lineares, se espera uma taxa de convergência de segunda ordem, enquanto para uma sequência formada por triângulos

Tabela 5.1 Erro e taxa de convergência obtidos neste trabalho para a solução da equação elíptica em meio homogêneo e isotrópico.

(a) Triângulo Linear			(b) Triângulo Quadrático		
N	$ E_{max} $	q_{max}	N	$ E_{max} $	q_{max}
8	2.5e-02	—	8	6.4e-04	—
16	6.4e-03	1.97	16	8.7e-05	2.87
32	1.6e-03	1.99	32	1.1e-05	2.94
64	4.0e-04	1.99	64	1.4e-06	2.98
128	1.0e-04	2.00	128	1.8e-07	2.98
256	2.5e-05	1.99	256	2.2e-08	3.00

Tabela 5.2 Erro e taxa de convergência obtidos em Chen *et al.* (2006) e Silva (2008) para a solução da equação elíptica em meio homogêneo e isotrópico.

(a) CVFA(Chen <i>et al.</i> , 2006)			(b) EBFV1(Silva, 2008)		
N	$ E_{rms} $	q_{rms}	N	$ E_{rms} $	q_{rms}
8	1.2e-02	—	8	6.9-03	—
16	3.0e-03	2.02	16	1.4e-03	2.29
32	7.4e-03	2.01	32	3.2e-04	2.13
64	1.8e-04	2.00	64	7.7e-05	2.06

quadráticos espera-se uma convergência de terceira ordem (Hughes, 2000). De modo a verificar o comportamento do método utilizado, resolvemos o problema proposto utilizando elementos de 1ª (Tabela 5.1(a)) e 2ª ordem (Tabela 5.1(b)), comparando entre si os resultados obtidos e também com outros disponíveis na literatura para elementos lineares utilizando outros tipos de formulação (Tabelas 5.2(a) e 5.2(b)) (Chen *et al.*, 2006; Silva, 2008). De um modo geral, as taxas de convergências obtidas se aproximaram bastante das taxas teóricas tanto para o caso de elementos triangulares lineares quanto quadráticos. A comparação com os resultados da literatura ficam um pouco prejudicadas por terem os erros sido calculados através de normas diferentes (norma do máximo neste trabalho, e RMS na literatura citada). Porém, é possível observar um comportamento coerente do erro obtido, pois com elementos quadráticos foram obtidos erros consideravelmente menores do que com os métodos de segunda ordem CVFA e EBFV1. Por outro lado, ao utilizar-se elementos lineares foram obtidos erros maiores porém na mesma ordem de grandeza daqueles obtidos na literatura.

Para a resolução do sistema de equações lineares provenientes da discretização do problema foram testadas três diferentes alternativas:

- Método dos Gradientes Conjugados (CG) aplicado isoladamente.

- Método do Multigrid Algébrico (AMG) aplicado isoladamente.
- Multigrid Algébrico aplicado como pré-condicionador para o método dos Gradientes Conjugados (AMG+CG).

É importante mencionar que a utilização de métodos iterativos normalmente pressupõe o uso de algum tipo de pré-condicionador, logo a não aplicação de pré-condicionadores em alguns dos testes realizados teve como objetivo apenas realçar as diferenças entre os métodos analisados.

Para a realização dos testes, foi utilizada uma malha de 32×32 elementos triangulares de 1ª ordem. Para todos os casos foi considerado como critério de parada uma tolerância $\varepsilon = 10^{-10}$, com um número máximo de 200 iterações. Para os casos nos quais o método AMG foi utilizado como forma de acelerar a convergência, considerou-se o uso de ciclos do tipo V em uma hierarquia de 4 “malhas” sucessivamente menos refinadas. É importante apenas lembrar que o método AMG não depende de malhas propriamente ditas, apenas das matrizes que representam o sistema de equações, sendo tais matrizes manipuladas algebricamente de modo a se obter os níveis menos refinados (Trottenberg *et al.*, 2001).

Como pode ser observado na Fig. 5.2, os esquemas utilizando o método Multigrid (AMG e AMG+CG) apresentaram resultados bastante superiores em relação ao método CG, o qual utiliza apenas um único sistema de equações lineares (ou seja, uma única “malha”). Enquanto o método AMG+CG reduz o resíduo em aproximadamente uma ordem de grandeza por iteração, o método CG necessita de aproximadamente 10 vezes mais iterações para reduzir o resíduo na mesma proporção. É interessante comparar também os resultados obtidos ao utilizar-se o AMG isoladamente e ao usá-lo como um pré-condicionador para o método dos Gradientes Conjugados. Percebe-se que nas primeiras iterações os dois métodos apresentam fatores de convergência para o resíduo praticamente idênticos, porém a medida que se prossegue com as iterações, o AMG usado isoladamente tende a apresentar uma piora na taxa de convergência, enquanto o método que utiliza a combinação AMG+CG mantém o excelente fator de convergência obtido inicialmente.

Isto se deve a um aspecto bastante observado na literatura (Trottenberg *et al.*, 2001; Oosteele e Washio, 1998; Mavriplis, 2001) no qual um único ou poucos autovalores da matriz representante do operador de iteração Multigrid estão com um valor muito acima daquele obtido para os demais autovalores. A magnitude do autovalor máximo (também chamado de raio espectral) do operador de iteração indica em última análise o fator de convergência assintótica para o problema (Briggs *et al.*, 2000), limitando portanto a taxa

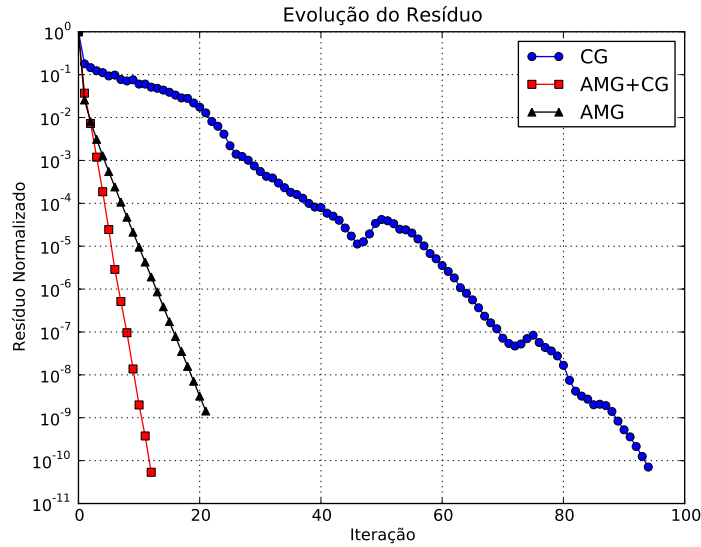


Figura 5.2 Caso homogêneo e isotrópico. Comparação da evolução dos resíduos para malha 32×32 discretizada por triângulos lineares.

de convergência possível de ser obtida. Um modo de contornar tal problema é justamente utilizar algum dos métodos de subespaço de Krylov, como é o caso do método CG, os quais têm a característica de em poucas iterações amortecer os autovetores relacionados aos poucos autovalores elevados, pois usam subespaços de menor dimensão (Trottenberg *et al.*, 2001).

5.1.2 Meio Homogêneo e Anisotrópico

Este problema foi originalmente proposto em Crumpton (1995), onde foi resolvido utilizando um método de volumes finitos centrado na célula (*Flux Continuous Finite Volume - FCFV*), tendo também sido estudado em Carvalho (2005) e Silva (2008) utilizando o já mencionado método dos volumes finitos baseado em arestas (EBFV).

Neste segundo exemplo, buscamos explorar a flexibilidade da metodologia utilizada, a qual permite considerar tensores não-diagonais para representar o parâmetro $\mathbf{K}$. O problema proposto pode ser descrito através da Eq. (5.6) aplicada em um domínio $\Omega = (0, 1) \times (0, 1)$.

$$\nabla(\mathbf{K}\nabla u) = -2(1 + x^2 + xy + y^2)e^{xy} \quad (5.6)$$

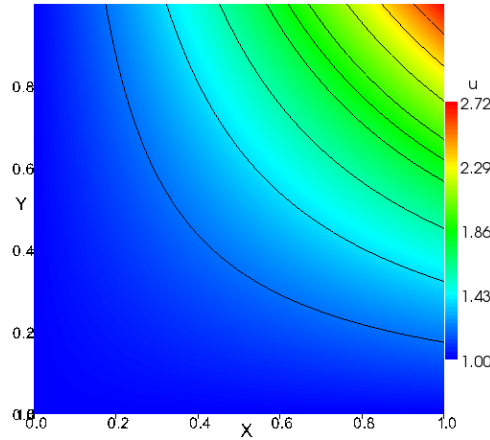


Figura 5.3 Caso homogêneo e anisotrópico. Campo escalar obtido utilizando o MEF com malha de 64×64 discretizada por triângulos lineares.

onde $\mathbf{K}$ é uma matriz simétrica e não-diagonal representada por:

$$\mathbf{K} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \quad (5.7)$$

A descrição deste problema está completa ao definir-se a condição de fronteira em todo o contorno Γ como $u(x, y) = e^{xy}$.

