



UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE FÍSICA
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA

MARCELIANO EDUARDO DE OLIVEIRA

NOVO PARADIGMA EM FÍSICA
COMPUTACIONAL - SOFTWARE LIVRE E
COMPUTAÇÃO EM NUVEM - APLICADO
À SOLUÇÃO NUMÉRICA DE EDPS

FORTALEZA

2014

MARCELIANO EDUARDO DE OLIVEIRA

**NOVO PARADIGMA EM FÍSICA
COMPUTACIONAL - SOFTWARE LIVRE E
COMPUTAÇÃO EM NUVEM - APLICADO
À SOLUÇÃO NUMÉRICA DE EDPS**

Tese de Doutorado submetida à Coordenação do Curso de Pós-Graduação em Física, da Universidade Federal do Ceará, como requisito parcial para a obtenção do Título de Doutor em Física

Orientador: Prof. Ph.D. Murilo Pereira de Almeida

FORTALEZA

2014

MARCELIANO EDUARDO DE OLIVEIRA

**NOVO PARADIGMA EM FÍSICA
COMPUTACIONAL - SOFTWARE LIVRE E
COMPUTAÇÃO EM NUVEM - APLICADO
À SOLUÇÃO NUMÉRICA DE EDPS**

Tese de Doutorado submetida à Coordenação do Curso de Pós-Graduação em Física, da Universidade Federal do Ceará, como requisito parcial para a obtenção do Título de Doutor em Física

Aprovada em 28/08/2014

BANCA EXAMINADORA

Prof. PhD. Murilo Pereira de Almeida (Orientador)
Universidade Federal do Ceará

Prof. Dr. Ascânio Dias Araújo
Universidade Federal do Ceará

Prof. Dr. Michael Ferreira de Souza
Universidade Federal do Ceará

Prof. Dr. Ivan C. da Cunha Lima
Universidade do Estado do Rio de Janeiro

Prof. Dr. Elton Márcio da Silva Santos
Universidade do Estado do Amazonas

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Setorial de Física

O48n Oliveira, Marceliano Eduardo.
Novo Paradigma em Física Computacional - Software Livre e
Computação em Nuvem - Aplicado à Solução Numérica de EDPs
/ Marceliano Eduardo de Oliveira. – 2014.
245 p.;il.

Tese de Doutorado - Universidade Federal do Ceará, Departamento de Física, Programa de Pós-Graduação em Física, Centro de Ciências, Fortaleza, 2014.

Área de Concentração: Física da Matéria Condensada
Orientação: Prof. Ph.D. Murilo Pereira de Almeida

1. Física Computacional. 2. Volumes Finitos. 3. Software Livre. 4. Métodos Numéricos. 5. Equações Diferenciais Parciais. I.

CDD:530.0285

*Aos amores da
minha vida:
Ione Pampolini de
Rezende Oliveira
e
Ingrid Rezende
Oliveira.*

AGRADECIMENTOS

Gostaria de agradecer a todos os meus familiares, sem distinção; aos amigos: Diego Araújo Frota, Francisco Leandro Oliveira Rodrigues e Cláudio Lucas Nunes de Oliveira, pela colaboração direta e indireta que deram a este trabalho e pela amizade compartilhada; aos amigos: George Chaves da Silva Valadares, Luis Fernando Dos Santos e Gustavo Oliveira de Meira Gusmão sempre presentes e disponíveis; ao Prof. Murilo Pereira de Almeida, pela paciência e disponibilidade ao longo deste extenso processo de orientação e também pela amizade compartilhada; ao Prof. Josué Mendes Filho, pelo convite e pela receptividade na minha migração do Mestrado em Mato Grosso para o Doutorado no Ceará; a todos os demais amigos da UFC; aos professores e amigos da UFMT que mesmo longe sempre se fizeram presentes: Prof. Alberto Sebastião de Arruda, Max de Oliveira Roos, Ricardo Rodrigues de França Bento, Jorge Luiz Brito de Faria, Teldo Anderson da Silva Pereira; aos professores e amigos do departamento de Física do CESP UEA: Adriano Márcio dos Santos, Edilson Barroso Gomes, Elton Márcio da Silva Santos, Francisco Otávio Miranda Farias, Antônio Leocádio Martins Ferreira e Nélío Martins da Silva Azevedo Sazaki, pela parceria e apoio nas minhas liberações para os estudos de doutorado, mesmo em período de estágio probatório; ao diretor do CESP UEA: Prof. David Xavier da Silva, pela parceria e apoio em todas as minhas solicitações de liberação.

RESUMO

O principal objetivo deste trabalho é resolver numericamente problemas físicos associados com os fenômenos de transporte, que são descritos por equações diferenciais parciais, cujas soluções numéricas exigem o uso de códigos de computadores. Optamos por usar plataformas de software livre, especificamente o código R e o OpenFOAM, escrevendo nossas próprios rotinas. O intuito é avaliar a aplicação em física computacional de um novo paradigma, formado pela combinação de software livre e computação em nuvem. A tarefa de resolver numericamente uma equação diferencial parcial passa pelo processo de sua discretização. Estudamos, numa etapa preliminar, os esquemas numéricos frequentemente utilizados para discretizar EDP, encontrados na literatura: Método de Diferenças Finitas (MDF), Método de Volumes Finitos (MVF), Método de Elementos Finitos (MEF) em suas variantes (Galerkin contínuo e descontínuo). Numa etapa posterior, fazemos a implementação computacional do método dos volumes finitos (MVF) em R para difusão de calor em uma e duas dimensões, e para o escoamento bifásico em meios porosos em OpenFoam. A escolha do (MVF) foi motivada pela sua relativa simplicidade de implementação e por apresentar propriedades conservativas baseadas em identidades de cálculo vetorial. Por último foram escolhidos alguns casos de estudo e nestes foram realizadas as tarefas de discretização, implementação computacional e simulação. Durante todo o processo foram levados em conta os seguintes parâmetros: portabilidade do código, reuso do código, estratégias para modificação do domínio. Nossas principais contribuições foram implementar em OpenFOAM e em R códigos de alta portabilidade (desktop, laptop, nuvem) e com bom reuso (segmentação em subrotinas especializadas que podem ser adaptadas a diferentes problemas). Neste trabalho mostramos também como modificar a geometria de um problema já implementado computacionalmente apenas usando um operador matricial que bloqueia pontos do domínio, criando uma heterogeneidade sem alterar drasticamente o código. Aplicamos as técnicas desenvolvidas para o estudo numérico do escoamento multifásico em meios porosos através do método de volumes finitos. Mostramos também a aplicação da mesma metodologia no estudo de problemas clássicos como a difusão em 1D e 2D.

Palavras-chave: Física Computacional. Volumes Finitos. Software Livre. Métodos Numéricos. Equações Diferenciais Parciais.

ABSTRACT

The main goal of this work is numerically solving physics problems associated with transport phenomena, which are described by partial differential equations, whose numerical solution requires the use of computer codes. We use open source software like R and OpenFOAM, writing our own routines. The aim is to approach computational physics from a new paradigm, composed by open source software and cloud computing. To numerically solve a partial differential equation one needs some sort of discretization. We first study the frequent found numerical methods: Finite Difference (FD), Finite Volume (FV), Finite Elements (FE). Latter on, we implement a finite volume scheme for the solution of the diffusion equations in 1D and 2D using R and the two phase flow in a porous media using OpenFOAM. We choose the (FV) for its simplicity and because it has some conservatives properties resulting from vector calculus identities. Finally we perform some numerical experiments. During the work we keep in mind code reusability and portability. Our main achievements are: implementation of highly portable and reusable codes (specialized routines that can be used in different problems) into R and OpenFOAM that can be used from notebooks to cloud computing. We present also a scheme to specify domain heterogeneity through a blocking operator. We apply the studied methodology for numerically solving multiphase flows in porous media. We also show its application for solving classical problems like 1D and 2D diffusion.

Keywords: Computational Physics. Finite Volume. Open Source Software. Numerical Methods. Partial Differential Equations.

SUMÁRIO

1	INTRODUÇÃO	p. 13
1.1	Computação em Nuvem	p. 18
2	CONSERVAÇÃO DE MASSA EM MEIOS POROSOS	p. 23
2.1	Lei de Darcy	p. 23
2.2	Conservação de massa em meios porosos	p. 29
3	MODELOS ESPECÍFICOS	p. 34
3.1	Classificação de Equações Diferenciais Parciais	p. 34
	Exemplos	p. 38
3.2	O Caso Compressível	p. 39
3.3	Aproximação para fluidos pouco compressíveis	p. 41
3.4	Escoamento bifásico	p. 42
4	FUNDAMENTOS DE DISCRETIZAÇÃO	p. 47
4.1	Integradores para Equações Diferenciais Ordinárias	p. 47
	Série de Taylor da função $y(x)$	p. 48
4.1.1	Integradores de Euler	p. 48
4.1.2	Integradores de Runge-Kutta de segunda ordem	p. 54
4.1.3	Integradores de Runge-Kutta de Terceira Ordem ou Superior	p. 58
4.1.4	Alguns Testes com os Integradores Clássicos	p. 59
4.1.5	Sistema de duas EDOs	p. 62
4.1.6	Seleção automática de integradores	p. 64

4.2	Diferenças Finitas e Método de Linhas	p. 66
4.3	Método de Galerkin (Elementos Finitos)	p. 71
4.3.1	Propriedades da aproximação de Galerkin	p. 82
4.3.2	Exemplo de Elementos Finitos: Dois Subdomínios em 1D	p. 90
4.4	Método de Galerkin Descontínuo com Pênalti	p. 95
4.4.1	Considerações gerais sobre o salto	p. 114
4.4.2	Salto através do contorno Γ_-	p. 115
4.4.3	Salto através do contorno Γ_+	p. 116
5	ALGUMAS APLICAÇÕES	p. 117
5.1	Fluxo bifásico em meios porosos com método de linhas	p. 117
5.2	Aplicações do Método de Volumes Finitos	p. 122
5.3	Equação Geral do Transporte	p. 122
5.3.1	Difusão no Estado Estacionário	p. 122
5.3.2	Implementação Computacional da Difusão no Estado Estacionário	p. 130
5.3.3	Difusão com o Termo Transiente	p. 136
5.3.4	Implementação Computacional da Difusão com Termo Transiente	p. 139
5.3.5	Difusão do Calor em 2D com “ θ -method”	p. 149
5.3.6	Validação do Solver	p. 153
5.3.7	Escoamento Bifásico com OpenFOAM	p. 156
5.3.8	Geração da Malha	p. 157
6	CONSIDERAÇÕES FINAIS	p. 160
Appendix A – Dimensionless Equations for Two-Phase Isothermal Flow		
	in Porous Media	p. 161
A.1	Introduction	p. 161
A.2	Methodology	p. 161

A.3	Illustrative Example: Vertical Motion	p. 162
A.4	Example: Navier-Stokes Equation	p. 164
A.5	Two-phase Isothermal Flow in Porous Media	p. 167
A.6	Concluding Remarks	p. 169
A.7	Acknowledgments	p. 169

REFERENCES p. 170

Appendix B – Solving Difusion-Convection Problem Using Finite Volume Method p. 171

B.1	Abstract	p. 171
B.2	Fundamentals	p. 172
B.3	Diffusion Scheme	p. 172
B.4	Convection: Central Difference Scheme	p. 173
B.5	Convection: Upwind Scheme	p. 173
B.6	Convection: QUICK Scheme	p. 174
B.7	Convection: Power Law Scheme	p. 174
B.8	Numerical Experiment	p. 175
B.9	Results	p. 175
B.10	Error Comparison Between Schemes	p. 177
B.11	Concluding Remarks	p. 178
B.12	Acknowledgments	p. 178

REFERENCES p. 179

Appendix C – Heat Transfer In Heterogeneous Domain Using R p. 180

C.1	Abstract	p. 180
C.2	Finite Volume Method	p. 181
C.3	Model Equations	p. 181

C.4	Discretization Procedures	p. 181
C.5	Solving The Linear System	p. 184
C.6	Construction of Freeze Operator	p. 185
C.7	About Our Program	p. 187
C.8	Some Configurations	p. 189
C.9	Concluding Remarks	p. 193
C.10	Acknowledgments	p. 193

REFERENCES		p. 194
-------------------	--	--------

**Appendix D – Influence of the Uncertainty of Geological Parameters
on Numerical Reservoir Simulators**

		p. 195
D.1	Abstract	p. 195
D.2	Introduction	p. 196
D.3	Model Description	p. 197
D.4	Equations	p. 198
D.5	Boundary Conditions	p. 199
D.6	Solver	p. 199
D.7	Simulations	p. 199
D.8	Parameters	p. 200
D.9	Results and Discussion	p. 201
D.10	Selection of variables.	p. 202
D.11	Time of Breakthrough.	p. 204
D.12	Water Injection Rate	p. 205
D.13	Accumulated Injected water	p. 206
D.14	Hypothesis tests	p. 207
D.15	Total produced oil	p. 208
D.16	Conclusions	p. 210

D.16.1 Figures	p. 210
Apêndice E – Código de Alguns Programas	p. 219
E.1 Programa heat.equation.1.R: Método de linhas	p. 219
E.2 Programa heat.equation.2.R: Método de linhas (otimizado)	p. 221
E.3 Programa 2phase.flow.1.R: Método de linhas	p. 223
E.4 Programa difusao1D.estacionario.R: Volumes Finitos	p. 227
E.5 heatTransfer2D.R	p. 228
REFERÊNCIAS	p. 244

1 INTRODUÇÃO

O computador é há muito tempo uma importante ferramenta para a Física. A nossa sociedade se tornou altamente dependente dos computadores em vários tipos de atividades e serviços. A Física foi um dos ramos precursores na utilização dos computadores nas suas atividades de pesquisa e ensino. Os físicos, entre outros cientistas, foram responsáveis pelo desenvolvimento de inúmeros hardwares e softwares. Da necessidade de ferramentas para auxiliar na busca incessante de solução de problemas, que faz parte do cotidiano do físico, apareceram dispositivos para compor os computadores e programas para utilizá-los. Das aplicações dos computadores na física podemos listar [26]

- Análise Numérica - execução de longos cálculos numéricos com algoritmos determinísticos, tais como solução de sistemas lineares de equações algébricas;
- Manipulação simbólica - execução de regras de inferência para obtenção de resultados de manipulações algébricas;
- Simulação - estudo de sistemas complexos compostos de muitos elementos, a partir da codificação das regras simples de interação entre os mesmos;
- Coleta e análise de dados - aplicação em experimentos onde os dados são coletados automaticamente pelos computadores e também são usados na sua manipulação e análise;
- visualização - representação gráfica de dados que facilita sua interpretação e utilização.

Os computadores são utilizados para testar modelos conceituais e/ou numéricos dos diversos fenômenos estudados na física, e para fazer previsões de resultados de futuros experimentos. Passam a ser um fator importante no ciclo *observar, modelar, prever, observar, ajustar modelo, etc.* que compõe a metodologia de trabalho de um cientista.

Os avanços tecnológicos muitas vezes são originados pelas necessidades da atividade científica e posteriormente passam a influenciar como essa atividade é executada. Nesse contexto, o computador passou a fazer parte das ferramentas básicas dos físicos, quer teóricos ou experimentalistas.

Um exemplo típico dessa relação entre computadores e a física é o Projeto Manhattan. Um breve histórico da *evolução de Calculadoras para Computadores* pode ser vista na página [1]. No início do projeto, em 1943, E. Nelson e S. Frankel faziam os cálculos para o estudo da difusão de nêutrons num arranjo de urânio em Los Alamos, usando calculadoras mecânicas de mesa do tipo Marchant and Friden, idênticas às que eles usavam na Universidade da Califórnia em Berkeley. Havia no entanto um problema: as calculadoras quebravam com frequência em consequência das repetidas operações que eram submetidas. Dana Mitchell, que trabalhava com Oppenheimer, reconhecendo a fragilidade das calculadoras, sugeriu o uso de máquinas de cartão perfurado produzidas pela IBM, que foram usadas para o cálculo de órbitas de planetas por Wallace Eckert na universidade de Columbia. Ao mesmo tempo, Richard Feynman e Nicholas Metropolis, que haviam aprendido a reparar as calculadoras de mesa, organizaram uma competição entre o grupo de pessoas responsáveis pelo cálculo com as calculadoras e as máquinas de cartão perfurado, na execução dos cálculos da simulação de implosões, que eram estudadas pelo grupo de Neddermeyer. O que Feynman e Metropolis fizeram foi organizar as mulheres *computadoras* que operavam as calculadoras de modo que cada uma executava apenas um tipo de operação (multiplicação, soma, etc.) passando o resultado para a pessoa seguinte. Essa organização das operações fez com que o time de mulheres acompanhasse o desempenho das máquinas; mas as pessoas cansam e não podem trabalhar ininterruptamente (como as máquinas podem).

Atualmente, um novo paradigma, composto por *computação em nuvem*, está se consolidando na nossa sociedade como um todo. A idéia por trás de tudo é manter o poder de computação, representado por infra-estrutura de hardware e suporte de operação, em localidades que são acessadas remotamente usando a internet. Desse modo, o usuário não necessita montar e manter seu próprio conjunto de computadores e toda a infra-estrutura associada que é necessária para seu funcionamento, tais como prédios, refrigeração de ambiente, geradores de energia elétrica de suporte, e pessoal de operação e manutenção. O usuário contrata o serviço e paga só pelo que usar.

Para o físico (computacional) poder utilizar a computação em nuvem, é necessário que haja disponibilidade de softwares capazes de realizar os cálculos e simulações dos mo-

delos que ele estuda. Uma grande parte dos modelos físicos lida com equações diferenciais parciais, e é para esse tipo de modelos que iremos abordar a viabilidade e os requisitos necessários para a aplicação da computação em nuvens. Basicamente, necessita-se de softwares que sejam facilmente instalados na *nuvem* e que produzam resultados confiáveis e dos quais possamos aferir a sua qualidade. Neste contexto, estudamos alguns pontos teóricos fundamentais na formulação de muitos métodos numéricos para solução de equações diferenciais parciais, que estão e/ou são facilmente implementados em softwares livres, e mostramos a aplicação desse arranjo computacional para resolução de modelos físicos, especificamente para difusão de calor e escoamento em meios porosos com mais de uma fase fluída. Daremos ênfase às técnicas usadas na formulação numérica para um modelo de modo a encontrar uma forma de minimizar o trabalho do pesquisador. Avaliamos especificamente a aplicação de dois softwares livres, **R** [2] e OpenFOAM [3], para a solução numérica de EDP. Nossa escolha é baseada na disponibilidade, facilidade de instalação nos sítios provedores de computação em nuvem, confiabilidade e qualidade dos métodos e ferramentas já implementadas e facilidade de uso.

O problema típico que focamos nessa Tese é o escoamento multi-fásico em meios porosos e sua solução numérica usando o método de volumes finitos. Estudamos esquemas numéricos de discretização para as equações que representam as leis de conservação em meios porosos juntamente com a Lei de Darcy. Apresentamos também a aplicação da metodologia empregada para a solução de problemas clássicos tais como a difusão em uma e duas dimensões. A tese é composta por este capítulo introdutório e mais 6 outros. No capítulo 2, apresentamos a dedução das equações que regem o escoamento de várias fases fluidas em um meio poroso. No capítulo 3, apresentamos alguns conceitos fundamentais da teoria das equações diferenciais parciais e a formulação de casos específicos de escoamentos. No capítulo 4, estudamos os conceitos fundamentais da discretização de equações diferenciais. No capítulo 5, fazemos aplicações das técnicas para o estudo do fluxo bifásico em meios porosos e por fim as conclusões e considerações finais, bem como propostas de trabalhos futuros. Nos apêndices, apresentamos alguns programas usados em nossas simulações e os trabalhos desenvolvidos com texto em inglês, num formato adequado para a submissão de artigo em revistas científica.

Cabe aqui apresentar alguns motivos para a escolha dos softwares R e OpenFoam como plataforma para nossas aplicações. Os provedores de serviços de computação em nuvens em geral oferecem, entre os sistemas operacionais disponíveis, a opção do Linux em uma (ou várias) de suas distribuições. Como o Linux satisfaz o nosso critério de ser software livre de reconhecida qualidade, as plataformas de software que adotamos de-

vem poder ser usadas nesse sistema operacional. O **R** é uma linguagem de alto nível e também uma plataforma para análise e visualização de dados. Esta linguagem está fartamente documentada (ver [20] e [8] e referências lá encontradas). O software é projeto GNU, desenvolvido sob os termos de licença livre estabelecidos pela *Free Software Foundation's GNU General Public License*, e está disponível na internet tanto em forma de binário e de código fonte. Existe também o software livre RStudio, que é um ambiente integrado de desenvolvimento de aplicações para **R**. Sua documentação está disponível na internet e contém vários pacotes para as mais diversas aplicações de tratamento de dados, desde análise estatística, solução de sistemas lineares a visualização e produção de gráficos de alta qualidade. A funcionalidade do *R* pode ser adequadamente ajustada para as necessidades do usuário através do sistema de carregamento de pacotes específicos. Particularmente interessante para nossas aplicações é o pacote **Matrix** que contém rotinas para manipulação de matrizes e solução de sistemas lineares de equações algébricas, e particularmente sistemas com matrizes esparsas.

Com relação ao OpenFoam, esse é também um software aberto e gratuito e conta com uma ampla gama de características que permite sua utilização no estudo de problemas que vão desde o escoamento de fluidos complexos com ou sem reações químicas, escoamentos turbulentos, transferência de massa e energia, eletromagnetismo, mecânica dos sólidos, e estudo de opções de mercados e bolsas de valores. Seus usuários estão tanto em empresas privadas ou públicas quanto em instituições de ensino e pesquisa. Esse software pode ser facilmente instalado usando pacotes das distribuições de Linux (Ubuntu, SuSE, Red Hat (RHEL)) ou usando códigos fonte.

Nosso objetivo é analisar escoamentos em meios porosos relacionados com a fluidodinâmica de reservatórios de petróleo. Em geral, o petróleo está preso nos poros de uma rocha e irá fluir através da rocha apenas sob influência de uma diferença de pressão muito alta. Os poros, que são espaços no interior da rocha gerados no seu processo de formação, apresentam-se geralmente conectados, permitindo assim o escoamento do fluido ¹. Todavia, os caminhos formados pelas conexões entre os poros são irregulares e retorcidos e por isso ao invés de um fluxo laminar tranquilo a tortuosidade produz instabilidade.

O tipo de rocha apresenta grande influência no fluxo dos fluidos do reservatório. A pressão encontrada nos reservatórios pode ser suficientemente alta, de modo a fazer o petróleo fluir através do meio poroso e sair pelo poço produtor espontaneamente; este tipo de processo de recuperação recebe o nome de **recuperação primária**. Comumente

¹A razão do volume dos poros disponível para o fluxo com relação ao volume total da rocha, é simplesmente denominada porosidade da rocha e é extremamente pequena, variando entre 1% e 20% [11].

a recuperação primária deixa de 70% a 85% do petróleo no reservatório [11].

Para poder recuperar mais petróleo do que no processo de recuperação primária, um fluido tal como água pode ser injetado em alguns poços (poços injetores), enquanto o petróleo é produzido em outros (poços produtores). A injeção de fluido serve tanto para a manutenção da pressão quanto para inundar os poros do meio deslocando o óleo; este tipo de processo de recuperação é denominado **recuperação secundária** e ainda deixa no reservatório cerca de 50% do petróleo.

A taxa de injeção de água não pode ser escolhida de maneira arbitrária, pois a injeção a taxas demasiadamente altas para um dado reservatório pode formar precipitadamente um canal preferencial de escoamento; após este canal ter sido formado, o poço produtor passará a produzir água em demasia, podendo a quantidade de água produzida ser bem maior do que a de óleo.

Para conhecer mais sobre um dado reservatório é preciso realizar simulações computacionais sobre o mesmo. A vantagem das simulações computacionais é que elas apresentam baixo custo financeiro e não comprometem a integridade do reservatório. Uma outra característica é a performance temporal, pois é possível simular o comportamento de centenas de dias de um reservatório gastando-se poucas horas ou até mesmo minutos de computação.

Para simular o comportamento de um reservatório é necessário: (1) conhecer as equações que regem o escoamento em meios porosos, (2) discretizar estas equações para resolvê-las num computador, (3) variar as taxas de injeção e ou parâmetros do reservatório e observar o comportamento do mesmo através de resultados e gráficos.

O foco central deste trabalho é verificar esquemas numéricos que possam ser utilizados em simulações de reservatórios de petróleo, pois softwares comerciais como o CMG apresentam elevado custo anual de licença. Num primeiro estágio, são obtidas as equações do modelo de escoamento bifásico em meios porosos (com o intuito de simular o processo de injeção de água); posteriormente são estudados métodos numéricos para discretizar as equações diferenciais parciais e por último testamos esquemas numéricos para resolver as equações.

Na maioria dos testes realizados ao longo deste trabalho, o software R foi escolhido, principalmente por apresentar uma documentação bastante extensiva. Cada pacote é escrito por usuários distintos (na maioria das vezes), o que permite ao software R uma dinâmica diferenciada. O pacote Matrix disponível no R inclui as rotinas adequadas para

a manipulação de matrizes esparsas e densas e é extensivamente documentado e discutido na web.

1.1 Computação em Nuvem

Nem sempre uma infra-estrutura de hardware está disponível para que as simulações possam ser executadas e mesmo quando dispomos de uma infra-estrutura devemos levar em conta o alto custo para mantê-la, bem como os custos com eletricidade e ou frequência de falhas na rede elétrica. Neste contexto a computação em nuvem surge como uma boa alternativa, permitindo a um usuário realizar computação de qualquer lugar do planeta usando a internet e uma conta em um provedor deste tipo de serviço.

Uma outra motivação é a possibilidade de contratar hardware da mais avançada tecnologia disponível por preços baixos, sem necessitar adquirir esse tipo de hardware, podendo inclusive o usuário contratar mais de uma máquina e interligá-las de modo a formar um cluster do tipo HPC para aplicações de alta performance.



Figura 1: Estrutura típica de computação em nuvem: Um conjunto de hardware com software previamente disponível que pode ser acessado de qualquer terminal.

Na figura (1) temos uma ilustração de estrutura necessária para computação em nuvem. Um terminal conectado a internet acessa de forma remota uma máquina² disponibilizada pelo servidor, em algum lugar da nuvem.

A figura (2) mostra um painel de administração de uma conta de usuário de computação em nuvem no servidor da UolHost. A imagem da máquina virtual pode ser ligada ou desligada e é possível também administrar IPs e portas. Neste servidor o usuário

²A máquina acessada pode ser um computador ou uma máquina virtual.

pode administrar várias máquinas simultaneamente e interligá-las em anel para realizar computação em paralelo.



Figura 2: Janela de administração de uma conta em um provedor de computação em nuvem (UolHost).

Há basicamente duas formas de interagir com a máquina na nuvem: 1 Acessar a máquina via terminal usando o protocolo ssh, conforme o exemplo ilustrado na figura (3); 2 Acessar a máquina virtual usando um programa tipo o FileZilla.

```

root@200-98-200-168: ~
marceliano@marceliano:~$ ssh root@200.98.200.168
root@200.98.200.168's password:
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

System information as of Wed Oct 15 09:51:22 BRT 2014

System load:  0.08          Processes:            234
Usage of /:   15.9% of 44.97GB  Users logged in:     0
Memory usage: 1%           IP address for eth0: 200.98.200.168
Swap usage:  0%

Graph this data and manage this system at:
  https://landscape.canonical.com/

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

root@200-98-200-168:~#

```

Figura 3: Acesso de uma conta em um provedor de computação em nuvem (UolHost), através do terminal por protocolo ssh: Útil para executar comandos e deixar aplicações rodando na nuvem.

O acesso pelo terminal permite instalar aplicações na nuvem colocar programas para rodar e administrar a máquina de forma geral, enquanto o acesso pelo FileZilla (ilustrado

na figura (4)) permite levar arquivos da máquina do usuário para a nuvem e vice-versa.

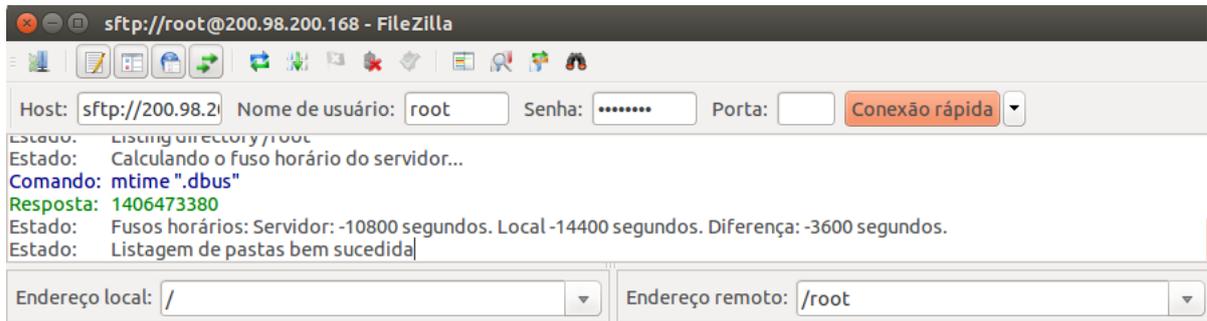


Figura 4: Acesso de uma conta em um provedor de computação em nuvem (UolHost), usando o programa FileZilla: Útil para trazer e levar arquivos da máquina do usuário para a nuvem.

Ao instalar os protocolos ssh e um filezilla o usuário estará pronto para realizar computação em nuvem, mas precisará encontrar um provedor de serviço e criar uma conta neste. A tarifação por uma máquina com a configuração da figura (2) está em torno de R\$0,15 por hora quando a máquina está desligada e R\$1,50 por hora quando a máquina está ligada.

Levando em conta que a tarifação por hora é bem maior quando a máquina está ligada, é útil preparar roteiros para instalar as aplicações necessárias na nuvem, como no exemplo

```
# installOpenFOAMinUbuntu12.04.sh (Roteiro de Instalação - "Execute linha por linha"!)

```

```
# para maiores detalhes consulte o link:

```

```
# http://openfoamwiki.net/index.php/Installation/Linux/OpenFOAM-2.1.1/...