Na Fig. 5.3 pode ser observado o campo escalar para a variável u , considerando uma discretização do domínio Ω com uma malha de 64×64 elementos triangulares de 1ª ordem.

De modo a comparar os resultados obtidos com aqueles da literatura, calculamos o erro segundo a norma do máximo e as taxas de convergências para o mesmo considerando uma sequência de malhas sucessivamente refinadas (ver Tabela 5.3(a)). Tais resultados foram comparados com aqueles obtidos tanto pelo FCFV quanto pelo EBFV (ver Tabelas 5.4(a) e 5.4(b)).

Pode-se observar pelos resultados obtidos que novamente o MEF se comportou como esperado, apresentado uma taxa de convergência de ordem 2 para os elementos lineares. Os valores absolutos do erro na norma do máximo se comparam de maneira ligeiramente favorável àqueles apresentados em Carvalho (2005).

Assim como para o primeiro exemplo, também neste fizemos um estudo a respeito do

Tabela 5.3 Erro e taxa de convergência obtidos neste trabalho para a solução da equação elíptica em meio homogêneo e anisotrópico.

(a) Triângulo Linear

N	$\ E_{max}\ $	q_{max}
8	2.1e-03	—
16	5.3e-04	2.01
32	1.3e-04	2.00
64	3.3e-05	1.99
128	8.3e-06	2.00
256	2.1e-06	2.00

Tabela 5.4 Erro e taxa de convergência obtidos em Crumpton (1995) e Carvalho (2005) para a solução da equação elíptica em meio homogêneo e anisotrópico.

(a) FCFV(Crumpton, 1995)

N	$\ E_{rms}\ $	q_{rms}
8	1.2e-03	—
16	2.9e-04	2.00
32	7.3e-05	1.99
64	1.8e-05	1.99

(b) EBFV(Carvalho, 2005)

N	$\ E_{max}\ $	q_{max}
8	3.0e-03	—
16	1.0e-03	1.58
32	3.4e-04	1.80
64	8.9e-05	1.94

desempenho de diferentes métodos iterativos para a resolução do sistema de equações lineares provenientes da discretização do problema. Cinco diferentes alternativas foram testadas:

- Método dos Gradientes Conjugados (CG) aplicado isoladamente.
- Método do Resíduo Mínimo Generalizado (GMRES) aplicado isoladamente.
- Método do Multigrid Algébrico (AMG) aplicado isoladamente.
- Multigrid Algébrico aplicado como pré-condicionador para o método dos Gradientes Conjugados (AMG+CG).
- Multigrid Algébrico aplicado como pré-condicionador para o método do Resíduo Mínimo Generalizado (AMG+GMRES).

É importante mencionar que o método GMRES é normalmente utilizado para problemas envolvendo matrizes não-simétricas (Saad, 2003), tendo sido avaliado apenas para verificar a flexibilidade do resolvido de sistemas de equações lineares, já que a

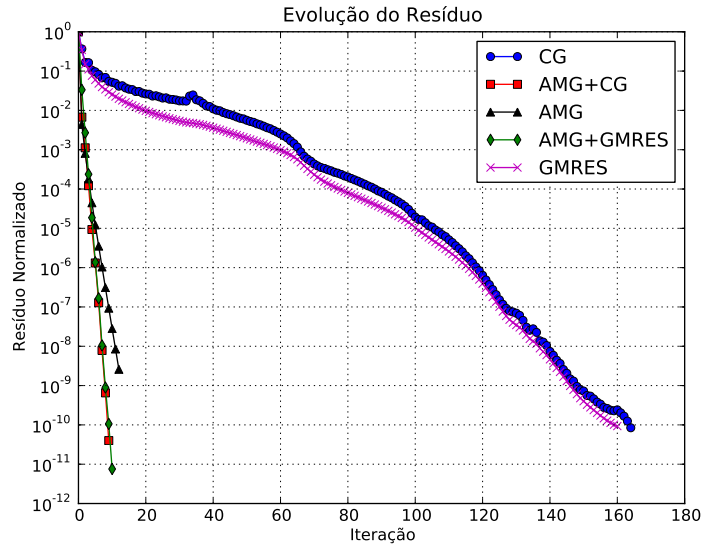


Figura 5.4 Caso homogêneo e anisotrópico. Comparação da evolução dos resíduos para malha 64×64 discretizada por triângulos lineares.

discretização deste problema resulta em uma matriz auto-adjunta (e portanto simétrica se considerarmos apenas termos reais).

Para a realização dos testes, foi utilizada uma malha de 64×64 elementos triangulares de 1ª ordem. Para todos os casos foi considerada como critério de parada uma tolerância $\varepsilon = 10^{-10}$, com um número máximo de 200 iterações. Em todos os casos nos quais o método AMG foi utilizado como forma de acelerar a convergência, considerou-se o uso de ciclos do tipo V em uma hierarquia de 4 “malhas” sucessivamente menos refinadas. Um gráfico mostrando a evolução do resíduo normalizado para todos os métodos testados pode ser observado na Fig. 5.4, onde fica claro novamente a grande superioridade dos métodos que utilizam aceleração de convergência via Multigrid em comparação com os outros métodos. É interessante notar também que para este exemplo os métodos CG e GMRES tiveram desempenhos bastante próximos, tanto isoladamente quanto em conjunto com o AMG usado como pré-condicionador.

5.1.3 Meio Heterogêneo e Anisotrópico

Neste exemplo foi considerado um meio heterogêneo e anisotrópico. Este caso foi proposto em Crumpton (1995), tendo sido também analisado em vários outros trabalhos (Hyman *et al.*, 1997; Carvalho, 2005; Silva, 2008). O domínio Ω é um quadrado $[-1, 1] \times [-1, 1]$ com condições de contornos de Dirichlet dadas pela solução exata. O

parâmetro $\mathbf{K}$ é definido por:

$$\mathbf{K} = \begin{cases} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \text{se } x \leq 0 \\ \psi \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} & \text{se } x > 0 \end{cases} \quad (5.8)$$

onde o fator ψ é um parâmetro utilizado para definir a intensidade de descontinuidade em $x = 0$.

A solução analítica (e condição de Dirichlet para os contornos) é dada por:

$$u(x, y) = \begin{cases} [2\sin(y) + \cos(y)]\psi x + \sin(y) & \text{se } x \leq 0 \\ \psi \exp(x) \sin(y) & \text{se } x > 0 \end{cases} \quad (5.9)$$

Por fim, podemos definir o também descontínuo termo fonte como:

$$f(x, y) = \begin{cases} [-2\sin(y) - \cos(y)]\psi x - \sin(y) & \text{se } x \leq 0 \\ 2\psi \exp(x) \cos(y) & \text{se } x > 0 \end{cases} \quad (5.10)$$

Para a resolução deste problema e obtenção de taxas de convergência, consideramos uma sequência de malhas sucessivamente refinadas com dimensões aproximadas de (8×8) , (16×16) , (32×32) , (64×64) e (128×128) . A malha não-estruturada utilizada com aproximadamente 8×8 elementos e a divisão dos subdomínios em $x = 0$ pode ser observada na Fig. 5.5.

Pode ser observado na Fig. 5.6 o efeito da descontinuidade na linha $x = 0$, efeito este que se torna mais pronunciado ao aumentarmos o valor do parâmetro ψ . Os valores para os erros da solução numérica quando comparada com a analítica para as normas do máximo e L_2 são apresentados na Tabela 5.5, onde pode ser visto que um aumento na descontinuidade do tensor $\mathbf{K}$ para os diferentes sub-domínios causa um aumento nos valores absolutos dos erros obtidos.

Um fato interessante que pode ser constatado a partir dos valores obtidos para os erros da solução numérica obtida é que os mesmos apresentam uma taxa de convergência na norma L_2 em torno de 1 para $\psi = 1$ e 10, sendo que esta taxa se aproxima de 2 quando aumentamos o fator de descontinuidade ($\psi = 100$ e 1000). Ao compararmos tais resultados com a literatura (Crumpton, 1995; Hyman *et al.*, 1997; Carvalho, 2005; Silva, 2008) percebemos que os métodos apresentados nos referidos trabalhos que fazem uma avaliação local dos fluxos tendem a obter uma convergência de ordem 2, enquanto métodos que trabalham apenas com a variável nodal (neste caso, $u(x, y)$) apresentam

Tabela 5.5 Erro e taxa de convergência obtidos neste trabalho para a solução da equação elíptica em meio heterogêneo e anisotrópico.