```

```
...Ubuntu#Ubuntu_12.04

```

```
sudo bash

```

```
apt-get install build-essential cmake flex bison zlib1g-dev qt4-dev-tools libqt4-dev

```

```
gnuplot libreadline-dev \ libncurses-dev libxt-dev libopenmpi-dev openmpi-bin

```

```
exit

```

```
cd ~; mkdir OpenFOAM; cd OpenFOAM

```

```
wget "http://downloads.sourceforge.net/foam/OpenFOAM-2.1.1.tgz?use_mirror=..."

```

```

...mesh" -O OpenFOAM-2.1.1.tgz

wget "http://downloads.sourceforge.net/foam/ThirdParty-2.1.1.tgz?use_mirror=...
...mesh" -O ThirdParty-2.1.1.tgz

tar -xzf OpenFOAM-2.1.1.tgz; tar -xzf ThirdParty-2.1.1.tgz

#Ubuntu

uname -m

# i386 - For i386 machines

source $HOME/OpenFOAM/OpenFOAM-2.1.1/etc/bashrc

WM_NCOMPPROCS=2 # number of cores

WM_MPLIB=SYSTEMOPEN

MPI WM_ARCH_OPTION=32

# x64bits - For x64bits machines

source $HOME/OpenFOAM/OpenFOAM-2.1.1/etc/bashrc

WM_NCOMPPROCS=2 # number of cores

WM_MPLIB=SYSTEMOPENMPI

WM_NCOMPPROCS=1

WM_MPLIB=SYSTEMOPENMPI

echo "alias of211='source \$HOME/OpenFOAM/OpenFOAM-2.1.1/...
...etc/bashrc \$FOAM_SETTINGS'"

>> $HOME/.bashrc

# run the alias when you open terminal with command: $ OpenFOAM211

cd OpenFOAM-2.1.1

./Allwmake > make.log 2>&1

/Allwmake > make.log 2>&1

icoFoam -help

kwrite make.log

```

```
tar -czf make.log.tar.gz make.log  
cd $WWM_THIRD_PARTY_DIR  
./makeParaView  
cd $FOAM_UTILITIES/postProcessing/graphics/PV3Readers  
wmSET $FOAM_SETTINGS  
./Allwclean  
./Allwmake
```

O software R já está instalado na nuvem bastando adicionar os pacotes necessários com o comando `install` de dentro do R.

2 CONSERVAÇÃO DE MASSA EM MEIOS POROSOS

Neste capítulo damos um primeiro passo para compreender o comportamento de fluidos escoando através de meios porosos, buscando obter uma equação para a conservação de massa, usando a lei empírica de Darcy e considerações sobre o fluxo através de um volume de controle.

2.1 Lei de Darcy

O engenheiro francês Henry Darcy realizou experimentos em 1856 com o objetivo de compreender melhor o processo de filtração da água através de filtros de areia. Darcy obteve uma lei que descreve o fluxo de um fluido num filtro (meio poroso) em função da diferença de pressão nos seus extremos, da secção transversal do filtro, da viscosidade do fluido e de uma propriedade do meio denominada permeabilidade.

Primeiramente, vamos contextualizar e destacar as grandezas relevantes no escoamento de um fluido através de um meio poroso. A figura (5) é uma ilustração da situação de filtração estudada por Darcy. Em seguida, combinações das grandezas relevantes para o escoamento serão estudadas com uso de análise dimensional e uma destas combinações deve dar origem à formulação matemática da lei de Darcy¹.

¹Os procedimentos aqui efetuados são procedimentos de verificação, a lei de Darcy é bastante conhecida e frequentemente é apenas citada em referências como [19].

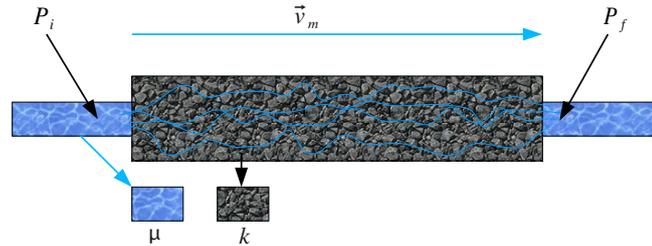


Figura 5: Ilustração para compreensão da lei de Darcy: O tubo horizontal contendo um material de permeabilidade k , é atravessado pelo fluido de viscosidade μ , dando origem a um fluxo Q . A pressão na entrada do tubo é P_i enquanto a pressão na saída do tubo é P_f e a velocidade de escoamento é uma velocidade média do fluido.

O protótipo de um elemento filtrante pode ser pensado como um tubo com área de seção transversal A e comprimento l , sendo atravessado por um fluido de velocidade v_m e submetido a uma diferença de pressão² ΔP_x . Adicionalmente, o líquido e o meio poroso possuem características que afetam o fluxo, sendo estas representadas pelas quantidades μ e k , denominadas viscosidade do líquido e permeabilidade do meio, respectivamente.

O escoamento ilustrado pela figura (5) deixa claro que o seguinte conjunto de grandezas é relevante no estudo do escoamento: $\{\mu, \kappa, l, A, v_m, Q, \Delta P_x\}$ e por isso combinações destas grandezas serão verificadas com uso de análise dimensional.

Para um primeiro estudo, vamos associar as dimensões de Q com as dimensões do conjunto de grandezas³ $\{A, l, \Delta P_x, v_m\}$, tal que

$$[Q] \propto [A]^a [\Delta P_x]^b [v_m]^c [l]^d. \quad (2.1)$$

Cada uma das dimensões em (2.1) pode ser escrita em termos das dimensões fundamentais {massa, comprimento, tempo} = $\{[M], [L], [T]\}$, ou seja

$$[Q] \propto [L^2]^a [ML^{-1}T^{-2}]^b [LT^{-1}]^c [L^1]^d. \quad (2.2)$$

O próximo passo é agrupar as dimensões fundamentais separadamente, usando a pro-

²No caso ilustrado pela figura (5) a diferença de pressão é dada por $\Delta P_x = P_f - P_i$, ou seja, $\Delta P_x = [P(x + \Delta x) + P_{atm}] - [P(x) + P_{atm}]$.

³É comum representar a dimensão de uma grandeza pelo símbolo desta grandeza entre colchetes: $[A]$ = dimensão da grandeza A .

priedade de soma de potências de mesma base,

$$[Q] \propto M^b L^{2a-b+c+d} T^{-2b-c}. \quad (2.3)$$

Como a dimensão de Q é conhecida, $[Q] = L^3 T^{-1}$, então é possível substituir seu valor na equação (2.3), obtendo

$$L^3 T^{-1} M^0 \propto M^b L^{2a-b+c+d} T^{-2b-c}. \quad (2.4)$$

Da equação (2.4) segue finalmente a relação que nos permite obter os valores das constantes,

$$b = 0, \quad 2a - b + c + d = 3, \quad -2b - c = -1, \quad (2.5)$$

portanto $b = 0$, $c = 1$, $a = 1 - \frac{d}{2}$.

Finalmente, substituindo os valores das constantes obtidas de (2.5) na proposta inicial em (2.1) obtém-se,

$$[Q] \propto [A]^{1-\frac{d}{2}} [\Delta P_x]^0 [v_m]^1 [l]^d. \quad (2.6)$$

Notamos em (2.6) a presença de um parâmetro d que não pode ser determinado. Isso ocorre com frequência em análise dimensional e está relacionado com o fato de que apenas a análise dimensional não é suficiente para permitir a escolha da forma correta da lei física envolvida com o fenômeno analisado, restando duas possibilidades: (1) realizar experimentos para determinar a constante faltante ou (2) escolher o valor da constante segundo critérios próprios. No presente caso, uma escolha razoável é de que o fluxo seja diretamente proporcional ao valor da área e, portanto, é necessário que d assumo valor nulo em (2.6), como segue⁴

$$[Q] \propto [A]^1 [\Delta P_x]^0 [v_m]^1 [l]^0. \quad (2.7)$$

A forma obtida em (2.7) traz apenas o conceito simples de fluxo, quando definido em termos do produto da velocidade do fluido pela área atravessada. Isolando a velocidade em (2.7) obtemos

⁴Note de (2.1) que esta escolha é equivalente a propor um fluxo independente do comprimento do tubo.

$$[v_m] \propto \frac{[Q]}{[A]}. \quad (2.8)$$

Finalmente podemos escrever a velocidade em termos do fluxo e da área a menos de uma constante multiplicativa de valor unitário,

$$v_m = \frac{Q}{A}. \quad (2.9)$$

Neste segundo momento, é possível estudar uma outra forma para v_m : ao invés de v_m dependente de Q e de A como em (2.9), pode ser construída uma relação na qual v_m dependa de ΔP_x e l , pois em (2.7) nota-se que essas grandezas não foram levadas em consideração⁵. Duas outras grandezas também não foram ainda estudadas: a permeabilidade k e a viscosidade μ . Portanto é útil reunir essas grandezas na seguinte proposta,

$$[v_m] \propto [\Delta P_x]^a [l]^b [\mu]^c [k]^d. \quad (2.10)$$

Em seguida as quantidades presentes em (2.10), ΔP_x , l , μ e k , devem ser escritas em termos das grandezas fundamentais,

$$[v_m] \propto [ML^{-1}T^{-2}]^a [L]^b [ML^{-1}T^{-1}]^c [L^2]^d. \quad (2.11)$$

A grandeza v_m em (2.11) possui suas dimensões conhecidas, e estas ao serem substituídas em (2.11) dão origem à seguinte equação exponencial,

$$M^0 L T^{-1} \propto M^{a+c} L^{-a+b-c+2d} T^{-2a-c}. \quad (2.12)$$

O passo seguinte é resolver a equação (2.12) de onde obtém-se os seguintes valores para as constantes,

$$a = 1, \quad b = 1 - 2d, \quad c = -1, \quad d = d. \quad (2.13)$$

Finalmente ao substituir (2.13) em (2.10) a proposta fica quase resolvida por inteiro, mas resta ainda uma constante não determinada, como segue,

$$[v_m] \propto [\Delta P_x]^1 [l]^{1-2d} [\mu]^{-1} [k]^d. \quad (2.14)$$

⁵Note os expoentes de ΔP_x e l nulos em (2.7).

A escolha mais razoável em (2.14) é que a velocidade v_m seja diretamente proporcional ao valor da permeabilidade k . Para que isso seja possível, a constante d deve assumir o valor unitário, logo

$$[v_m] \propto [\Delta P_x]^1 [l]^{-1} [\mu]^{-1} [k]^1. \quad (2.15)$$

Notando que $l = x_f - x_i = (x + \Delta x) - x = \Delta x$ na equação acima, é possível obter o valor da velocidade em termos da variação de pressão, a menos de uma constante multiplicativa de valor -1 ,

$$v_m = -\frac{k \Delta P_x}{\mu \Delta x},$$

Lei de Darcy unidimensional

$$v_m = -\frac{k \partial P_x}{\mu \partial x}. \quad (2.16)$$

O sinal negativo da constante multiplicativa deve ser adotado, pois é fato que a pressão decresce no sentido do escoamento, em outras palavras, o sentido da velocidade de escoamento é contrário ao sentido do gradiente de pressão.

A lei de Darcy pode ser generalizada de uma para três dimensões conforme o esquema:

$$P_x(x, y = cte, z = cte) \rightarrow P(x, y, z). \quad (2.17)$$

a mudança proposta em (2.17) será aplicada em (2.16) como segue,

$$v_m = -\frac{k}{\mu} \lim_{\Delta x \rightarrow 0} \left[\frac{P_x(x + \Delta x, y = cte, z = cte) - P_x(x, y = cte, z = cte)}{\Delta x} \right], \quad (2.18)$$

obtendo-se

$$\vec{v}_m = -\frac{k}{\mu} \vec{\nabla} P_x. \quad (2.19)$$

Se a pressão agora pode variar em 3D, é razoável propor que o tubo possa sair da horizontal e ganhar inclinação. Nesse caso existe a necessidade de considerar a contribuição da pressão dada pela componente do peso do líquido relacionada com a inclinação do tubo.

A conhecida pressão hidrostática $P_h = \rho g h$ poderá ser utilizada para construir a

parcela extra usando a equação (2.19), como segue,

$$\vec{v}_m = -\frac{k}{\mu} \vec{\nabla} P + \lim_{\Delta z \rightarrow 0} \frac{k}{\mu} \frac{\Delta P_h}{\Delta z} \hat{z} = -\frac{k}{\mu} \left(\vec{\nabla} P - \lim_{\Delta z \rightarrow 0} \frac{\rho g \Delta z}{\Delta z} \hat{z} \right). \quad (2.20)$$

Note que esta nova parcela tem sinal positivo, pois se o tubo for colocado na vertical à medida que se penetra de cima para baixo no fluido a contribuição da pressão hidrostática aumenta. Simplificando a equação (2.20) ficamos com

$$\vec{v}_m = -\frac{k}{\mu} \left(\vec{\nabla} P - \rho g \hat{z} \right). \quad (2.21)$$

Em (2.21) o vetor unitário \hat{z} pode ser trocado por

$$\vec{\nabla} z = \underbrace{\hat{x} \frac{\partial}{\partial x} z}_{\text{nulo}} + \underbrace{\hat{y} \frac{\partial}{\partial y} z}_{\text{nulo}} + \hat{z} \frac{\partial}{\partial z} z = \hat{z}. \quad (2.22)$$

Substituindo então (2.22) em (2.21),

$$\vec{v}_m = -\frac{k}{\mu} \left(\vec{\nabla} P - \rho g \vec{\nabla} z \right). \quad (2.23)$$

Na ocasião da contribuição gravitacional, a proposição em (2.20) leva em conta o alinhamento do tubo com o eixo Z , mas para um posicionamento arbitrário do sistema de eixos basta introduzir uma função arbitrária $Z(x, y, z)$; isto equivale ao seguinte esquema para a generalização de (2.23)

$$z \rightarrow Z(x, y, z) \equiv Z, \quad (2.24)$$

ou seja

Lei de Darcy tridimensional

$$\vec{v}_m = -\frac{k}{\mu} \left(\vec{\nabla} P - \rho g \vec{\nabla} Z \right). \quad (2.25)$$

A lei de Darcy em sua forma tridimensional é descrita pela equação (2.25), conforme [19] e [7].

Para uma descrição mais geral, a quantidade k deve ser escrita como um tensor⁶ de

⁶Cada componente do tensor é determinada geralmente empiricamente como uma propriedade da interação rocha-fluido, [11].

segunda ordem, tal como segue

$$k = \begin{pmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{pmatrix}. \quad (2.26)$$

O tensor k possui inicialmente 9 componentes. Caso o tensor seja simétrico o número de componentes independentes é reduzido para 6,

$$k = \begin{pmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{xy} & k_{yy} & k_{yz} \\ k_{xz} & k_{yz} & k_{zz} \end{pmatrix}. \quad (2.27)$$

Frequentemente tal tensor é representado apenas em sua forma diagonal⁷, com apenas 3 componentes,

$$k = \begin{pmatrix} k_{xx} & 0 & 0 \\ 0 & k_{yy} & 0 \\ 0 & 0 & k_{zz} \end{pmatrix}, \quad (2.28)$$

que implica em considerar que a permeabilidade é isotrópica, e que as velocidade são alinhadas com o gradiente de pressão total “ $\vec{\nabla}P - \rho g \vec{\nabla}Z$ ”,

$$k = k_{xx} = k_{yy} = k_{zz}. \quad (2.29)$$

2.2 Conservação de massa em meios porosos

Suponha um elemento de volume que represente o meio poroso, com volume total V e densidade ρ , denominado volume de controle. A massa no interior deste volume pode aumentar ou diminuir, devido a presença de fontes ou sorvedouros, respectivamente, conforme ilustrado pela figura (6).

⁷Isto ocorre quando $k_{xy} = k_{xz} = k_{yz} = 0$.

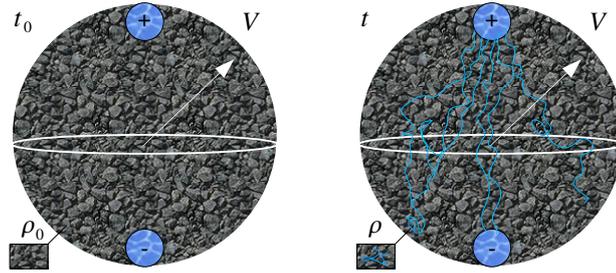


Figura 6: Ilustração para dois instantes distintos de tempo: t_0 e t . Na esquerda no instante t_0 , o volume encontra-se com menos massa de fluido em seu interior, enquanto na direita no instante t , o volume está com uma quantidade maior de fluido em seu interior.

Neste caso, podemos definir uma contribuição para o aumento da massa como

$$\frac{dm_{\pm}}{dt} \equiv \int q dV. \quad (2.30)$$

O fator q é denominado termo de fonte e está associado com a taxa de acréscimo ou decréscimo de massa no interior do volume para fontes ou sorvedouros respectivamente. Note que o índice \pm indica fontes e sorvedouros. Além de fontes e sorvedouros, é possível que o fluido ao atravessar o volume de controle entre em maior quantidade e saia em menor quantidade (conforme ilustração da figura (7)), ou seja, é possível que uma parte do fluido que entra fique retida no interior do volume de controle, implicando num aumento da massa no interior do volume com relação ao tempo. Neste caso o incremento de massa acumulado no interior do volume é dado por

$$\frac{dm_{>}}{dt} = - \int_S \rho \vec{v} \cdot d\vec{A}. \quad (2.31)$$

O índice $>$ usado em (2.31) leva em conta a suposição de que a massa que entra é maior do que a que sai, aumentando a quantidade de massa no interior do volume.

A integral em (2.31) representa uma soma de todas as integrais, uma para cada porção de área, similar ao processo de integração da lei de Gauss para o campo elétrico.

Para o caso de um cubo atravessado em uma das direções por um fluido, a integral em (2.31) resultaria em duas integrais: uma para o fluxo de entrada, outra para o fluxo de saída. No caso de uma superfície de muitas faces ou de geometria não regular, a integração continua válida, pois sempre é possível definir dois contornos, um de entrada (Γ_-) e outro

de saída (Γ_+), onde valem as seguintes relações

$$\vec{v} \cdot d\vec{A} = \begin{cases} \vec{v} \cdot d\vec{A} < 0 & \text{no contorno } \Gamma_-, \\ \vec{v} \cdot d\vec{A} > 0 & \text{no contorno } \Gamma_+. \end{cases} \quad (2.32)$$

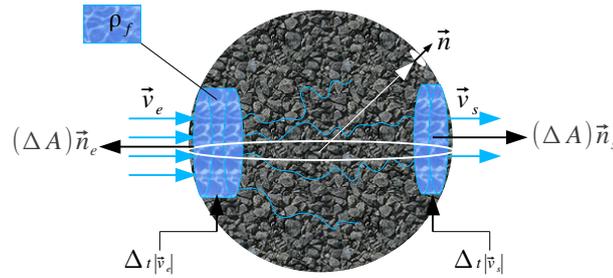


Figura 7: Ilustração de um volume de controle atravessado por fluido: a frente de fluido atinge uma porção de área orientada por um vetor normal a esta. A eficiência tanto da penetração do fluido no volume de controle quanto da saída do mesmo depende do ângulo entre a velocidade de deslocamento do fluido e o vetor de área, sendo máxima quando os referidos vetores são paralelos (saída do volume) ou antiparalelos (penetração no volume) e sendo nula nos casos em que estes vetores são perpendiculares.

O sinal negativo na integral (2.31) é necessário, pois esta integral representa o fluxo de massa que deixa o volume menos o fluxo de massa que entra no volume, enquanto para medir o aumento de massa no volume é necessário medir a quantidade que entra menos a que sai do volume (supondo que a quantidade que entra no volume é maior do que a quantidade que abandona o volume).

Reunindo as contribuições dadas ao aumento de massa no volume de controle, pelas equações (2.30) e (2.31), vem

$$\frac{dm}{dt} = \frac{dm_{>}}{dt} + \frac{dm_{\pm}}{dt} = - \int_S \rho \vec{v} \cdot d\vec{A} + \int q dV, \quad (2.33)$$

o próximo passo consiste em trocar a massa total pela integral da densidade no volume de controle poroso, logo⁸,

$$\frac{d}{dt} \int \rho \phi dV = - \int_S \rho \vec{v} \cdot d\vec{A} + \int q dV, \quad (2.34)$$

⁸Considerando a porosidade ϕ constante.

como o volume de controle é fixo no tempo, a equação (2.34) pode ser convertida em (2.35) ou seja ,

$$\phi \int \frac{\partial \rho}{\partial t} dV + \int_S \rho \vec{v} \cdot d\vec{A} = \int q dV. \quad (2.35)$$

Em (2.35) nota-se a presença de duas integrais de volume e uma de superfície, para que as três sejam integrais de volume basta utilizar o teorema da divergência na integral de superfície.

O teorema da divergência relaciona o divergente de uma função vetorial \vec{M} através de um volume V encerrado por uma superfície fechada S com o fluxo do vetor \vec{M} através da superfície fechada conforme a figura (8), da seguinte maneira

$$\int_V \vec{\nabla} \cdot \vec{M} dV = \oint_S \vec{M} \cdot d\vec{A}. \quad (2.36)$$

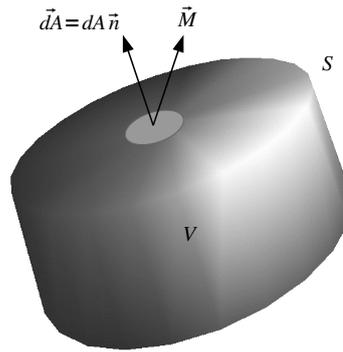


Figura 8: Ilustração para a compreensão do Teorema da Divergência de Gauss: A soma do divergente de um dado campo vetorial \vec{M} , que diverge de um dado volume V , pode ser relacionada diretamente com a integral do elemento $\vec{M} \cdot d\vec{A}$ sobre toda a superfície, pois ambas as contribuições representam o fluxo do campo vetorial \vec{M} através da superfície S que enclausura o volume V .

Usando o teorema da divergência da equação (2.36) na integral (2.35) obtemos finalmente a equação de conservação de massa,

$$\int \phi \frac{\partial \rho}{\partial t} dV + \int \nabla \cdot (\rho \vec{v}) dV = \int q dV, \quad (2.37)$$

$$\int \left[\phi \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) - q \right] dV = \int [0] dV, \quad (2.38)$$

$$\phi \frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \vec{v}) + q. \quad (2.39)$$

Para o escoamento em meios porosos a velocidade pode ser substituída usando a lei de Darcy em sua forma tridimensional, ou seja, substituindo (2.25) em (2.39), [11], [19], [7], [4].

Conservação de massa num volume de controle poroso

$$\phi \frac{\partial \rho}{\partial t} = \vec{\nabla} \cdot \frac{\rho k}{\mu} \left(\vec{\nabla} P - \rho g \vec{\nabla} Z \right) + q. \quad (2.40)$$

A equação de conservação na forma (2.40) será utilizada posteriormente para a obtenção das equações que regem o escoamento bifásico.

3 MODELOS ESPECÍFICOS

Neste capítulo apresentaremos alguns casos específicos da equação (2.37) desenvolvida no capítulo anterior que representa a conservação de massa num escoamento em um meio poroso. Analisaremos os efeitos da compressibilidade do fluido bem como a possibilidade deste conter múltiplas fases. Inicialmente apresentaremos um esquema de classificação das equações diferenciais parciais de segunda ordem, pois o comportamento dessas equações está diretamente ligado ao seu tipo. Da mesma forma, existem esquemas numéricos de solução adequados para cada tipo de equação.

3.1 Classificação de Equações Diferenciais Parciais

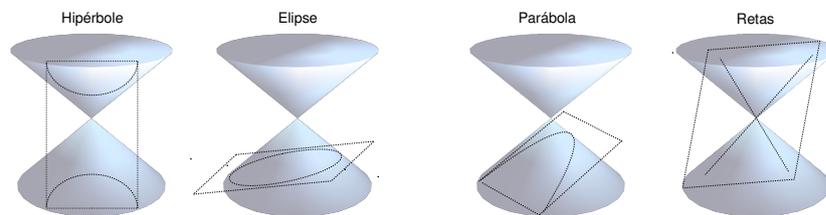


Figura 9: Ilustração das seções cônicas: a inserção de um plano através dos cones origina curvas distintas para diferentes configurações entre o plano e os cones.

Na figura (9), da esquerda para a direita, as seguintes curvas¹ podem ser notadas:

1. Hipérbole

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1, \quad (3.1)$$

¹O desenho de cada curva pode ser realizado num plano cartesiano com auxílio de uma equação correspondente.

2. Elipse

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1, \quad (3.2)$$

3. Parábola

$$y = ax^2 + bx + c, \quad (3.3)$$

4. Retas

$$y = ax + b. \quad (3.4)$$

Uma forma geral de representar estas curvas consiste em agrupar em uma única equação todos os coeficientes e as variáveis x e y em suas formas linear e quadrática, [6]. Tal equação é denominada equação das cônicas e assume a seguinte forma

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0, \text{ para } A \text{ ou } B \text{ ou } C \neq 0. \quad (3.5)$$

É possível com auxílio de matrizes, através de uma forma quadrática e outra linear, representar a equação (3.5) partindo da seguinte relação

$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + c_1 = 0, \quad (3.6)$$

de onde segue imediatamente

$$xa_1x + ya_3x + xa_2y + ya_4y + b_1x + b_2y + c_1 = 0; \quad (3.7)$$

usando a propriedade comutativa do produto torna-se possível escrever

$$a_1x^2 + (a_2 + a_3)xy + a_4y^2 + b_1x + b_2y + c_1 = 0. \quad (3.8)$$

Ao estabelecer uma analogia entre (3.8) e (3.5), obtém-se os coeficientes relacionados em (3.6), de modo que a nova representação

$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} D & E \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + F = 0, \quad (3.9)$$

corresponde à representação geral de uma cônica qualquer:

1. Retas

$$\begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} D & E \end{bmatrix} = \begin{bmatrix} -a & 1 \end{bmatrix}, \quad F = -b, \quad (3.10)$$

2. Parábolas

$$\begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} = \begin{bmatrix} -a & 0 \\ 0 & 0 \end{bmatrix}, \quad [D \quad E] = [-b \quad 1], \quad F = -c, \quad (3.11)$$

3. Elipses

$$\begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} = \begin{bmatrix} \frac{1}{a^2} & 0 \\ 0 & \frac{1}{b^2} \end{bmatrix}, \quad [D \quad E] = [0 \quad 0], \quad F = -1, \quad (3.12)$$

4. Hipérboles

$$\begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} = \begin{bmatrix} \frac{1}{a^2} & 0 \\ 0 & -\frac{1}{b^2} \end{bmatrix}, \quad [D \quad E] = [0 \quad 0], \quad F = -1. \quad (3.13)$$

É possível notar ainda que o determinante da forma quadrática

$$\begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix} = AC - \frac{B^2}{4}, \quad (3.14)$$

assemelha-se ao discriminante de uma equação do segundo grau

$$\Delta = B^2 - 4AC. \quad (3.15)$$

Por comparação entre (3.15) e (3.14) é possível estabelecer

$$\Delta = -4 \begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix}. \quad (3.16)$$

Estudando Δ das equações (3.11)-(3.13) para parábolas, elipses e hipérboles, com auxílio da equação (3.16) nota-se que

$$\Delta_{par.} = -4 \begin{vmatrix} -a & 0 \\ 0 & 0 \end{vmatrix} = 0 \Rightarrow \Delta_{par.} = 0 \{\forall a | a \in \mathfrak{R}\}, \quad (3.17)$$

$$\Delta_{elip.} = -4 \begin{vmatrix} \frac{1}{a^2} & 0 \\ 0 & \frac{1}{b^2} \end{vmatrix} = -4 \frac{1}{a^2 b^2} \Rightarrow \Delta_{elip.} < 0 \{\forall a, b | a, b \in \mathfrak{R}\}, \quad (3.18)$$

$$\Delta_{hip.} = -4 \begin{vmatrix} \frac{1}{a^2} & 0 \\ 0 & -\frac{1}{b^2} \end{vmatrix} = +4 \frac{1}{a^2 b^2} \Rightarrow \Delta_{hip.} > 0 \{\forall a, b | a, b \in \mathfrak{R}\}. \quad (3.19)$$

Finalmente, fica claro que o discriminante Δ guarda informações sobre a natureza geométrica das cônicas.