(a) $\psi = 1$

N	$\ E_{max}\ $	q_{max}	$\ E_{L_2}\ $	q_{L_2}
8	5.4e-02	—	4.5e-02	—
16	2.7e-02	0.99	2.0e-02	1.1
32	1.3e-02	1.02	9.8e-03	1.05
64	6.6e-03	1.02	4.8e-03	1.02
128	3.2e-03	1.02	2.4e-03	1.00

(b) $\psi = 10$

N	$\ E_{max}\ $	q_{max}	$\ E_{L_2}\ $	q_{L_2}
8	9.2e-02	—	1.8e-01	—
16	3.8e-02	1.3	5.2e-02	1.78
32	1.9e-02	1.02	1.8e-02	1.52
64	9.2e-03	1.02	7.7e-03	1.24
128	4.6e-03	1.00	3.6e-03	1.08

(c) $\psi = 100$

N	$\ E_{max}\ $	q_{max}	$\ E_{L_2}\ $	q_{L_2}
8	8.6e-01	—	1.7e-00	—
16	2.9e-01	1.59	4.3e-01	1.98
32	8.8e-02	1.70	1.1e-01	1.98
64	2.5e-02	1.79	2.8e-02	1.94
128	7.2e-03	1.81	7.9e-03	1.84

(d) $\psi = 1000$

N	$\ E_{max}\ $	q_{max}	$\ E_{L_2}\ $	q_{L_2}
8	8.8e-00	—	1.7e+01	—
16	2.9e-00	1.58	4.3e-00	1.98
32	9.2e-01	1.68	1.1e-00	2.0
64	2.7e-01	1.74	2.7e-01	1.99
128	7.9e-02	1.79	6.8e-02	1.99

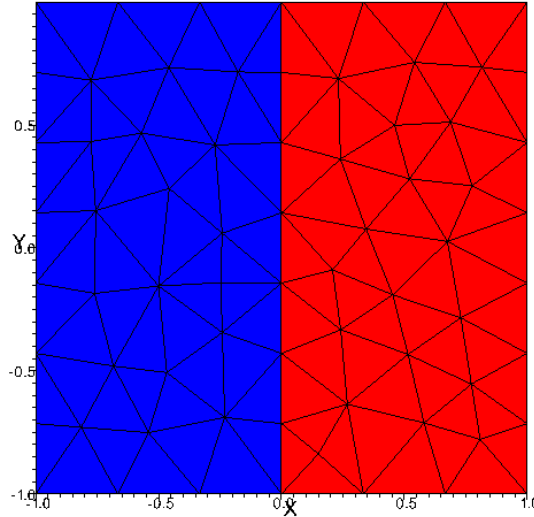


Figura 5.5 Domínio utilizado e malha não-estruturada com aproximadamente 8×8 elementos e heterogeneidade em $x = 0$.

ordem 1 para o erro, tanto na norma do máximo quanto para a L_2 .

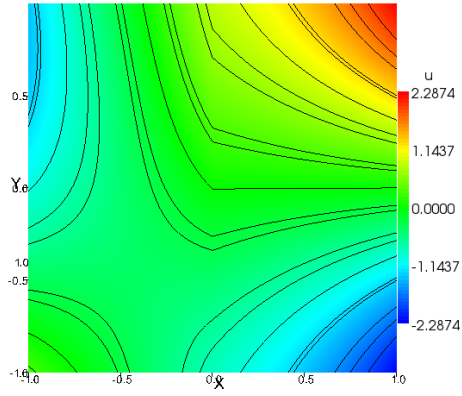
Para a resolução do sistema de equações lineares resultante da discretização foi novamente verificado o desempenho para diferentes tipos de métodos, mesmo sabendo de antemão que este problema é representado por uma matriz simétrica positiva-definida, sendo portanto um candidato natural a ser resolvido usando métodos do tipo gradiente conjugado, seja isoladamente ou com algum tipo de pré-condicionador como o Multigrid Algébrico (AMG).

Os resultados obtidos para a evolução dos resíduos podem ser verificados na Fig. 5.7, onde mais uma vez é constatada a superioridade dos métodos que utilizam Multigrid, mesmo para este problema que apresenta sub-domínios com propriedades diferentes.

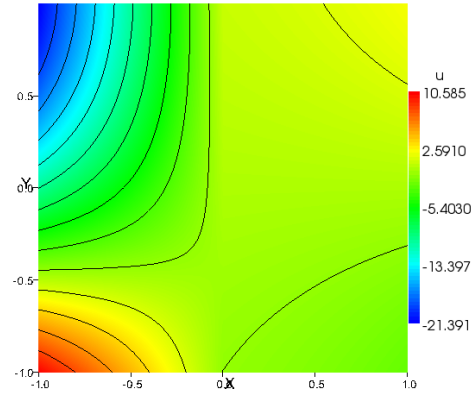
5.2 Escoamentos Bifásicos em Meios Porosos

5.2.1 Buckley-Leverett 1-D

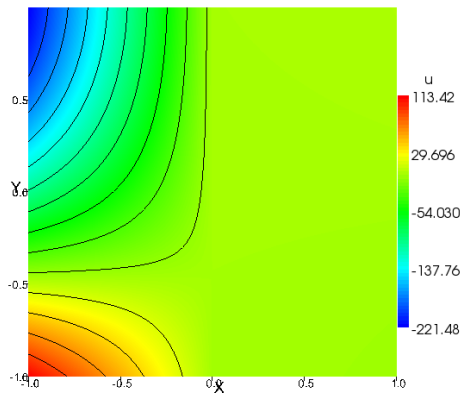
Este exemplo trata do deslocamento imiscível de óleo por água em um meio poroso rígido unidimensional e homogêneo, onde foram desprezados os efeitos da gravidade



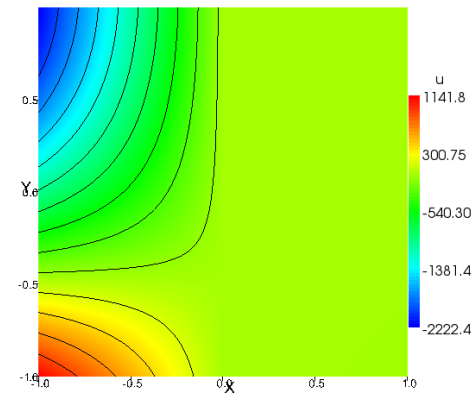
(a) $\psi = 1$



(b) $\psi = 10$



(c) $\psi = 100$



(d) $\psi = 1000$

Figura 5.6 Campo escalar obtido para diferentes fatores de descontinuidade ψ utilizando o MEF com malha de 64×64 discretizada por triângulos lineares.

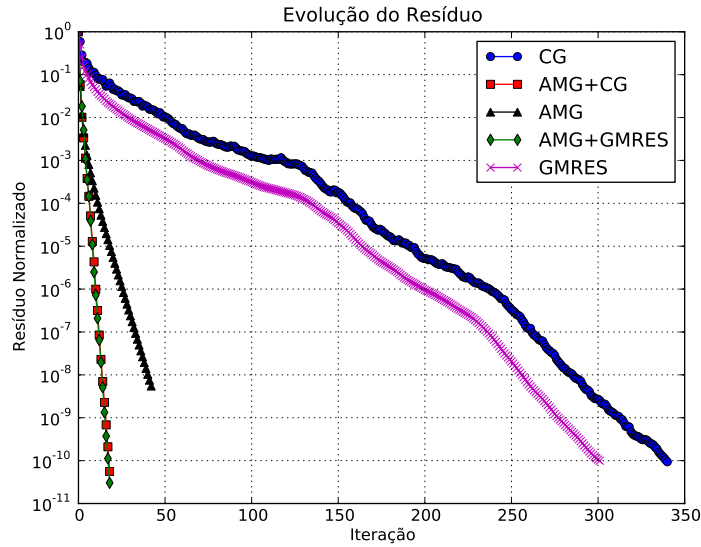


Figura 5.7 Caso heterogêneo e anisotrópico. Comparação da evolução dos resíduos para malha 64×64 discretizada por triângulos lineares e com $\psi = 1$.

e capilaridade (ver Fig. 5.8). O modelo para representar este fenômeno foi proposto originalmente por Buckley e Leverett (1942) e sua representação matemática está descrita na seção 2.4.1 deste trabalho.

Foi utilizado o modelo de Corey (descrito na seção 2.1) com coeficientes $n_w = n_o = 2$ para a dependência da permeabilidade relativa em relação à saturação. A saturação residual de água e óleo foi assumida como nula e a razão de viscosidade entre as fases como unitária. Para este exemplo consideramos um vetor velocidade $\mathbf{v}$ unitário e as

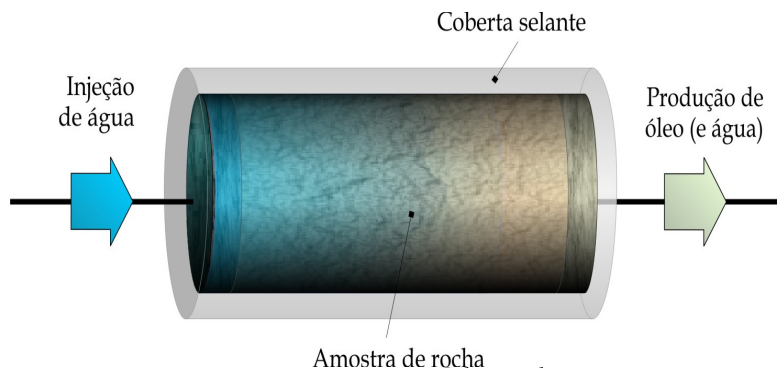


Figura 5.8 Representação de escoamento unidimensional imiscível de óleo por água (retirado de Hurtado (2005)).