Seja agora uma equação diferencial parcial a duas variáveis, linear, homogênea e de segunda ordem, do tipo

$$\left[A \frac{\partial^2}{\partial x^2} + B \frac{\partial}{\partial y \partial x} + C \frac{\partial^2}{\partial y^2} + D \frac{\partial}{\partial x} + E \frac{\partial}{\partial y} + F \right] u(x, y) = 0, \quad (3.20)$$

É útil estabelecer uma analogia com as cônicas, e isto pode ser feito com auxílio da representação matricial através da equação (3.6) adaptada para representar os operadores \hat{X} e \hat{Y} que atuam na função $u(x, y)$, logo temos

$$\left[\begin{array}{cc} \hat{X} & \hat{Y} \end{array} \right] \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} + c_1 u(x, y) = 0, \quad (3.21)$$

os operadores \hat{X} e \hat{Y} certamente devem corresponder à operação de derivação parcial, pois só assim a equação (3.21) pode ser equiparada com a (3.20). Fazendo

$$\hat{X} \equiv \frac{\partial}{\partial x} \equiv \partial_x \text{ enquanto } \hat{Y} \equiv \frac{\partial}{\partial y} \equiv \partial_y, \quad (3.22)$$

torna-se possível notar de (3.22) e (3.21) a seguinte forma

$$\begin{aligned} & [\partial_x (a_1 \partial_x) + \partial_y (a_3 \partial_x) + \partial_x (a_2 \partial_y) + \\ & + \partial_y (a_4 \partial_y) + b_1 \partial_x + b_2 \partial_y + c_1] u(x, y) = 0, \end{aligned} \quad (3.23)$$

que é a análoga de (3.7). Porém a derivação será efetuada levando em conta que os coeficientes a_1, a_2, a_3 e a_4 não são necessariamente constantes,

$$\begin{aligned} & [a_1 \partial_{xx} + (a_3 + a_2) \partial_{xy} + (\partial_x a_1 + \partial_y a_2 + b_1) \partial_x + \\ & + a_4 \partial_{yy} + (\partial_x a_2 + \partial_y a_4 + b_2) \partial_y + c_1] u(x, y) = 0. \end{aligned} \quad (3.24)$$

Nota-se que os coeficientes $\{A, B, C, D, E, F\}$ possuem dependência arbitrária de x ou de y na equação abaixo

$$\left[\begin{array}{cc} \hat{X} & \hat{Y} \end{array} \right] \begin{bmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{Y} \end{bmatrix} + F = 0, \quad (3.25)$$

pois ao invés de b_1 e b_2 assumirem respectivamente os valores D e E , aparecem termos adicionais dependentes de derivações dos coeficientes A , B e C ,

$$\partial_x A + \frac{1}{2} \partial_y B + b_1 \equiv D \therefore b_1 = D - \partial_x A - \frac{1}{2} \partial_y B, \quad (3.26)$$

$$\frac{1}{2} \partial_x B + \partial_y C + b_2 \equiv E \therefore b_2 = E - \partial_y C - \frac{1}{2} \partial_x B. \quad (3.27)$$

Apesar disso, a forma quadrática permanece inalterada, o que permite inferir a validade das propriedades algébricas do discriminante Δ estudadas nas equações (3.17)-(3.19), também para a equação diferencial parcial (3.20), ou seja, dada uma equação do tipo

$$\left[A \frac{\partial^2}{\partial x^2} + B \frac{\partial}{\partial y \partial x} + C \frac{\partial^2}{\partial y^2} + D \frac{\partial}{\partial x} + E \frac{\partial}{\partial y} + F \right] u(x, y) = 0, \quad (3.28)$$

onde A , B e C são funções de x e y . Esta pode ser classificada da seguinte forma

- Elíptica: se $B^2 - 4AC < 0$,
- Parabólica: se $B^2 - 4AC = 0$,
- Hiperbólica: se $B^2 - 4AC > 0$.

Exemplos

- Equação do calor

$$\nabla^2 u(x, t) = \frac{1}{k} \frac{\partial u(x, t)}{\partial t},$$

$$\Delta = -4 \begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix} = -4 \begin{vmatrix} 1 & 0 \\ 0 & 0 \end{vmatrix} = 0 \text{ Equação parabólica } (\Delta = 0).$$

- Equação de Laplace

$$\nabla^2 u(x, y) = 0,$$

$$\Delta = -4 \begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix} = -4 \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} = -4 \text{ Equação elíptica } (\Delta < 0).$$

- Equação da onda

$$\nabla^2 u(x, t) = \frac{1}{v^2} \frac{\partial^2 u(x, t)}{\partial t^2},$$

$$\Delta = -4 \begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix} = -4 \begin{vmatrix} 1 & 0 \\ 0 & -\frac{1}{v^2} \end{vmatrix} = \frac{4}{v^2} \text{ Equação hiperbólica } (\Delta > 0).$$

3.2 O Caso Compressível

A equação (2.40) é uma equação diferencial parcial de segunda ordem, com duas variáveis: a pressão P e a densidade ρ . Escrevendo P como função de ρ ou vice versa, é possível obter uma equação diferencial parcial de apenas uma variável. Para que isso seja possível, basta usar adicionalmente, a equação de estado² da compressibilidade isotérmica c ,

Equação de estado: compressibilidade isotérmica

$$c \equiv -\frac{1}{V} \left. \frac{\partial V}{\partial P} \right|_T. \quad (3.29)$$

Considerando a densidade como $\rho = mV^{-1}$, o que implica num volume $V = m\rho^{-1}$ a equação (3.29) pode ser escrita como

$$c = -\frac{d}{dP} \ln(\rho^{-1}) \Big|_T = \frac{1}{\rho} \left. \frac{\partial \rho}{\partial P} \right|_T, \quad (3.30)$$

resolvendo (3.30) vem

$$\int_{P_o}^P c dP = \int_{\rho_o}^{\rho} \frac{d\rho}{\rho}, \quad (3.31)$$

de onde segue

$$P = P_o + \frac{1}{c} \ln\left(\frac{\rho}{\rho_o}\right), \quad (3.32)$$

ou equivalentemente

$$\rho = \rho_o e^{c(P-P_o)}. \quad (3.33)$$

Reescrevendo a equação (2.40) na seguinte forma

$$\phi \frac{\partial \rho}{\partial t} = \vec{\nabla} \cdot \left(\frac{k}{\mu} \rho \vec{\nabla} P - \frac{k}{\mu} g \rho^2 \vec{\nabla} Z \right) + q, \quad (3.34)$$

e estudando o termo $\rho \vec{\nabla} P$ em termos de (3.30) com auxílio da regra da cadeia,

$$\rho \vec{\nabla} P = \left(\frac{1}{c} \frac{\partial \rho}{\partial P} \right) \hat{e}_i \frac{\partial}{\partial x_i} P = \frac{1}{c} \hat{e}_i \frac{\partial \rho}{\partial x_i} = \frac{1}{c} \vec{\nabla} \rho, \quad (3.35)$$

2

Uma equação de estado é uma relação entre as variáveis de estado de um sistema físico; as variáveis de estado são aquelas que descrevem as condições em que um sistema físico se encontra num referido instante de tempo.

obtém-se uma relação que torna possível reescrever (3.34) como uma equação diferencial parcial de apenas uma variável. Isto pode ser feito substituindo (3.35) em (3.34), obtendo-se

Equação para o fluxo de massa^a: fluidos compressíveis

$$\phi \frac{\partial \rho}{\partial t} = \vec{\nabla} \cdot \frac{k}{\mu} \frac{1}{c} \left(\vec{\nabla} \rho - \rho^2 c g \vec{\nabla} Z \right) + q. \quad (3.36)$$

^aObtida usando a equação da continuidade e a lei de Darcy

Uma outra possibilidade é usar a equação (3.30) para obter a derivada temporal da densidade em termos da pressão,

$$\phi \frac{\partial \rho}{\partial t} = c \rho \phi \frac{\partial P}{\partial t}, \quad (3.37)$$

e em seguida usar (3.37) em (3.36) com o objetivo obter uma equação dependente da pressão,

$$c \phi \frac{\partial P}{\partial t} = \frac{1}{\rho} \vec{\nabla} \cdot \left(\frac{k}{\mu} \frac{1}{c} \vec{\nabla} \rho - \rho^2 \frac{k}{\mu} g \vec{\nabla} Z \right) + \frac{q}{\rho}. \quad (3.38)$$

Na sequência o gradiente de ρ de (3.38) pode ser substituído com auxílio de (3.35),

$$c \phi \frac{\partial P}{\partial t} = \frac{1}{\rho} \vec{\nabla} \cdot \left[\rho \left(\frac{k}{\mu} \vec{\nabla} P - \rho \frac{k}{\mu} g \vec{\nabla} Z \right) \right] + \frac{q}{\rho}. \quad (3.39)$$

Para facilitar alguns procedimentos, torna-se útil definir uma função auxiliar \vec{G} , como segue,

$$\vec{G} \equiv \frac{k}{\mu} \vec{\nabla} P - \rho \frac{k}{\mu} g \vec{\nabla} Z. \quad (3.40)$$

Esta função \vec{G} tem o objetivo de simplificar a equação (3.39), que ficará

$$c \phi \frac{\partial P}{\partial t} = \frac{1}{\rho} \vec{\nabla} \cdot \left[\rho \vec{G} \right] + \frac{q}{\rho}. \quad (3.41)$$

O termo $\frac{1}{\rho} \vec{\nabla} \cdot \left[\rho \vec{G} \right]$ da equação (3.41) pode ser reescrito em termos da pressão, pois

$$\frac{1}{\rho} \vec{\nabla} \cdot \left[\rho \vec{G} \right] = \frac{1}{\rho} \vec{G} \cdot \vec{\nabla} \rho + \vec{\nabla} \cdot \vec{G}, \quad (3.42)$$

onde o termo com gradiente da densidade está relacionado na equação (3.35). Então basta substituir (3.35) em (3.42) para obter

$$\frac{1}{\rho} \vec{\nabla} \cdot \left[\rho \vec{G} \right] = \vec{G} \cdot c \vec{\nabla} P + \vec{\nabla} \cdot \vec{G}. \quad (3.43)$$

Finalmente o uso das identidades (3.43) e (3.40) em (3.39) permite obter a equação de conservação de massa em termos da pressão

$$c\phi \frac{\partial P}{\partial t} = c\vec{\nabla}P \cdot \left[\frac{k}{\mu} (\vec{\nabla}P - \rho g \vec{\nabla}Z) \right] + \vec{\nabla} \cdot \left[\frac{k}{\mu} (\vec{\nabla}P - \rho g \vec{\nabla}Z) \right] + \frac{q}{\rho}. \quad (3.44)$$

3.3 Aproximação para fluidos pouco compressíveis

Para fluidos de compressibilidade pequena, compatível com muitos fluidos existentes na natureza, $\rho \sim \rho_0$, é possível expandir (3.33) em série de Taylor em torno de ρ_0 , onde $\rho(P) \sim \rho_0(P_0)$, logo

$$\rho \approx \frac{\rho_0}{0!} (P - P_0)^0 e^{c(P-P_0)} \Big|_{P=P_0} + \frac{\rho_0}{1!} c (P - P_0)^1 e^{c(P-P_0)} \Big|_{P=P_0} + \dots \quad (3.45)$$

Nota-se em (3.45) que aproximação foi truncada, pois termos do tipo c^m , onde $m \geq 2$, não contribuem. Portanto substituindo a equação (3.45) em (3.33) obtém-se

Aproximação para fluidos pouco compressíveis

$$\rho \approx \rho_0 + \rho_0 c (P - P_0). \quad (3.46)$$

O uso da aproximação (3.46) em (3.44) permite obter finalmente a equação que rege o fluxo de massa, no caso de fluidos pouco compressíveis,

Equação para o fluxo de massa: fluidos pouco compressíveis

$$c\phi \frac{\partial P}{\partial t} = c\vec{\nabla}P \cdot \left\{ \frac{k}{\mu} (\vec{\nabla}P - \rho_0 [1 + c(P - P_0)] g \vec{\nabla}Z) \right\} + \\ + \vec{\nabla} \cdot \left\{ \frac{k}{\mu} (\vec{\nabla}P - \rho_0 [1 + c(P - P_0)] g \vec{\nabla}Z) \right\} + \frac{q}{\rho}. \quad (3.47)$$

Para obter a equação que representa o escoamento de fluidos incompressíveis, basta assumir o limite com $c \rightarrow 0$ na equação (3.47),

Equação para o fluxo de massa: fluidos incompressíveis

$$0 = \vec{\nabla} \cdot \left[\frac{k}{\mu} \left(\vec{\nabla} P - \rho_0 g \vec{\nabla} Z \right) \right] + \frac{q}{\rho}. \quad (3.48)$$

3.4 Escoamento bifásico

O escoamento bifásico leva em conta dois fluidos ocupando os volumes disponíveis através do reservatório; vamos admitir um escoamento com fluidos imiscíveis, não há transferência de massa entre eles. Um fluido molha o meio poroso mais do que o outro, e por isso será denominado de fase molhante - “wetting phase” -, o outro fluido é denominado fase não molhante - “nonwetting phase”- .

A saturação de uma fase é a fração do volume poroso por ela ocupada, de modo que temos

$$\underbrace{S_w}_{\frac{V_w}{V}} + \underbrace{S_n}_{\frac{V_n}{V}} = 1. \quad (3.49)$$

Onde S_w é a saturação da fase molhante, S_n é a saturação da fase não molhante, V_w é o volume ocupado pela fase molhante, V_n é o volume da fase não molhante e V é o volume total disponível.

A diferença de pressão entre as fases é a pressão capilar $P_c = P_c(S_n)$. Essa pressão é ocasionada devido a fase molhante aderir ao reservatório mais do que a fase não molhante e é responsável por empurrar os fluidos na direção de escoamento,

$$P_c = P_n - P_w. \quad (3.50)$$

Da lei de Darcy, equação (2.25), podemos obter as velocidades de escoamento para cada fase apenas introduzindo os índices correspondentes:

$$\begin{aligned} \vec{v}_w &= -\frac{k_w}{\mu_w} \left(\vec{\nabla} P_w - \rho_w g \vec{\nabla} Z \right), \\ \vec{v}_n &= -\frac{k_n}{\mu_n} \left(\vec{\nabla} P_n - \rho_n g \vec{\nabla} Z \right). \end{aligned} \quad (3.51)$$

Onde k_w é a permeabilidade da fase w enquanto k_n é a permeabilidade da fase n . Adicionalmente, também são definidas as permeabilidades relativas a cada fase, conforme

segue[11],

$$\begin{aligned} k_{rw} &\equiv \frac{k_w}{k} \leq 1, \\ k_{rn} &\equiv \frac{k_n}{k} \leq 1. \end{aligned} \quad (3.52)$$

Usando a conservação de massa dada pela equação (2.40), e as permeabilidades relativas definidas imediatamente acima, temos,

$$\phi \frac{\partial \rho}{\partial t} = \begin{cases} \vec{\nabla} \cdot \frac{\rho_w k k_{rw}}{\mu_w} (\vec{\nabla} P_w - \rho_w g \vec{\nabla} Z) + \rho_w q_w, \\ \vec{\nabla} \cdot \frac{\rho_n k k_{rn}}{\mu_n} (\vec{\nabla} P_n - \rho_n g \vec{\nabla} Z) + \rho_n q_n. \end{cases} \quad (3.53)$$

A equação acima é uma extensão da equação (2.40): seu lado direito segue os mesmos fundamentos usados na obtenção do lado direito de (2.40) e o lado esquerdo refere-se a toda a massa do reservatório que pode ser decomposto conforme abaixo,

$$\begin{aligned} \phi \frac{\partial \rho}{\partial t} &= \phi \frac{\partial}{\partial t} \left(\frac{m}{V} \right) = \phi \frac{\partial}{\partial t} \left(\frac{m_w + m_n}{V} \right) = \phi \frac{\partial}{\partial t} \left(\frac{\rho_w V_w + \rho_n V_n}{V} \right), \\ \phi \frac{\partial \rho}{\partial t} &= \phi \frac{\partial}{\partial t} \left(\frac{\rho_w V_w + \rho_n V_n}{V} \right) = \phi \frac{\partial}{\partial t} \left(\underbrace{\rho_w \frac{V_w}{V}}_{S_w} + \rho_n \underbrace{\frac{V_n}{V}}_{S_n} \right), \\ \phi \frac{\partial \rho}{\partial t} &= \phi \frac{\partial}{\partial t} (\rho_w S_w + \rho_n S_n) = \begin{cases} \phi \frac{\partial}{\partial t} (\rho_w S_w), \\ \phi \frac{\partial}{\partial t} (\rho_n S_n). \end{cases} \end{aligned} \quad (3.54)$$