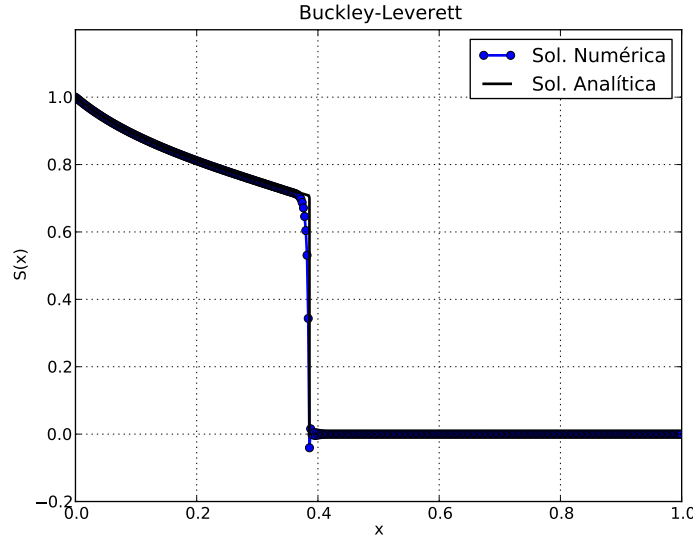


Figura 5.9 Perfil de saturação para a resolução da equação de Buckley-Leverett com MEF estabilizado via SUPG e com operador de captura de choque.

seguintes condições iniciais e de contorno:

$$\begin{aligned} S_w(x, 0) &= 0.0 & \text{em } \Omega \\ S_w(x, t) &= 1.0 & \text{em } x = 0 \end{aligned} \quad (5.11)$$

Devido às características essencialmente hiperbólicas deste problema, foi utilizado para sua solução o MEF com estabilização via SUPG e adição de viscosidade artificial para diminuir as oscilações. Na Fig. 5.9 são apresentadas as soluções analítica e numérica para um $t = 0.3$ em domínio com comprimento unitário e considerando uma malha com 500 subdivisões uniformemente distribuídas.

Como pode ser observado, na região de gradiente mais suave as soluções são praticamente coincidentes. Além disso, a posição do choque é corretamente capturada pela solução numérica. Entretanto, observa-se a existência de uma região de oscilação logo após o choque, mesmo ao se utilizar difusão artificial. Uma alternativa seria aumentar a quantidade de difusão adicionada, porém isto teria o efeito colateral de causar uma maior dispersão na região de choque, o que não é desejado. De um modo geral, consideramos os resultados satisfatórios, porém os mesmos são dependentes de termos para estabilização e captura de choque que contém parâmetros que devem ser calibrados para cada problema. Idealmente, tal calibração seria realizada de modo automático a partir de modelos pré-definidos para o comportamento destes parâmetros.

5.2.2 1/4 de Cinco Poços Heterogêneo

Este último exemplo considera o escoamento bifásico imiscível óleo-água em um meio poroso heterogêneo e bidimensional. Este problema foi adaptado de (Carvalho, 2005), e consiste em um domínio Ω quadrilateral com uma região de baixa permeabilidade entre os poços injetor e produtor, os quais se encontram nas diagonais inferior esquerda e superior direita, respectivamente. A Fig 5.10 mostra uma descrição das principais características e condições de contorno deste problema.

A saturação inicial considerada foi $S_w(\mathbf{x}, 0) = 0.0$ em Ω . Foi utilizado o modelo de Corey (descrito na seção 2.1) com coeficientes $n_w = n_o = 2$ para a dependência da permeabilidade relativa em relação à saturação. Tanto a porosidade ϕ quanto a viscosidade μ foram consideradas constantes e com valor unitário. As permeabilidades das regiões 1 e 2 de Ω são isotrópicas, porém distintas por 4 ordens de magnitude e podem ser representadas pelos tensores a seguir:

$$\mathbf{K} = \begin{cases} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \text{se } \mathbf{K} = \mathbf{K}_1 \\ \begin{pmatrix} 10^{-4} & 0 \\ 0 & 10^{-4} \end{pmatrix} & \text{se } \mathbf{K} = \mathbf{K}_2 \end{cases} \quad (5.12)$$

Para a solução numérica deste problema utilizamos uma malha não-estruturada com 898 elementos triangulares de 1ª ordem. As variáveis pressão e velocidade foram obtidas através do MEFM resolvido de modo implícito. A velocidade obtida foi utilizada para o acoplamento com a equação da saturação, a qual foi discretizada utilizando o MEF com estabilização via SUPG e adição de difusão artificial nas regiões de maiores gradientes, cuja indicação é obtida através de um termo de captura de choque descrito no capítulo 3. A parte hiperbólica também foi resolvida de maneira implícita resultando portanto em um esquema de avanço no tempo do tipo sequencial implícito. A saturação calculada é usada para atualizar as propriedades (apenas a permeabilidade relativa para este caso particular) utilizadas pela equação da pressão e velocidade.

Na Fig. 5.11 é apresentado o campo de velocidade e o campo de saturação (extrudado para fins de visualização) para o tempo $t = 0.3$ PVI (Porous Volume Injected). Pode ser observado nesta figura que a região de baixa permeabilidade é “contornada” pelo campo de velocidade, o que era esperado devido à razão entre as permeabilidades de 10000. Nota-se também de modo qualitativo que as maiores velocidades são obtidas próximas aos poços injetores e produtos, como era de se esperar.

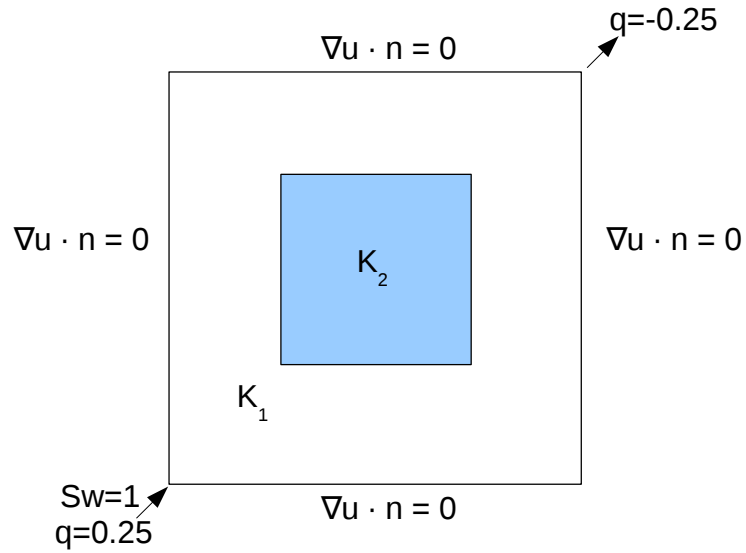


Figura 5.10 Descrição do problema de 1/4 de cinco poços heterogêneo.

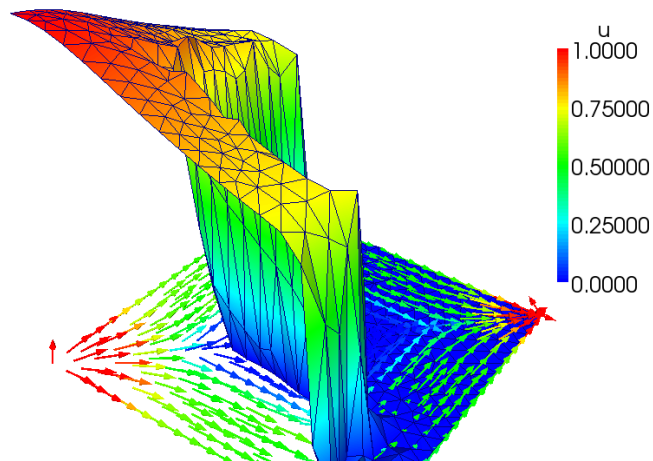


Figura 5.11 Campo de velocidade e representação extrudada da saturação para problema de 1/4 de cinco poços heterogêneo.

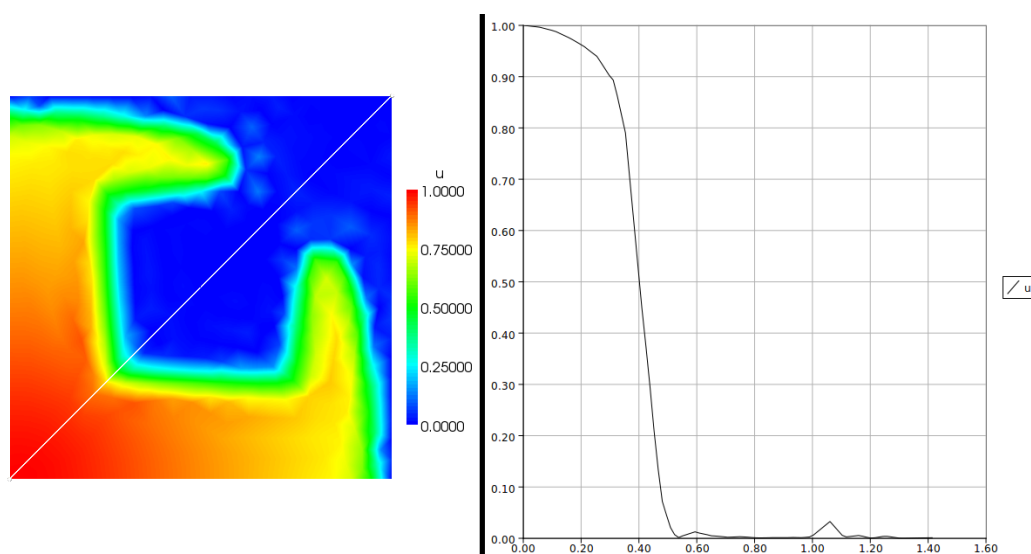


Figura 5.12 Campo de saturação e representação do perfil ao longo da diagonal entre poços para problema de 1/4 de cinco poços heterogêneo.