Usando a equação (3.54) no lado esquerdo de (3.53) obtemos finalmente as equações de conservação de massa para ambas as fases,

$$\begin{aligned} \phi \frac{\partial}{\partial t} (\rho_w S_w) &= \vec{\nabla} \cdot \frac{\rho_w k k_{rw}}{\mu_w} (\vec{\nabla} P_w - \rho_w g \vec{\nabla} Z) + \rho_w q_w, \\ \phi \frac{\partial}{\partial t} (\rho_n S_n) &= \vec{\nabla} \cdot \frac{\rho_n k k_{rn}}{\mu_n} (\vec{\nabla} P_n - \rho_n g \vec{\nabla} Z) + \rho_n q_n. \end{aligned} \quad (3.55)$$

Desprezando-se os efeitos de gravidade, a equação acima fica apenas

$$\begin{aligned} \phi \frac{\partial}{\partial t} (\rho_w S_w) &= \vec{\nabla} \cdot \frac{\rho_w k k_{rw}}{\mu_w} (\vec{\nabla} P_w - 0) + \rho_w q_w, \\ \phi \frac{\partial}{\partial t} (\rho_n S_n) &= \vec{\nabla} \cdot \frac{\rho_n k k_{rn}}{\mu_n} (\vec{\nabla} P_n - 0) + \rho_n q_n. \end{aligned} \quad (3.56)$$

Levando em conta agora que o escoamento é incompressível, podemos fatorar as densidades na equação acima, logo,

$$\begin{aligned}\phi\rho_w\frac{\partial S_w}{\partial t} &= \rho_w\left[\vec{\nabla}\cdot\frac{k k_{rw}}{\mu_w}\left(\vec{\nabla}P_w\right)+q_w\right], \\ \phi\rho_n\frac{\partial S_n}{\partial t} &= \rho_n\left[\vec{\nabla}\cdot\frac{k k_{rn}}{\mu_n}\left(\vec{\nabla}P_n\right)+q_n\right].\end{aligned}\quad (3.57)$$

Para simplificar a quantidade de termos em (3.57), é útil definir os coeficientes de mobilidade, $\{\lambda_w, \lambda_n\} \equiv \{k_{rw}\mu_w^{-1}, k_{rn}\mu_n^{-1}\}$, logo

$$\begin{aligned}\phi\frac{\partial S_w}{\partial t}-\vec{\nabla}\cdot\left(k\lambda_w\vec{\nabla}P_w\right) &= q_w, \\ \phi\frac{\partial S_n}{\partial t}-\vec{\nabla}\cdot\left(k\lambda_n\vec{\nabla}P_n\right) &= q_n.\end{aligned}\quad (3.58)$$

Somando ambas as equações de (3.58), é possível obter a equação da pressão conforme segue

$$\underbrace{\phi\frac{\partial}{\partial t}\left(\underbrace{S_w+S_n}_1\right)}_0-\vec{\nabla}\cdot\left(k\lambda_w\vec{\nabla}P_w\right)-\vec{\nabla}\cdot\left(k\lambda_n\vec{\nabla}P_n\right)=q_w+q_n.\quad (3.59)$$

Em (3.59) é necessário trocar P_n por $P_c + P_w$ e $\lambda_w + \lambda_n$ por λ_t , para obter a forma final da equação da pressão,

Equação para a Pressão

$$-\vec{\nabla}\cdot\left(\lambda_t k\vec{\nabla}P_w+\lambda_n k\vec{\nabla}P_c\right)=q_w+q_n.\quad (3.60)$$

A outra equação a ser obtida é a equação da saturação. Para obtê-la, basta utilizar a primeira de (3.58) juntamente com a equação (3.61), conforme segue

Equação para a Saturação

$$-\phi\frac{\partial S_n}{\partial t}-\vec{\nabla}\cdot\left(k\lambda_w\vec{\nabla}P_w\right)=q_w.\quad (3.61)$$

Além das duas equações obtidas acima, equação para a pressão e equação para a saturação, será utilizado adicionalmente o modelo de Brooks-Corey, frequentemente encontrado na literatura [5] para descrever as permeabilidades relativas e a pressão capilar como funções não lineares da saturação da fase não molhante. O modelo completo pode ser resumido conforme o quadro abaixo, [9],

Equações do Modelo

- Relações fundamentais:

$$P_c = P_n - P_w, \quad S_w + S_n = 1, \quad \lambda_n = \frac{k_{rn}}{\mu_n}, \quad \lambda_w = \frac{k_{rw}}{\mu_w}. \quad (3.62)$$

- Modelo de Brooks-Corey:

$$P_c(S_n) = P_d(1 - S_n)^{-\frac{1}{\theta}}, \quad k_{rw}(S_n) = (1 - S_n)^{\frac{2+3\theta}{\theta}}, \quad k_{rn}(S_n) = S_n^2 \left[1 - (1 - S_n)^{\frac{2+\theta}{\theta}} \right]. \quad (3.63)$$

- Equações para pressão e saturação:

$$\begin{aligned} -\vec{\nabla} \cdot \left(\lambda_t k \vec{\nabla} P_w + \lambda_n k \vec{\nabla} P_c \right) &= q_w + q_n, \\ -\phi \frac{\partial}{\partial t} S_n - \vec{\nabla} \cdot \left(k \lambda_w \vec{\nabla} P_w \right) &= q_w. \end{aligned} \quad (3.64)$$

As equações da pressão e saturação descritas em (3.64) podem ser classificadas como: elíptica e parabólica respectivamente, [19], pois a equação da pressão pode ser colocada na seguinte forma qualitativa

$$\underbrace{\lambda_t k}_{A} \frac{\partial^2 P_w}{\partial x^2} + \underbrace{\lambda_n k}_{C} \frac{\partial^2 P_w}{\partial y^2} + cte. = 0, \quad (3.65)$$

Para determinar a classe da equação de saturação é necessário levar em conta que a pressão depende da saturação,

$$\underbrace{k \lambda_w \frac{\partial P_w}{\partial S_n}}_A \frac{\partial^2 S_n}{\partial x^2} + \underbrace{k \lambda_w \frac{\partial^2 P_w}{\partial S_n^2}}_D \frac{\partial S_n}{\partial x} + \underbrace{\phi}_{E} \frac{\partial}{\partial t} S_n + cte. = 0. \quad (3.66)$$

Finalmente,

$$\Delta = -4 \begin{vmatrix} A & \frac{B}{2} \\ \frac{B}{2} & C \end{vmatrix} = -4 \begin{vmatrix} k \lambda_w \frac{\partial P_w}{\partial S_n} & 0 \\ 0 & 0 \end{vmatrix} = -4 k \lambda_w \frac{\partial P_w}{\partial S_n} 0 \therefore \Delta = 0. \quad (3.67)$$

A classificação de uma dada equação diferencial parcial é importante, pois as propriedades associadas com a solução de um tipo de equação (parabólica, hiperbólica, elíptica) são preservadas para equações do mesmo tipo, [13]. Portanto é interessante testar as estratégias de solução com uma equação simples e da mesma classe, e posteriormente tentar aplicar estas estratégias na equação mais complexa.

4 FUNDAMENTOS DE DISCRETIZAÇÃO

Neste capítulo uma revisão bibliográfica dos principais métodos de discretização de equações diferenciais parciais é apresentada, com o objetivo de verificar a possibilidade de implementação de determinados esquemas numéricos, na solução de problemas de transporte. Primeiramente são estudados métodos para equações ordinárias e posteriormente métodos utilizados para equações diferenciais parciais, tais como: método de linhas, diferenças finitas, volumes finitos, Galerkin e Galerkin descontínuo.

4.1 Integradores para Equações Diferenciais Ordinárias

Nem sempre é possível resolver uma equação diferencial ordinária analiticamente, simplesmente pelo fato de que algumas equações diferenciais não possuem soluções analíticas. Mesmo nestes casos, é interessante fazer previsões do comportamento dos modelos estudados. Essa necessidade é satisfeita pela solução numérica onde uma tabela de números é obtida ao invés de uma função matemática, permitindo plotar gráficos, os quais podem ser interpretados para avaliar o comportamento dos fenômenos em seus respectivos modelos [16].

Quando uma equação diferencial possui solução analítica, é fácil conhecer o erro da aproximação numérica: basta avaliar a diferença entre a solução analítica e a solução numérica. Este procedimento é útil para testar a validade do integrador e a possibilidade de sua utilização em modelos de regime similar que não possuem solução analítica.

De forma geral é necessário que se estude analiticamente a consistência, estabilidade e convergência do método numérico aplicado a uma dada equação diferencial. Esse tipo de estudo é abundante na literatura. Para exemplos de análise de método de diferenças finitas podemos citar [21], [15].

Uma função analítica nas vizinhanças de um ponto pode ser expandida em séries de Taylor em torno deste¹. A expansão em série de Taylor será o ponto de partida para a construção dos integradores que estudaremos a seguir. O truncamento da série de Taylor em uma quantidade reduzida de termos pode ser utilizado como base para obtenção de esquemas de diferenças finitas [16], [24].

Série de Taylor da função $y(x)$

$$\begin{aligned}
 y(x) &= y(x_0) + (x - x_0) \frac{d}{dx} y(x_0) + \frac{1}{2} (x - x_0)^2 \frac{d^2}{dx^2} y(x_0) + \frac{1}{6} (x - x_0)^3 \frac{d^3}{dx^3} y(x_0) + \dots \\
 &\dots + \frac{1}{n!} (x - x_0)^n \frac{d^n}{dx^n} y(x_0).
 \end{aligned}
 \tag{4.1}$$

4.1.1 Integradores de Euler

Sempre que $(x - x_0)$ for suficientemente pequeno para que os valores de $(x - x_0)^2$ sejam próximos de zero, certamente os valores de $(x - x_0)^n$ para $n \geq 2$ podem ser desprezados. Neste caso a série de Taylor de (4.1) pode ser truncada como

$$y(x) = y(x_0) + (x - x_0) \frac{d}{dx} y(x_0). \tag{4.2}$$

Neste contexto, vamos denotar as quantidades da equação acima como segue:

$$\begin{aligned}
 \{x_0, x, y(x_0), y(x)\} &\equiv \{x_i, x_{i+1}, y_i, y_{i+1}\}, \\
 x_{i+1} &\equiv x_i + h.
 \end{aligned}
 \tag{4.3}$$

O procedimento acima é denominado discretização, onde a função $y(x)$, que era contínua em todo o intervalo de x e podia receber qualquer valor, passará a ser avaliada somente nos pontos $x_n = x_1 + (n - 1)h$, $\forall n \in \mathbb{N}^*$, onde h está definido em (4.3). A partir de então, num dado intervalo de x_1 até x_n teremos exatamente n pares de pontos (x_i, y_i) , onde

$$n = \frac{(x_n - x_1)}{h} + 1. \tag{4.4}$$

¹Quando há uma singularidade, uma série de Laurent deve ser usada para contornar o ponto de não analiticidade.

Substituindo (4.3) em (4.2) obtemos

$$y_{i+1} = y_i + h \left. \frac{d}{dx} y \right|_{x_i} . \quad (4.5)$$

que é denominada integrador de Euler, [16], [24], [12], [17].

Integrador de Euler

$$y_{i+1} = y_i + h \left. \frac{d}{dx} y \right|_{x_i} . \quad (4.6)$$

Para ilustrar o funcionamento do integrador de Euler vamos resolver a equação

$$\frac{d}{dx} y(x) = \sin(x) - \ln(x), \quad y(0.1300) = 0.3200, \quad (4.7)$$

no intervalo de $x = [0.13, 0.14]$. Primeiramente vamos segmentar o intervalo em 5 pontos, e de acordo com (4.4) teremos

$$h = \frac{(x_n - x_1)}{(n - 1)} = \frac{(0.14 - 0.13)}{(5 - 1)} = 0.0025. \quad (4.8)$$

Em seguida basta usar os dados de (4.7) e (4.8) na equação (4.6) para obter

$$y_2 = y_1 + h \left. \frac{d}{dt} y \right|_{x_1} \Rightarrow y_2 = 0.3200 + (0.0025) (\sin(0.1300) - \ln(0.1300)) = 0.3251,$$

$$y_3 = y_2 + h \left. \frac{d}{dt} y \right|_{x_2} \Rightarrow y_3 = 0.3251 + (0.0025) (\sin(0.1325) - \ln(0.1325)) = 0.3301,$$

$$y_4 = y_3 + h \left. \frac{d}{dt} y \right|_{x_3} \Rightarrow y_4 = 0.3301 + (0.0025) (\sin(0.1350) - \ln(0.1350)) = 0.3351,$$

$$y_5 = y_4 + h \left. \frac{d}{dt} y \right|_{x_4} \Rightarrow y_5 = 0.3351 + (0.0025) (\sin(0.1375) - \ln(0.1375)) = 0.3351.$$

- A solução numérica obtida acima pode ser sintetizada na tabela (1)

x_i	y_i
$x_1 = 0.1300$	$y_1 = 0.3200$
$x_2 = 0.1325$	$y_2 = 0.3251$
$x_3 = 0.1350$	$y_3 = 0.3301$
$x_4 = 0.1375$	$y_4 = 0.3351$
$x_5 = 0.1400$	$y_5 = 0.3400$

Tabela 1: Resultado numérico obtido com uso do integrador de Euler da equação (4.6).

- A solução analítica é determinada pela seguinte equação

$$y = y_0 + \cos(x_0) + x_0 \ln(x_0) - x_0 - \cos(x) - x \ln(x) + x.$$

A seguir, podemos observar a precisão do procedimento efetuado acima comparando a solução analítica com a solução numérica, plotando os valores no gráfico (10)

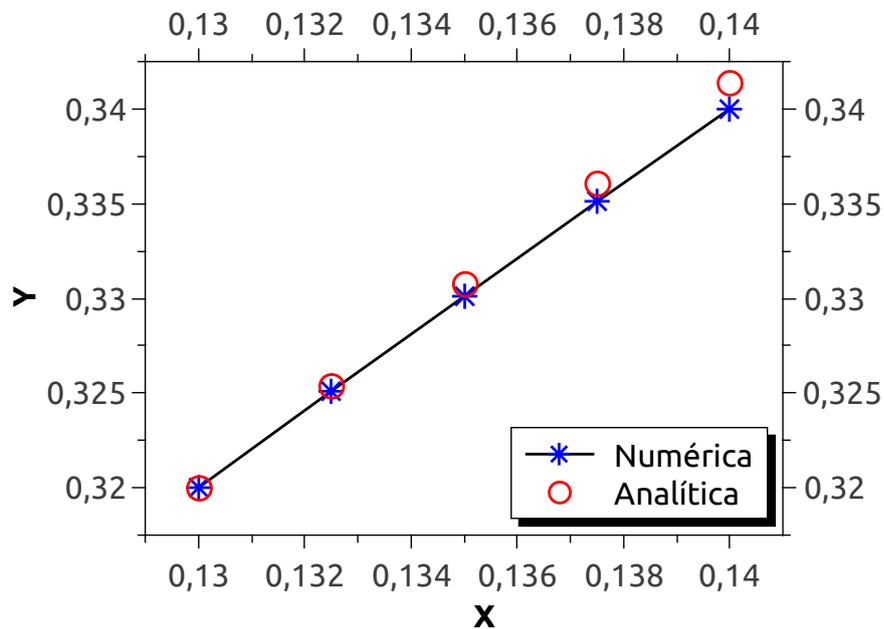


Figura 10: Ilustração do uso do integrador de Euler, quanto mais o integrador avança maior é a diferença entre o valor obtido numericamente e o valor dado pela função analítica.