Na Fig. 5.12 é apresentado o campo de saturação e o seu perfil traçado na diagonal entre os poços injetores e produtores. Percebe-se que a frente de saturação não atravessa a região de baixa permeabilidade, ou seja, praticamente não há entrada de água nela. De um modo geral, o uso de difusão artificial foi suficiente para reduzir bastante a presença de oscilações não-físicas, porém percebe-se na representação do perfil de saturação ao longo da diagonal que ainda assim existem algumas oscilações espúrias após a frente de saturação na solução numérica.

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Nesta dissertação foi apresentado o desenvolvimento, utilizando técnicas de modelagem automática, de um software para a simulação de escoamentos bifásicos óleo-água em meios porosos. Na formulação matemática foi considerado que tal escoamento obedece a Lei de Darcy, sendo que o procedimento adotado se mostrou suficientemente flexível para lidar com domínios homogêneos e heterogêneos e com tensores de permeabilidade iso- e anisotrópicos.

As EDPs resultantes do modelo matemático foram resolvidas numericamente utilizando o MEF, o que permitiu o uso de malhas tanto estruturadas quanto não-estruturadas. Algumas variantes deste método foram utilizadas ao longo deste trabalho, sendo as conclusões do uso de cada uma mencionadas a seguir:

- *Método dos Elementos Finitos (MEF) de Galerkin*: Utilizado para calcular a pressão. Se mostrou adequado para resolver problemas elípticos com soluções suaves. As soluções numéricas obtidas foram comparadas com as analíticas para problemas modelo da literatura, tendo sido encontrada uma boa concordância entre os resultados. Foram efetuados estudos de convergência de malha com o uso deste método e taxas de convergência condizentes com a ordem dos elementos triangulares utilizados foram obtidas. Entretanto, para se resolver o problema completo de escoamento em meios porosos utilizando a formulação matemática adotada, é necessário que se obtenha o campo de velocidade com uma boa precisão, o que não se mostrou possível com este método ao se calcular a velocidade utilizando diretamente o gradiente da pressão. Sendo assim, foi utilizada uma segunda variante do MEF, cujas conclusões são descritas a seguir.

- *Método dos Elementos Finitos Mistos (MEFM)*: Utilizado para calcular a pressão e a velocidade simultaneamente. Apresenta uma formulação numérica mais complexa do que o método anterior e além disso apresenta uma restrição de estabilidade em relação às combinações de espaços de funções que podem ser escolhidas para as funções tentativa e de ponderação. Por outro lado, o fato de calcular simultaneamente a pressão e a velocidade permitiu obter um campo de velocidades adequado sem a necessidade de pós-processamento para a solução da equação da saturação, a qual foi resolvida pelo método a seguir.
- *MEF com estabilização via Streamline Upwind Petrov-Galerkin (SUPG) e adição de difusão numérica artificial*: Utilizado para calcular a saturação. O uso do MEF sem nenhuma estabilização se mostrou completamente inadequado para as partes essencialmente hiperbólicas dos problemas considerados. Por isso, foi inicialmente adicionado um termo de estabilização do tipo SUPG, o qual adiciona difusão na direção das linhas de corrente do fluxo. Este se mostrou adequado para estabilizar a solução e impedir que o erro crescesse indefinidamente, porém a existência de oscilações não-físicas na solução levou à adição de um termo de captura de choque para adição de difusão numérica artificial apenas nas regiões de variação acentuada do gradiente. A combinação destes dois termos se mostrou adequada para os casos abordados, tendo sido obtida uma boa concordância na comparação realizada entre a solução numérica e a analítica para problemas modelo unidimensionais. O uso deste método também foi analisado para um domínio bidimensional e heterogêneo, para o qual foram obtidos bons resultados. Entretanto, há de se observar que as oscilações não-físicas foram reduzidas mas não completamente eliminadas das soluções numéricas obtidas.

Foi testada uma técnica de aceleração de convergência para a resolução do sistema de equações lineares provenientes da discretização da parte elíptica do problema. A técnica utilizada foi a de Multigrid Algébrico (AMG), a qual apresenta como diferencial permitir resolver todas as faixas de frequências do erro ao mesmo tempo, ao contrário dos métodos convencionais que resolvem apenas a faixa de alta-frequência. Foram comparadas a quantidade de iterações até a convergência necessária para métodos Multigrid do tipo AMG, métodos iterativos do tipo Método dos Gradientes Conjugados (CG) e *Generalized Minimal Residual Method* (GMRES) sem o uso de pré-condicionadores (apenas para fins de comparação, já que esta não é a prática usual) e por fim foi testado o uso do AMG como preconditionador para os métodos CG e GMRES. Observou-se que métodos do tipo Multigrid apresentaram uma convergência mais rápida para os problemas considerados,

sendo que o seu uso como preconditionador aumentou ainda mais a taxa de convergência.

Por fim, é fundamental frisar a importância deste trabalho ter sido realizado utilizando de ferramentas computacionais para automação do processo de desenvolvimento do código. O pacote *FEniCS/DOLFIN* se mostrou bastante útil e versátil para a aplicação na área de simulação de escoamentos em meios porosos, já que permite acrescentar com grande facilidade novos métodos numéricos e testar diferentes formulações matemáticas. Isto se deve em grande parte ao fato da codificação por parte do usuário poder ser feita na linguagem *Python* de uma forma bastante similar àquela utilizada na descrição matemática do problema. Ou seja, ao invés de se implementar diversas funções para o cálculo de operadores como o de gradiente ou divergente, os mesmos já estão implementados no pacote sendo necessário apenas chamá-los dentro do programa principal. Como exemplo da praticidade de uso deste sistema para automação da geração de código, podemos citar o fato de que toda a parte computacional apresentada nesta dissertação foi desenvolvida em menos de 1 ano de trabalho, sendo que o desenvolvimento de todo o aparato numérico-computacional utilizado provavelmente teria demorado muito mais tempo caso todas as ferramentas tivessem que ter sido desenvolvidas de modo independente.

6.2 Trabalhos Futuros

Devido à abrangência do campo estudado, logicamente não é possível abordar todas as possíveis áreas de interesse em um único trabalho. Entretanto, graças ao uso de técnicas de modelagem automática como as utilizadas e descritas nesta dissertação, se torna relativamente simples a expansão do código aqui desenvolvido para lidar com problemas mais complexos ou gerais. Sendo assim, podemos citar como temas de interesse para trabalhos futuros os seguintes tópicos:

- *Adição de Difusão Artificial Anisotrópica:* Neste trabalho foi utilizada a adição de difusão isotrópica, sendo que é demonstrado na literatura (Codina, 1992) que o termo do tipo SUPG já adiciona dissipação suficiente ao longo das linhas de corrente, sendo portanto necessário a adição de difusão apenas perpendicularmente ao fluxo, evitando assim uma sobre-difusão em determinada direção.
- *Formulação Conservativa do MEF de Galerkin:* A conservação local é uma propriedade de interesse na simulação do escoamento de fluidos de um modo geral e especialmente na modelagem e gerenciamento de reservatórios de petróleo. Uma das críticas comuns ao MEF de Galerkin e às formulações estabilizadas derivadas

dele é exatamente o fato destas formulações supostamente garantirem conservação apenas no nível global, o que justificaria a escolha de métodos localmente conservativos como o Método dos Volumes Finitos (MVF) ou o Galerkin Descontínuo. Entretanto, na literatura (Hughes *et al.*, 2000; Hughes e Wells, 2005) é demonstrado que o tradicional método de Galerkin contínuo é conservativo localmente, desde que se utilize na formulação original uma modificação baseada no resíduo. Logo, uma extensão natural deste trabalho seria a adição dos termos necessários para garantir a conservação também a nível local.

- *Mais “Física”: Capilaridade, Gravidade e Modelo Black-Oil*: A formulação matemática utilizada neste trabalho teve como foco representar apenas as características principais do escoamento para verificar a utilização da metodologia numérica e das ferramentas computacionais descritas nesta dissertação. Entretanto, o escoamento multifásico em meios porosos envolve várias outras nuances que devem ser melhor abordadas para representar mais fielmente o problema real.
- *Computação Paralela com MPI ou OpenMP*: Foi utilizado neste trabalho o método Multigrid para aceleração de convergência que permite diminuir o tempo necessário para a resolução do sistema de equações lineares para a parte elíptica do problema. Entretanto, este tempo de computação provavelmente continuaria sendo proibitivo para problemas de maior escala envolvendo milhões de graus de liberdade. Sendo assim, uma das alternativas seria utilizar técnicas de computação paralela para distribuir a carga total entre vários processadores. O pacote *FEniCS/DOLFIN* possui uma interface simples que permite o uso tanto de computadores com memória distribuída (com MPI) quanto compartilhada (com OpenMP) sem que seja necessário fazer alterações significativas no código desenvolvido para uso em apenas um processador.

Referências Bibliográficas

- Aarnes, J., Gimse, T., e Lie, K.-A. (2007). *An Introduction to the Numerics of Flow in Porous Media using Matlab*, p. 265–306. Springer, Berlin.
- Ahmed, T. (2006). *Reservoir Engineering Handbook*. Elsevier, Oxford.
- Aziz, K. e Settari, A. (1979). *Petroleum Reservoir Simulation*. Elsevier, Calgary.
- Barbosa, A., Catabriga, L., Valli, A., Malta, S., e Lima, L. (2009). Experiments using a finite element formulation of incompressible miscible displacement in porous media. In *32th Congresso Nacional de Matemática Aplicada e Computacional*, Cuiabá.
- Bear, J., Beljin, M., e Ross, R. (1992). Fundamentals of ground-water modeling. *EPA Ground Water Issue*, **13**(1), 1–11.
- Bell, W. N., Olson, L. N., e Schroder, J. (2008). Pyamg: Algebraic multigrid solvers in Python. Version 1.0.
- Bray, T., Paoli, J., Sperberg-McQueen, C., e Maler, E. (2000). Extensible Markup Language (XML). Version 1.0.
- Brezzi, F. e Fortin, M. (1991). *Mixed and Hybrid Finite Element Methods*. Springer, Nova Iorque.
- Brezzi, F., Douglas, J., e Marini, L. (1985). Two families of mixed finite elements for second order elliptic problems. *Numerische Mathematik*, **47**(1), 217–235.
- Briggs, W., Henson, V. E., e McCormick, S. (2000). *A Multigrid Tutorial*. SIAM, Filadélfia.
- Brooks, A. e Hughes, T. (1982). Streamline Upwind/Petrov-Galerkin formulations for convections dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, **32**(1), 199–259.

- Buckley, S. e Leverett, M. (1942). Mechanism of fluid displacement in sands. *Transactions of the AIME*, **146**(1), 107–116.
- Carvalho, D. K. E. (2005). *Uma Formulação do Método dos Volumes Finitos com Estrutura de Dados por Aresta para a Simulação de Escoamentos em Meios Porosos*. Tese de Doutorado, Universidade Federal de Pernambuco (UFPE), Recife.
- Chavent, G. e Jaffre, J. (1986). *Mathematical Models and Finite Elements for Reservoir Simulation*. North Holland, Amsterdam.
- Chen, Z., Huan, G., e Ma, Y. (2006). *Computational Methods for Multiphase Flows in Porous Media*. SIAM, Filadélfia.
- Clark, J. (2001). Relax NG Specification. Version 1.0.
- Codina, R. (1992). A discontinuity-capturing crosswind-dissipation for the finite element solution of the convection-diffusion equation. *Computer Methods in Applied Mechanics and Engineering*, **110**(1), 325–342.
- Cordazzo, J. (2006). *Simulação de Reservatórios de Petróleo Utilizando o Método EbFVM e Multigrid Algébrico*. Tese de Doutorado, Universidade Federal de Santa Catarina (UFSC), Florianópolis.
- Crumpton, P. (1995). Discretization and multigrid solution of elliptic equations with mixed derivative terms and strongly discontinuous coefficients. *Journal of Computational Physics*, **116**(1), 343–358.
- Ewing, R. (1983). *The Mathematics of Reservoir Simulation*. SIAM, Philadelphia.
- Fortuna, A. (2000). *Técnicas Computacionais para Dinâmica dos Fluidos: Conceitos Básicos e Aplicações*. Edusp, São Paulo.
- Garcia, E. L. M. (1997). *Formulações de Elementos Finitos Bi e Tridimensionais para Simulação em Paralelo de Escoamentos em Reservatórios de Petróleo*. Tese de Doutorado, Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro.
- Geuzaine, C. e Remacle, J.-F. (2009). Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, **79**(11), 1309–1331.
-

- Henderson, A. (2007). Paraview Guide, A Parallel Visualization Application. *Kitware Inc. Report*.
- Hughes, T. (2000). *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, Nova Iorque.
- Hughes, T. e Wells, G. (2005). Conservation properties for the Galerkin and stabilized forms of the advection-diffusion and incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, **194**(9), 1141–1159.
- Hughes, T., Engel, G., Mazzei, L., e Larson, M. (2000). The continuous Galerkin method is locally conservative. *Journal of Computational Physics*, **163**(1), 467–488.
- Hurtado, F. S. V. (2005). *Uma Formulação de Volumes Finitos Baseada em Elementos para a Simulação do Deslocamento Bifásico Imiscível em Meios Porosos*. Dissertação de Mestrado, Universidade Federal de Santa Catarina (UFSC), Florianópolis.
- Hyman, J., Shashkov, M., e Steinberg, S. (1997). The numerical solution of diffusion problems in strongly heterogeneous non-isotropic materials. *Journal of Computational Physics*, **132**(1), 130–148.
- Hyne, N. (2001). *Nontechnical Guide to Petroleum Geology, Exploration, Drilling and Production*. Pennwell, Tulsa.
- Jones, E., Oliphant, T., e Peterson, P. (2001). SciPy: Open source scientific tools for Python.
- Langtangen, H. (2011). *A FEniCS Tutorial*. Springer, Berlin.
- Leveque, R. (1992). *Numerical Methods for Conservation Laws*. Birkhäuser, Basel.
- Logg, A. e Wells, G. (2010). Dofin: Automated finite element computing. *ACM Transactions on Mathematical Software*, **37**(2), 417–444.
- Logg, A., Mardal, K.-A., e Wells, G. (2011). *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*. Springer, Berlin.
- Loula, A., Guerreiro, J., Ribeiro, F., e Landau, L. (1995). Tracer injection simulations by finite element methods. *SPE Advanced Technology Series*, **4**(1), 150–156.

- Luna, B., Lyra, P., e Willmersdorf, R. (2011). Automated finite element computation of the pressure equation for porous media flow using convergence acceleration via multigrid. In *32th Ibero Latin American Congress on Computational Methods in Engineering*, Ouro Preto.
- Marle, C. (1981). *Multiphase Flow in Porous Media*. Gulf Publishing, Paris.
- Matamoros, L. e Brueggemann, D. (2006). Simulation of the water and heat management in proton exchange membrane fuel cells. *Journal of Power Sources*, **161**(1), 203–213.
- Mavriplis, D. (2001). An assessment of linear versus non-linear multigrid methods for unstructured mesh solvers. *NASA/ICASE Technical Report*, **2001**(12).
- Menezes, D. S. (2009). *Utilização de Métodos de Transferência de Escala na Simulação de Recuperação de Hidrocarbonetos com Aplicação de Computação Distribuída*. Dissertação de Mestrado, Universidade Federal de Pernambuco (UFPE), Recife.
- Monajemi, H. (2009). *Data Assimilation for Shallow Water Waves: Application to Flood Forecasting*. Dissertação de Mestrado, Carleton University, Ottawa.
- Oosteele, C. e Washio, T. (1998). On the use of multigrid as a preconditioner. In *Ninth International Conference on Domain Decomposition Methods*, Bergen.
- Papastavrou, A. (1998). *Adaptive Finite Element Methoden für Konvektions-Diffusionsprobleme*. Tese de Doutorado, Ruhr-Universität Bochum, Bochum.
- Peaceman, D. (1977). *Fundamentals of Numerical Reservoir Simulation*. Elsevier, Amsterdam.
- Peaceman, D. e Nash, S. (1990). *A History of Scientific Computing*. ACM Press, New York.
- Rathgeber, F. (2010). *Automated Finite Element Computation in the FEniCS Framework using General Purpose Graphics Processing Units*. Dissertação de Mestrado, Royal Institute of Technology (KTH), Estocolmo.
- Raviart, P. e Thomas, J. (1977). *A mixed finite element method for 2nd order elliptic problems*, volume 606, p. 292—315. *Lecture Notes in Math.*, Vol. 606. Springer.
- Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia.
-

- Schneider, J. (2003). New least squares model used for development of permeability-porosity correlation. Technical report, Schneider Reservoir Engineering Consultant, Poteet.
- Sheen, D. (2007). Relax NG validator. Version 1.7.
- Silva, R. S. (2008). *Simulação de Escoamento Bifásico Oléo-Água em Reservatórios de Petróleo Usando Computadores Paralelos de Memória Distribuída*. Tese de Doutorado, Universidade Federal de Pernambuco (UFPE), Recife.
- Thomas, J. E. (2001). *Fundamentos de Engenharia de Petróleo*. Ed. Interciência, Rio de Janeiro.
- Trottenberg, U., Oosterlee, C., e Schüller, A. (2001). *Multigrid*. Elsevier, Oxford.
- Wells, G., Hooijkaas, T., e Shan, X. (2008). Modelling temperature effects on multiphase flow through porous media. *Philosophical Magazine*, **88**(28), 3265–3279.