Pelo gráfico na figura (10) podemos observar que o erro cresce à medida que o integrador avança em x . Neste caso, podemos interpretar que não foi bom desprezar os termos

de ordem superior a 2 na série de Taylor.

Para melhorar a precisão do esquema numérico podemos incluir termos extras além dos termos de primeira ordem. Os termos de segunda ordem podem ser obtidos da derivação da equação diferencial inicial,

$$\frac{d}{dx}y = f(x, y), \quad (4.9)$$

obtendo-se

$$\frac{d^2}{dx^2}y = \frac{\partial}{\partial x}f(x, y) \underbrace{\frac{dx}{dx}}_1 + \frac{\partial}{\partial y}f(x, y) \underbrace{\frac{dy}{dx}}_{f(x,y)}. \quad (4.10)$$

Uma aproximação por série de Taylor, conforme a equação (4.1), truncada apenas para $n \geq 3$, pode ser escrita conforme as expressões abaixo,

$$y(x) = y(x_0) + (x - x_0) \frac{d}{dx}y(x_0) + \frac{1}{2}(x - x_0)^2 \frac{d^2}{dx^2}y(x_0). \quad (4.11)$$

Passando para a notação discreta,

$$y_{i+1} = y_i + (x_{i+1} - x_i) \left. \frac{d}{dx}y \right|_{x_i} + \frac{1}{2}(x_{i+1} - x_i)^2 \left. \frac{d^2}{dx^2}y \right|_{x_i}, \quad (4.12)$$

lembrando que $x_{i+1} - x_i = h$, e usando (4.9) e (4.10) em (4.12) vem,

Integrador com termo de segunda ordem

$$y_{i+1} = y_i + hf(x_i, y_i) + \frac{1}{2}h^2 \left(\frac{\partial}{\partial x}f(x_i, y_i) + f(x_i, y_i) \frac{\partial}{\partial y}f(x_i, y_i) \right). \quad (4.13)$$

A equação (4.13) tem o mesmo propósito da equação (4.6), mas permite obter resultados mais corretos ao levar em conta o termo de segunda ordem.

Baseado na equação (4.7) é possível estabelecer para o exemplo anterior as seguintes relações,

$$f(x, y) = \sin(x) - \ln(x), \quad (4.14)$$

$$\frac{\partial}{\partial x}f(x_i, y_i) = \cos(x) - \frac{1}{x}, \quad \frac{\partial}{\partial y}f(x_i, y_i) = 0; \quad (4.15)$$

em seguida, substituindo (4.14) e (4.15) em (4.13) vem

$$y_{i+1} = y_i + h[\sin(x) - \ln(x)] + \frac{1}{2}h^2 \left[\cos(x) - \frac{1}{x} \right]. \quad (4.16)$$

A equação (4.16) pode ser utilizada para obter uma tabela similar a tabela (1), mas os resultados serão mais precisos. Nos casos em que não há uma integração analítica, é possível obter um erro estimado considerando corretos os valores de y obtidos com termos de ordem superior, enquanto os valores de ordem inferior são considerados valores de y previstos.

Para o exemplo acima, a equação (4.6) seria responsável pela obtenção dos valores previstos de y , denotados como y^p ,

$$y_{i+1}^p = y_i + h \left. \frac{d}{dx} y \right|_{x_i}, \quad (4.17)$$

enquanto a equação que fornece os valores de y considerados corretos, pode ser escrita de acordo com (4.12), conforme segue,

$$y_{i+1}^c = y_i + h \left. \frac{d}{dx} y \right|_{x_i} + \frac{1}{2} h^2 \left. \frac{d^2}{dx^2} y \right|_{x_i}; \quad (4.18)$$

o erro estimado ε ao usar a equação (4.6) pode ser obtido da seguinte forma

$$\varepsilon_{i+1} = y_{i+1}^c - y_{i+1}^p. \quad (4.19)$$

Usando as equações (4.17) e (4.18) em (4.19) obtemos a expressão para o erro estimado,

$$\varepsilon_{i+1} = \frac{1}{2} h^2 \frac{d^2}{dx^2} y_i = \frac{1}{2} h^2 \left(\frac{\partial}{\partial x} f(x_i, y_i) + f(x_i, y_i) \frac{\partial}{\partial y} f(x_i, y_i) \right). \quad (4.20)$$

Para o exemplo da equação (4.7) o erro estimado, calculado pela equação (4.20) é dado pela seguinte expressão

$$\varepsilon_{i+1} = \frac{1}{2} h^2 \left(\cos(x) - \frac{1}{x} \right). \quad (4.21)$$

Anteriormente, para obter a derivada segunda, a função $f(x, y)$ foi utilizada, conforme pode ser notado em (4.14) e (4.15), mas o procedimento de obtenção do integrador de Euler modificado (que inclui a derivada de segunda ordem), pode ser feito inteiramente através do uso de séries de Taylor, de maneira rápida, pois a aproximação

$$y_{i+1} = y_i + h \left. \frac{d}{dx} y \right|_{x_i} + \frac{1}{2} h^2 \left. \frac{d^2}{dx^2} y \right|_{x_i}, \quad (4.22)$$

ao ser derivada produz

$$\left. \frac{d}{dx} y \right|_{x_{i+1}} = \left. \frac{d}{dx} y \right|_{x_i} + h \left. \frac{d^2}{dx^2} y \right|_{x_i} + \dots + \frac{1}{2} h^2 \left. \frac{d^3}{dx^3} y \right|_{x_i}; \quad (4.23)$$

desprezando o termo de terceira ordem, podemos isolar a derivada de segunda ordem de (4.23), logo

$$\frac{d^2}{dx^2} y_i = \frac{1}{h} \left[\left. \frac{d}{dx} y \right|_{x_{i+1}} - \left. \frac{d}{dx} y \right|_{x_i} \right]. \quad (4.24)$$

Ao substituir (4.24) em (4.22) obtemos o integrador de Euler modificado, [16]:

Integrador de Euler modificado

$$y_{i+1} = y_i + \frac{1}{2} h \left[\left. \frac{d}{dx} y \right|_{x_{i+1}} - \left. \frac{d}{dx} y \right|_{x_i} \right]. \quad (4.25)$$

A derivada segunda obtida da série de Taylor em (4.24) também serve para calcular o erro estimado conforme segue,

$$\varepsilon_{i+1} = \frac{1}{2} h^2 \frac{d^2}{dx^2} y_i \simeq \frac{1}{2} h \left[\left. \frac{d}{dx} y \right|_{x_{i+1}} - \left. \frac{d}{dx} y \right|_{x_i} \right], \quad (4.26)$$

a equação acima é útil e pode ser utilizada num método de predição-correção equivalente ao integrador de Euler modificado, porém de implementação mais simples,

Integrador de Euler modificado (predição-correção)

- o valor predito é calculado com o termo de primeira ordem apenas,

$$y_{i+1}^p = y_i + h \left. \frac{d}{dx} y \right|_i, \quad (4.27)$$

- em seguida o erro estimado é calculado usando a derivada segunda com auxílio do valor predito y_{i+1}^p ,

$$\varepsilon_{i+1} = \frac{1}{2} h \left[\left. \frac{d}{dx} y \right|_{x_{i+1}} - \left. \frac{d}{dx} y \right|_{x_i} \right], \quad (4.28)$$

- o valor correto pode ser obtido adicionando o erro estimado ao valor predito,

$$y_{i+1}^c = y_{i+1}^p + \varepsilon_{i+1}. \quad (4.29)$$

O uso das equações (4.27)-(4.29) é simples e baseia-se no fato de que a derivada de $y(x)$ é uma função, $f(x, y)$, conforme (4.9), neste caso

$$y_{i+1}^p = y_i + hf(x_i, y_i), \quad \varepsilon_{i+1} = \frac{1}{2}h[f(x_{i+1}, y_{i+1}) - f(x_i, y_i)], \quad y_{i+1}^c = y_{i+1}^p + \varepsilon_{i+1}. \quad (4.30)$$

4.1.2 Integradores de Runge-Kutta de segunda ordem

A predição de um valor posterior, conforme visto anteriormente para métodos de primeira ordem, pode ser escrita como segue

$$y_{i+1} \simeq y_i + h \left. \frac{d}{dx} y \right|_{x_i}; \quad (4.31)$$

e usando novamente a suposição de que há uma função para a derivada primeira, em concordância com (4.9),

$$\left. \frac{d}{dx} y \right|_{x_i} \equiv f(x_i, y_i), \quad (4.32)$$

então substituindo (4.32) em (4.31) obtemos,

$$y_{i+1} = y_i + hf(x_i, y_i). \quad (4.33)$$

Na equação (4.33) é possível introduzir a notação dos coeficientes de Runge-Kutta, conforme abaixo,

$$k_1 \equiv hf(x_i, y_i), \quad (4.34)$$

$$y_{i+1} = y_i + c_1 k_1. \quad (4.35)$$

Comparando as equações (4.34) e (4.35) com a equação (4.33), é notável que a constante c_1 assume valor unitário.

Para a integração com integrador de segunda ordem de acordo com as equações (4.22)-(4.24), é possível escrever:

$$y_{i+1} = y_i + h \left. \frac{d}{dx} y \right|_{x_i} + \frac{h^2}{2} \left. \frac{d^2}{dx^2} y \right|_{x_i}, \quad (4.36)$$

e

$$y_{i+1} = y_i + h \left. \frac{d}{dx} y \right|_{x_i} + \frac{1}{2}h \left[\left. \frac{d}{dx} y \right|_{x_{i+1}} - \left. \frac{d}{dx} y \right|_{x_i} \right], \quad (4.37)$$

de modo análogo ao executado para levar (4.31) para (4.33), é possível utilizar (4.32) em

(4.37) para obter a equação

$$y_{i+1} = y_i + hf(x_i, y_i) - \frac{1}{2}hf(x_i, y_i) + \frac{1}{2}hf(x_{i+1}, y_{i+1}). \quad (4.38)$$

Na equação acima podemos notar a existência de dois coeficientes de Runge-Kutta,

$$\{k_1, k_2\} \equiv \{hf(x_i, y_i), hf(x_{i+1}, y_{i+1})\}. \quad (4.39)$$

Substituindo (4.39) em (4.38) com constantes arbitrárias a_1, a_2 e a_3 , é possível notar que são necessárias apenas duas constantes arbitrárias, c_1 e c_2 , conforme abaixo,

$$\begin{aligned} y_{i+1} &= y_i + a_1k_1 - a_2k_1 + a_3k_2, \\ y_{i+1} &= y_i + \underbrace{(a_1 - a_2)}_{c_1}k_1 + \underbrace{a_3}_{c_2}k_2, \end{aligned} \quad (4.40)$$

ou seja,

$$y_{i+1} = y_i + c_1k_1 + c_2k_2, \quad (4.41)$$

onde

$$k_1(x_i, y_i) \equiv hf(x_i, y_i), \quad (4.42)$$

e

$$k_2(x_{i+1}, y_{i+1}) = hf(x_{i+1}, y_{i+1}) = htaylor(f(x_i, y_i), x_i, y_i). \quad (4.43)$$

Na equação (4.43) a expansão em série de Taylor² servirá para obter $f(x_{i+1}, y_{i+1})$ nas vizinhanças de $f(x_i, y_i)$, na sequência vamos investigar melhor o coeficiente de Runge-Kutta k_2 .

A série de Taylor para duas variáveis pode ser escrita como segue:

$$\begin{aligned} f(x, y) &= f(x_0, y_0) + \Delta x \partial_x f(x_0, y_0) + \Delta y \partial_y f(x_0, y_0) + \frac{1}{2} \Delta x^2 \partial_{xx} f(x_0, y_0) + \\ &+ \frac{1}{2} \Delta y^2 \partial_{yy} f(x_0, y_0) + \frac{1}{2} 2 \Delta x \Delta y \partial_{xy} f(x_0, y_0), \end{aligned} \quad (4.44)$$

na equação acima os valores Δx e Δy obedecem a seguinte relação:

$$\{\Delta x, \Delta y\} \equiv \{(x - x_0), (y - y_0)\}.$$

²Na equação (4.43) o fator $htaylor(f(x_i, y_i), x_i, y_i)$ é usado para representar a expansão em série de Taylor da função $f(x_i, y_i)$ em torno dos valores (x_i, y_i) .

Mantendo apenas termos de primeira ordem e usando notação discreta vem,

$$f(x_{i+1}, y_{i+1}) = f(x_i, y_i) + \Delta x \partial_x f(x_i, y_i) + \Delta y \partial_y f(x_i, y_i) + \dots \quad (4.45)$$

Extrapolando para um incremento de x bastante pequeno, é possível escrever

$$\frac{d}{dx} y \simeq \frac{\Delta y}{\Delta x} \simeq f(x_i, y_i), \quad (4.46)$$

também é possível considerar que o intervalo Δx é um múltiplo de h , ou seja,

$$\Delta x \equiv c_3 h, \quad (4.47)$$

e reunindo (4.47) e (4.46) obtém-se

$$\Delta y \equiv \underbrace{c_3 h}_{\Delta x} f(x_i, y_i). \quad (4.48)$$

O próximo passo é substituir (4.46)-(4.48) em (4.45) para obter

$$f(x_{i+1}, y_{i+1}) = f(x_i, y_i) + \underbrace{[c_3 h]}_{\Delta x} \partial_x f(x_i, y_i) + \underbrace{[c_3 h f(x_i, y_i)]}_{\Delta y} \partial_y f(x_i, y_i) + \dots, \quad (4.49)$$

finalmente, ao usar (4.49) e (4.43), é possível obter uma forma explícita para k_2 ,

$$k_2(x_{i+1}, y_{i+1}) = h f(x_{i+1}, y_{i+1}) = f(x_i, y_i) h + c_3 [\partial_x f(x_i, y_i) + f(x_i, y_i) \partial_y f(x_i, y_i)] h^2, \quad (4.50)$$

na sequência é necessário substituir a equação (4.50) em (4.41), como segue

$$\begin{aligned} y_{i+1} &= y_i + c_1 k_1 + c_2 k_2 = y_i + c_1 k_1(x_i, y_i) + c_2 k_2(x_{i+1}, y_{i+1}), \\ y_{i+1} &= y_i + c_1 h f(x_i, y_i) + c_2 f(x_i, y_i) h + \\ &\quad + c_2 c_3 [\partial_x f(x_i, y_i) + f(x_i, y_i) \partial_y f(x_i, y_i)] h^2, \\ y_{i+1} &= [y_i] h^0 + [(c_1 + c_2) f(x_i, y_i)] h^1 + \\ &\quad + [c_2 c_3 \partial_x f(x_i, y_i) + c_2 c_3 f(x_i, y_i) \partial_y f(x_i, y_i)] h^2. \end{aligned} \quad (4.51)$$

É interessante notar que a equação (4.51) oferece um integrador de segunda ordem similar ao obtido em (4.13); o integrador em (4.13) também pode ser organizado em potências de h , conforme abaixo,

$$y_{i+1} = [y_i] h^0 + [f(x_i, y_i)] h^1 + \left[\frac{1}{2} \partial_x f(x_i, y_i) + \frac{1}{2} f(x_i, y_i) \partial_y f(x_i, y_i) \right] h^2. \quad (4.52)$$

A seguir, vamos organizar em uma tabela o integrador do tipo Runge-Kutta da equação (4.51) com o integrador da equação acima:

Coeficientes	h^0	h^1	h^2
Runge-Kutta	y_i	$(c_1 + c_2) f(x_i, y_i)$	$c_2 c_3 \partial_x f(x_i, y_i) + c_2 c_3 f(x_i, y_i) \partial_y f(x_i, y_i)$
Segunda Ordem	y_i	$f(x_i, y_i)$	$\frac{1}{2} \partial_x f(x_i, y_i) + \frac{1}{2} f(x_i, y_i) \partial_y f(x_i, y_i)$

Tabela 2: Na tabela acima estão listadas duas equações apenas e três constantes a determinar, escolhendo uma das três constantes as outras duas são determinadas.