Apêndice A

Delfine - Manual do usuário

Ao longo desta dissertação foi discutido todo o arcabouço matemático, numérico e computacional necessário para a obtenção dos resultados apresentados. Entretanto, como comentado no Cap. 4, todo este trabalho só foi possível de ser realizado graças às inúmeras ferramentas computacionais disponibilizadas publicamente na internet através de licenças livres. Sendo assim, este apêndice tem o objetivo de servir como manual prático para obtenção e uso do programa desenvolvido, na esperança de que possa ser mais útil para outrem e assim fornecer uma pequena contribuição à comunidade.

A.1 Dependências e *Download* do Programa

O *software* foi todo desenvolvido na distribuição *Ubuntu 10.04* do sistema operacional *GNU/Linux*, logo a descrição do procedimento de instalação e uso do programa será focada neste ambiente. Em todo caso, os comandos e pacotes aqui descritos podem ser executados ou instalados com alterações mínimas em qualquer outra distribuição recente do *GNU/Linux*.

Para uso do *Delfine* é necessário que alguns programas já estejam instalados no sistema. A lista a seguir indica quais são estes pré-requisitos e onde eles podem ser obtidos. Obviamente, qualquer programa gerenciador de pacotes (*apt-get*, *rpm*, etc.) pode ser utilizado para agilizar o processo de instalação deles.

- *FEniCS* (www.fenicsproject.org): Biblioteca para automação da solução de EDPs. O *DOLFIN* é parte integrante dela.
- *Python 2.x* (www.python.org): Linguagem de programação interpretada utilizada neste trabalho.

- *SciPy* (www.scipy.org): Biblioteca para computação científica em *Python*.
- *PyAMG* (www.code.google.com/p/pyamg): Biblioteca de resolvedores AMG com interface para *Python*.
- *RNV* (www.davidashen.net/rnv.html): Validador de arquivos *XML*.

Além destas dependências, os programas listados a seguir foram usados ao longo deste trabalho em conjunto com o *Delfine* para a geração dos modelos e análise dos resultados obtidos, logo recomenda-se a instalação e uso deles.

- *Gmsh* (www.geuz.org/gmsh): Programa para geração de malhas estruturadas e não-estruturadas em 2 ou 3 dimensões.
- *Paraview* (www.paraview.org): Programa para visualização dos resultados.
- *Matplotlib* (<http://matplotlib.sourceforge.net/>): Biblioteca para geração de gráficos.

O próximo passo é a obtenção do código-fonte do *Delfine*. O serviço de hospedagem de projetos da Google (*Google Codes*) foi utilizado em conjunto com o sistema de controle de versão *Apache Subversion* para gerenciamento do código. Na página deste projeto na internet (<http://code.google.com/p/delfine>) estão disponíveis, além do código em si, vários documentos, como a versão digital desta dissertação, artigos e apresentações utilizados em congressos, entre outros. A seguir é mostrado o comando a ser digitado no terminal em uma pasta qualquer para baixar a versão mais recente do programa.

```
$ svn checkout http://delfine.googlecode.com/\
svn/trunk/delfine
```

Listagem 8: *Download* do programa a partir de repositório *svn*.

A.2 Dados de Entrada

O programa é executado a partir da linha de comando utilizando o script `run.py`, o qual se encontra na pasta-raiz *Delfine*. O comando necessário para executar uma análise é apresentado a seguir, onde “caseName” deve ser substituído pelo nome do arquivo de dados considerado.

```
$ ./run.py 'caseName'
```

Listagem 9: Execução do programa a partir do terminal.

Uma vez solicitada a execução do programa, são necessários dois arquivos de entrada, um com os dados da análise e descrição do problema em si e outro com as informações da malha (existe uma exceção para este caso, a qual será discutida adiante). No diretório para o qual o programa foi baixado existe uma pasta chamada *Delfine/CaseFiles*, dentro da qual existem alguns arquivos que podem ser usados como base para a criação de outros modelos. Descreveremos a seguir como estes dois arquivos devem ser gerados.

A.2.1 Arquivo de Dados

As informações necessárias para a análise encontram-se neste arquivo subdivididas em três grupos:

Geometry: Neste grupo é informado para o programa qual a malha que será considerada. Além disso, são fornecidas neste grupo as condições de contorno que descrevem o problema. Um modelo para este grupo pode ser visto na listagem a seguir:

XML

```
<geometry>
  <mesh dimension="2" order="1" type="gmsh">
    <filename>malha.msh</filename>
  </mesh>
  <boundary-conditions>
    <well function="injection" id="1">1.0</well>
    <well function="production" id="2">-1.0</well>
  </boundary-conditions>
</geometry>
```

Listagem 10: Grupo *Geometry*.

O tipo de malha é escolhido através do parâmetro `type`, o qual pode ser definido como: `gmsh`, caso a malha tenha sido gerada externamente por este programa, sendo que neste caso ela será automaticamente convertida para o formato `*.xml` pelo script *delfine-convert* considerando os indicadores de condição de contorno fornecidos; `xml`, caso a malha tenha sido gerado por outro programa (*Triangle*, *Medit*, *ExodusII*, etc.) e

convertida manualmente pelo script *delfine-convert*; ou *dolfin-generated*, caso a malha seja criada utilizando o gerador interno do *DOLFIN*. Neste último caso, as opções de malha são definidas usando os seguintes parâmetros:

```
XML
<geometry>
  <mesh dimension="2" type="dolfin-generated">
    <dolfin-generated type="UnitSquare" nx="8" ny="8"/>
  <subdomains quantity="1"/>
</mesh>
.
.
</geometry>
```

Listagem 11: Definição de malha a ser gerada pelo *DOLFIN*.

Apenas geometrias simples podem ser geradas com esta opção, sendo que as alternativas são as seguintes: *UnitSquare* (para domínios bidimensionais com $n_x \times n_y$ elementos), *UnitInterval* (para domínios unidimensionais com n_x elementos), *UnitCube* (para domínios tridimensionais com $n_x \times n_y \times n_z$ elementos) e *UnitCircle* (para domínios bidimensionais com n_r elementos na direção radial) .

Por fim, para cada poço existente no domínio deve ser escrita uma linha no subgrupo *boundary-conditions* contendo o tipo de poço (injeção ou produção), o identificador dele (correspondente ao definido durante a geração da malha) e o valor do fluxo total prescrito. No momento estas condições ainda são bastante restritivas, por isso um dos objetivos para o futuro é adicionar modelos de poços mais gerais e realistas.

Physical: Neste grupo são descritas as propriedades das rochas, dos fluidos e de interação rocha-fluido. É importante ressaltar que alguns dos parâmetros existentes no arquivo de dados são referentes a funcionalidades ainda não implementadas. A viscosidade e densidade de fluidos, por exemplo, são normalmente consideradas dependentes da temperatura e pressão, respectivamente, mas nesse momento são ainda definidas com valores constantes, por isto se utilizou a opção *model="none"* no arquivo de dados. O mesmo comentário vale para a porosidade da rocha. Um modelo para este grupo pode ser visto na listagem a seguir:

XML

```

<physical>
  <fluid-properties>
    <water use="no">
      <viscosity model="yes">1.0</viscosity>
    </water>
    <oil use="yes">
      <viscosity model="none">1.0</viscosity>
    </oil>
  </fluid-properties>
  <rock-properties>
    <rock-type id="1">
      <porosity compressible="no">1.0</porosity>
      <permeability type="per-domain">
        <Kxx>1</Kxx>
        <Kxy>0.0</Kxy>
        <Kyy>1</Kyy>
      </permeability>
    </rock-type>
  </rock-properties>
  <rock-fluid-properties>
    <relative-permeability model="corey">
      <krw>
        <krw_end>1.0</krw_end>
        <Swc>0.0</Swc>
        <nw>2.0</nw>
      </krw>
      <kro>
        <Sor>0.0</Sor>
        <no>2.0</no>
      </kro>
    </relative-permeability>
  </rock-fluid-properties>
</physical>

```

Listagem 12: Grupo *Physical*.

Em relação às propriedades das rochas, deve ser adicionado um subgrupo do tipo `rock-properties` com uma número de identificação correspondente ao de cada um dos subdomínios definidos no arquivo de malha. O tensor de permeabilidade pode ser expandido ou reduzido de acordo com o número de dimensões do problema. Para um caso tridimensional, por exemplo, bastaria adicionar linhas para os valores de K_{xz} e K_{zz} .

Numerical: Neste grupo são descritos os parâmetros de análise utilizados para a simulação numérica do problema. A seguir é apresentado um exemplo com as principais opções:

```
XML
<numerical>
  <pressure-solver formulation="galerkin" type="cg">
    <tolerance>1e-10</tolerance>
    <max-number-steps>1000</max-number-steps>
    <pre-conditioning type="amg">
      <number-coarse-levels>5</number-coarse-levels>
    </pre-conditioning>
  </pressure-solver>
  <saturation-solver>
    <total-time-analysis>200</total-time-analysis>
    <courant>0.9</courant>
    <limiter type="none"/>
  </saturation-solver>
</numerical>
```

Listagem 13: Grupo *Numerical*.

O primeiro subgrupo (`pressure-solver`) pode receber as formulações `galerkin` ou `mixedfem` (ver Cap. 3 para detalhes sobre as características de cada uma) e os tipos `cg` (Método dos Gradientes Conjugados (CG)), `gmres` (Método do Resíduo Mínimo Generalizado (GMRES)) ou `none`, sendo que neste último caso o AMG é utilizado isoladamente. Já o subgrupo `pre-conditioning` pode ser do tipo `amg` ou `none`, sendo que neste caso deve obrigatoriamente ser escolhido algum tipo de resolvidor em `pressure-solver`. Devido à disponibilidade de vários métodos iterativos dentro da biblioteca *PyAMG*, espera-se em breve adicionar novas opções de resolvidores iterativos e diretos.

As opções para o parâmetro `limiter` são: `SUPG` (ou seja, apenas o termo de estabilização), `SUPG+Wells` (estabilização via `SUPG` e adição de difusão artificial) e `none` (MEF de Galerkin aplicado diretamente para a equação hiperbólica). Em relação ao tipo de limitador para captura de choque, até o momento apenas o termo de difusão artificial isotrópica sugerido por Wells *et al.* (2008) está disponível, porém um dos trabalhos futuros (ver Seção 6.2) é exatamente a adição de difusão artificial anisotrópica conforme proposto por Codina (1992).

A.2.2 Arquivo de Malha

Existe a opção de ter a malha gerada pelo próprio *DOLFIN* ou então usar um gerador externo e convertê-la utilizando um *script*. No primeiro caso não é necessário fornecer um arquivo, já na segunda alternativa a malha é descrita através de um arquivo do tipo `*.xml`, o qual contém as informações referentes aos nós, elementos e *flags* indicadores de domínio ou condição de contorno. A estrutura geral do arquivo é conforme o exemplo a seguir:

```

XML
<dolfin xmlns:dolfin="http://www.fenicsproject.org">
  <mesh celltype="triangle" dim="2">
    <vertices size="25">
      <vertex index="0" x="0" y="0"/>
      <vertex index="1" x="0" y="1"/>
      .
      .
    </vertices>
    <cells size="32">
      <triangle index="0" v0="0" v1="4" v2="18"/>
      <triangle index="1" v0="4" v1="17" v2="18"/>
      .
      .
    </cells>
  </mesh>
</dolfin>

```

Listagem 14: Arquivo de malha.

A.3 Exemplo Detalhado

Nesta seção, apresentaremos a construção detalhada de um exemplo utilizado neste trabalho de modo a consolidar todas as informações apresentadas nas seções anteriores deste manual. Os dados do problema que serão utilizados para a geração do arquivo de dados são apresentados no capítulo de resultados na Seção 5.2.2.

Este problema trata da solução do escoamento em meios porosos em um domínio heterogêneo representado por um quadrado unitário com uma região de baixa permeabilidade no seu interior, sendo portanto uma geometria bastante simples. Mesmo assim, iremos considerar o uso do *Gmsh* para a geração desta malha devido à existência dos 2 subdomínios. Detalhes sobre o uso da interface gráfica do *Gmsh* para se gerar interativamente a geometria podem ser encontrados na página do programa citada na Seção A.1. Uma vez criada a geometria na interface, o *Gmsh* salva as informações em um arquivo do tipo `*.geo`, o qual é utilizado para gerar a malha em si.

O arquivo de malha (`*.msh`) pode ser obtido executando o seguinte comando em um terminal:

```
$ gmsh file.geo
```

Listagem 15: Geração da malha usando o *Gmsh* a partir de um arquivo de geometria.

Além das informações geométricas (coordenadas dos pontos, linhas que formam os contornos, etc.), são fornecidos no arquivo de geometria os identificadores (*flags*) de subdomínio e de condições de contorno (p.ex., poços), os quais são indicados pelos comandos `Physical Surface` e `Physical Point`, respectivamente. O arquivo do tipo `*.geo` usado para gerar a geometria desejada para este exemplo é descrito na listagem a seguir:

```

Point (1) = {0, 0, 0, 0.05};
Point (2) = {0, 1, 0, 0.05};
Point (3) = {1, 0, 0, 0.05};
Point (4) = {1, 1, 0, 0.05};
Point (5) = {0.25, 0.25, 0, 0.05};
Point (6) = {0.25, 0.75, 0, 0.05};
Point (7) = {0.75, 0.25, 0, 0.05};
Point (8) = {0.75, 0.75, 0, 0.05};

Line (1) = {1, 3};
Line (2) = {3, 4};
Line (3) = {4, 2};
Line (4) = {2, 1};
Line (5) = {5, 7};
Line (6) = {7, 8};
Line (7) = {8, 6};
Line (8) = {6, 5};

Line Loop (352) = {7, 8, 5, 6};
Plane Surface (352) = {352};

Line Loop (353) = {3, 4, 1, 2, -6, -5, -8, -7};
Plane Surface (353) = {353};

Physical Surface (352) = {352};
Physical Surface (353) = {353};

Physical Point (301) = {1};
Physical Point (351) = {4};

```

Listagem 16: Arquivo de geometria.

Em seguida, é criado um arquivo de dados para o problema. Inicialmente, precisamos fornecer os dados relativos ao grupo *Geometry*, conforme descrito na Seção A.2.1. Para este caso, definimos o tipo de malha como `gmsht-generated`, a dimensão como 2, e o nome do arquivo como `exemplo_heterogeneo.msh`. As condições de contorno

são os poços de injeção e produção, os quais recebem os identificadores definidos no arquivo de geometria (301 para injeção e 351 para produção). Para isto, definimos os parâmetros *function* e *id*. Assim, este primeiro trecho do arquivo de dados deve ter a seguinte aparência:

```
XML
<geometry>
  <mesh dimension="2" order="1" type="gmsht">
    <filename>exemplo_heterogeneo.msh</filename>
  </mesh>
  <boundary-conditions>
    <well function="injection" id="301">0.25</well>
    <well function="production" id="351">-0.25</well>
  </boundary-conditions>
</geometry>
```

Listagem 17: Grupo *Geometry* para exemplo detalhado.

Em seguida, é necessário definir os parâmetros dos fluidos e rochas considerados no grupo *Physical*. Os valores para as propriedades dos fluidos e de interação rocha-fluidos são informados do mesmo modo que na Listagem 12. Já as propriedades das rochas devem ser informadas utilizando dois blocos do tipo *rock-properties*, um para cada domínio (regiões de alta e de baixa permeabilidade). No caso de existirem outros domínios, basta adicionar quantas seções foram necessárias, já que a leitura de dados de entrada é suficientemente flexível para só considerar os dados de interesse, desde que o ID da rocha adicionada corresponda ao *flag* dos elementos no arquivo de malha (comando *Physical Surface* no arquivo de geometria). Os dados para este exemplo específico são mostrados a seguir:

XML

```

<rock-properties>
  <rock-type id="352">
    <porosity compressible="no">1.0</porosity>
    <permeability type="per-domain">
      <Kxx>0.0001</Kxx>
      <Kxy>0.0</Kxy>
      <Kyy>0.0001</Kyy>
    </permeability>
  </rock-type>
  <rock-type id="353">
    <porosity compressible="no">1.0</porosity>
    <permeability type="per-domain">
      <Kxx>1</Kxx>
      <Kxy>0.0</Kxy>
      <Kyy>1</Kyy>
    </permeability>
  </rock-type>
</rock-properties>

```

Listagem 18: Propriedades das rochas.

Por fim, os parâmetros numéricos para a simulação são fornecidos dentro do grupo *Numerical*. Para a solução do problema elíptico utilizaremos a formulação do Método dos Elementos Finitos Mistos (MEFM) devido à sua característica de calcular simultaneamente os campos de pressão e de velocidade (ver Cap. 3 para detalhes). Já para a solução do problema hiperbólico, foi utilizada a estabilização da solução via *Streamline Upwind Petrov-Galerkin* (SUPG) e um termo de difusão artificial isotrópica para redução das oscilações espúrias conforme descrito em (Wells *et al.*, 2008). O trecho a seguir mostra os dados utilizados para este exemplo:

XML

```
<numerical>
  <pressure-solver formulation="mixedfem" type="gmres">
    <tolerance>1e-10</tolerance>
    <max-number-steps>1000</max-number-steps>
    <pre-conditioning type="none">
      <number-coarse-levels>5</number-coarse-levels>
    </pre-conditioning>
  </pressure-solver>
  <saturation-solver>
    <total-time-analysis>200</total-time-analysis>
    <courant>0.9</courant>
    <limiter type="SUPG+Wells"/>
  </saturation-solver>
</numerical>
```

Listagem 19: Grupo *Numerical* para exemplo detalhado.

Uma vez prontos os arquivos de malha e de dados (o qual nomearemos como `ExampleCase.xml`), podemos executar o *Delfine* a partir do terminal utilizando o seguinte comando:

```
$ ./run.py ExampleCase.xml
```

Listagem 20: Execução do caso gerado como exemplo.

A partir deste momento, o programa irá executar todos os passos automaticamente, informando a respeito do passo de tempo da solução no qual se encontra. Ao final, os arquivos com os resultados dos campos de pressão, velocidade e saturação são escritos na pasta `Delfine/Results`. O formato dos arquivos é o `*.vtk`, o qual pode ser aberto utilizando programas como o *Paraview* ou o *VisIt*.