Da tabela (2), obtemos as seguintes relações entre as constantes:

$$(c_1 + c_2) = 1, \quad c_2 c_3 = \frac{1}{2}, \quad (4.53)$$

nas relações acima, a escolha de uma constante e a determinação das outras duas permite obter uma família de métodos de Runge-Kutta, [16], [24].

- Escolhendo a constante c_1 :

$$c_1 = \frac{1}{2}, \quad (4.54)$$

- como consequência da escolha de c_1 , obrigatoriamente temos que c_2 e c_3 devem valer:

$$c_2 = \frac{1}{2}, \quad c_3 = 1, \quad (4.55)$$

- neste caso os coeficientes de Runge-Kutta k_1 e k_2 serão

$$k_1 = hf(x_i, y_i), \quad (4.56)$$

$$k_2 = hf(x_{i+1}, y_{i+1}) = hf(x_i + \Delta x, y_i + \Delta y), \quad (4.57)$$

$$k_2 = hf(x_i + c_3 h, y_i + c_3 hf(x_i, y_i)), \quad (4.58)$$

$$k_2 = hf(x_i + h, y_i + hf(x_i, y_i)). \quad (4.59)$$

Finalmente uma equação para y_{i+1} pode ser obtida pelo método de Runge-Kutta, em consequência da terna $\{c_1, c_2, c_3\} = \{(1/2), (1/2), 1\}$:

$$\begin{aligned} y_{i+1} &= y_i + c_1 k_1 + c_2 k_2, \\ y_{i+1} &= y_i + c_1 h f(x_i, y_i) + c_2 h f(x_i + c_3 h, y_i + c_3 h f(x_i, y_i)), \\ y_{i+1} &= y_i + \frac{1}{2} h f(x_i, y_i) + \frac{1}{2} h f(x_i + h, y_i + h f(x_i, y_i)), \end{aligned}$$

a equação acima é o integrador de Euler modificado.

- Para os valores $\{c_1, c_2, c_3\} = \{0, 1, (1/2)\}$, o integrador obtido é dado pela seguinte equação

$$\begin{aligned} y_{i+1} &= y_i + c_1 h f(x_i, y_i) + c_2 h f(x_i + c_3 h, y_i + c_3 h f(x_i, y_i)), \\ y_{i+1} &= y_i + 0 h f(x_i, y_i) + 1 h f\left(x_i + \frac{1}{2} h, y_i + \frac{1}{2} h f(x_i, y_i)\right), \\ y_{i+1} &= y_i + h f\left(x_i + \frac{1}{2} h, y_i + \frac{1}{2} h f(x_i, y_i)\right), \end{aligned}$$

a equação acima é o integrador do ponto médio.

4.1.3 Integradores de Runge-Kutta de Terceira Ordem ou Superior

Os integradores de Runge-Kutta de terceira ordem ou superior podem ser obtidos de maneira similar aos de ordem inferior, para isso basta estender a proposta feita em (4.41), conforme abaixo

$$y_{i+1} = y_i + c_1 k_1 + c_2 k_2 + c_3 k_3 + \dots + c_n k_n, \quad (4.60)$$

um estudo sistemático das séries de Taylor similar ao feito previamente, permite obter os coeficientes de Runge-Kutta conforme segue, [16]

$$k_1 = h f(x_i, y_i), \quad (4.61)$$

$$k_2 = h f(x_i + c_4 h, y_i + c_4 h f(x_i, y_i)), \quad (4.62)$$

$$k_3 = h f(x_i + c_5 h, y_i + c_6 k_1 + (c_5 - c_6) k_2), \quad (4.63)$$

neste caso as relações para as constantes arbitrárias ficam:

$$c_1 + c_2 + c_3 = 1, \quad (4.64)$$

$$c_2 c_4 + c_3 c_5 = \frac{1}{2}, \quad (4.65)$$

$$c_2 c_4^2 + c_3 c_5^2 = \frac{1}{3}, \quad (4.66)$$

$$c_3 (c_5 - c_6) c_4 = \frac{1}{6}. \quad (4.67)$$

- Para os valores: $\{c_1, c_2, c_3, c_4, c_5, c_6\} = \{(2/8), (3/8), (3/8), (2/3), (2/3), 0\}$, obtém-se o integrador de Nystrom, [16], conforme abaixo:

$$y_{i+1} = y_i + \frac{2}{8}k_1 + \frac{3}{8}k_2 + \frac{3}{8}k_3, \quad (4.68)$$

$$k_1 = f(x_i, y_i) h, \quad (4.69)$$

$$k_2 = f\left(x_i + \frac{2}{3}h, y_i + \frac{2}{3}k_1\right), \quad (4.70)$$

$$k_3 = f\left(x_i + \frac{2}{3}h, y_i + \frac{2}{3}k_2\right). \quad (4.71)$$

4.1.4 Alguns Testes com os Integradores Clássicos

Para verificar a acurácia dos integradores mais comumente utilizados, vamos resolver um problema simples, cuja solução analítica é conhecida e comparar a solução numérica com a analítica graficamente, como segue,

- equação diferencial

$$\frac{d}{dt}V = V\lambda e^{-\alpha t}, \quad (4.72)$$

- solução analítica conhecida

$$V(t) = V_0 e^{\frac{\lambda}{\alpha}(1-e^{-\alpha t})}, \quad (4.73)$$

- valor inicial e constantes

$$\{V_0, \lambda, \alpha\} = \{1, 1, 1\}. \quad (4.74)$$

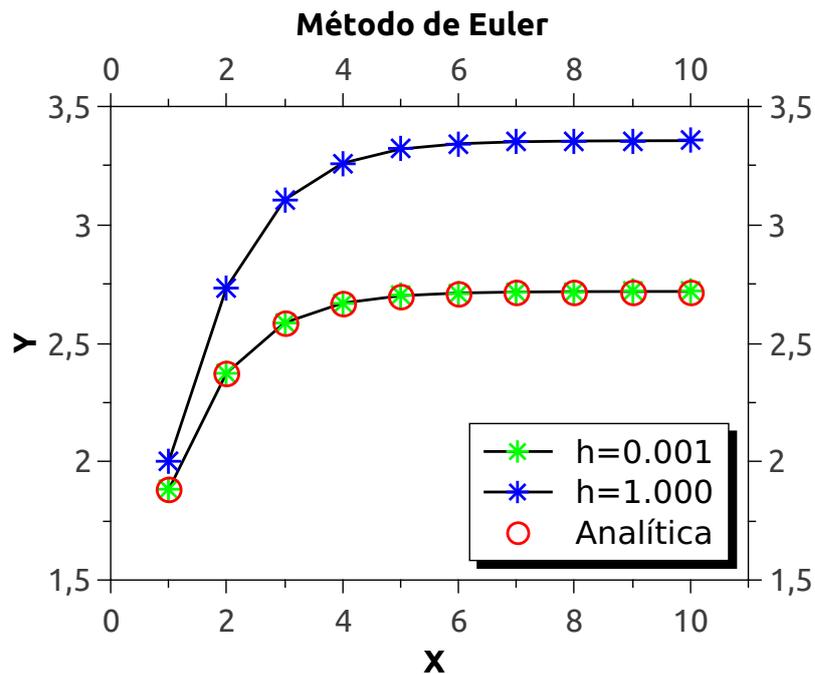


Figura 11: Resolução numérica da equação (4.72) usando o integrador de Euler para dois diferentes valores de h . É possível notar que para h suficientemente pequeno, o resultado é satisfatório, do contrário o resultado difere do analítico de maneira expressiva.

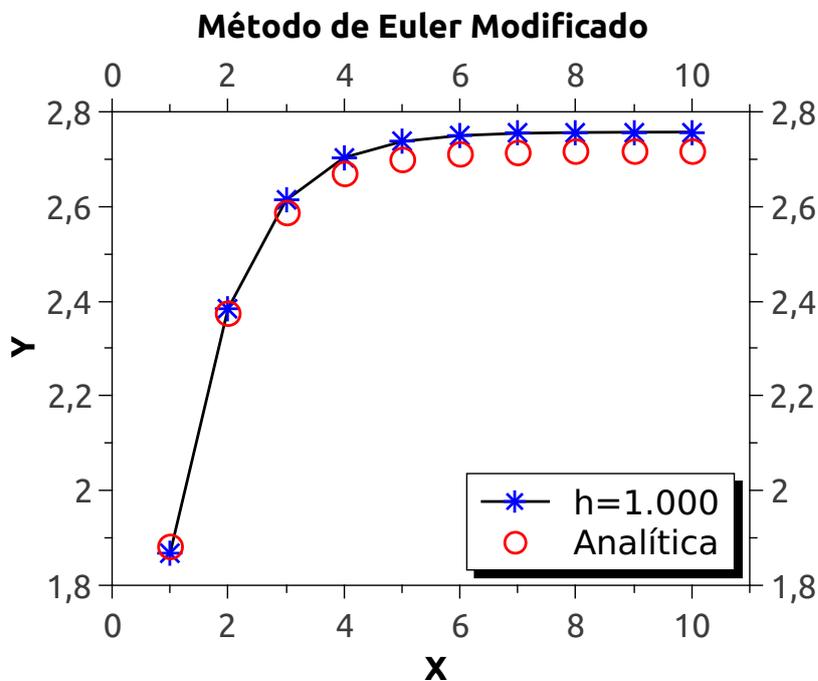


Figura 12: Resolução numérica da equação (4.72) usando o integrador de Euler modificado, é possível ver o ganho de performance comparando a solução para $h = 1$ na figura acima com a solução para $h = 1$ da figura anterior.

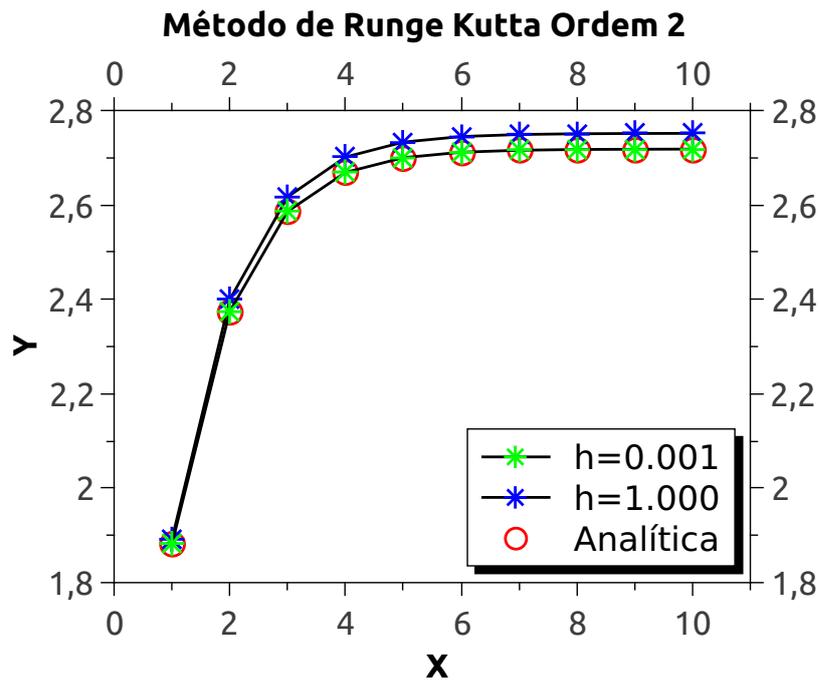


Figura 13: Resolução numérica da equação (4.72) usando o integrador de Runge-Kutta de segunda ordem, é possível ver o ganho de performance comparando a solução para os valores de h na figura acima com a solução para os valores de h da figura (11).

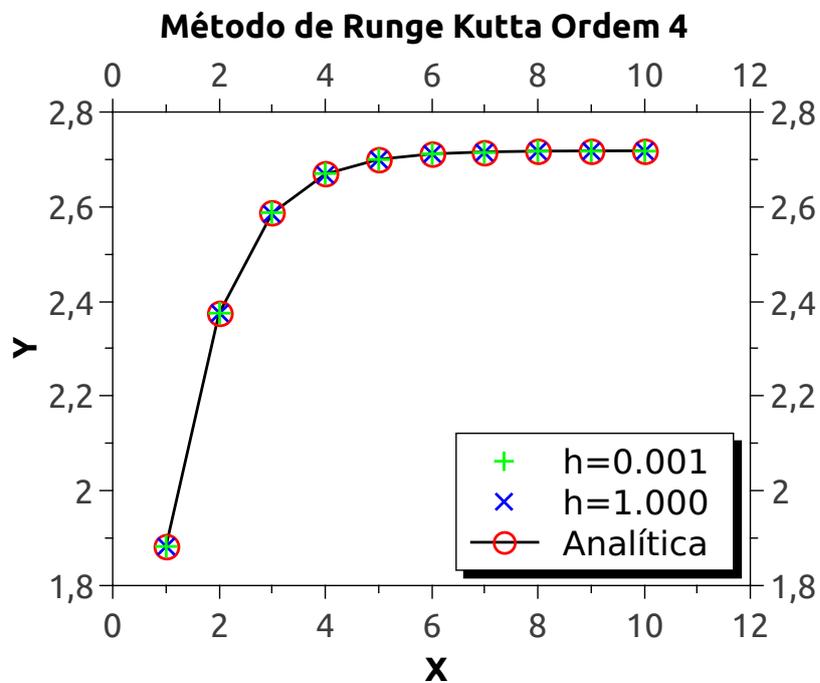


Figura 14: Resolução numérica da equação (4.72) usando o integrador de Runge-Kutta de quarta ordem, é possível ver o ganho de performance comparando a solução para os valores de h na figura acima com a solução para os valores de h da figura (13).

4.1.5 Sistema de duas EDOs

A aplicação de integradores para resolver equações diferenciais parciais é possível, pelo método de linhas [16], onde a equação diferencial parcial é substituída por um sistema de equações diferenciais ordinárias com várias equações através de um processo conhecido como discretização. Portanto, é útil verificar a utilização dos integradores por exemplo, para resolver um sistema simples de duas equações diferenciais ordinárias, conforme segue, [16],

- sistema com duas equações diferenciais ordinárias,

$$\begin{aligned}\frac{d}{dx}y_1(x) &= y_2, \\ \frac{d}{dx}y_2(x) &= -y_1,\end{aligned}\tag{4.75}$$

- condições iniciais

$$\{y_1(0), y_2(0)\} = \{0, 1\},\tag{4.76}$$

- soluções analíticas conhecidas

$$\begin{aligned}y_1 &= \sin(x), \\ y_2 &= \cos(x).\end{aligned}\tag{4.77}$$

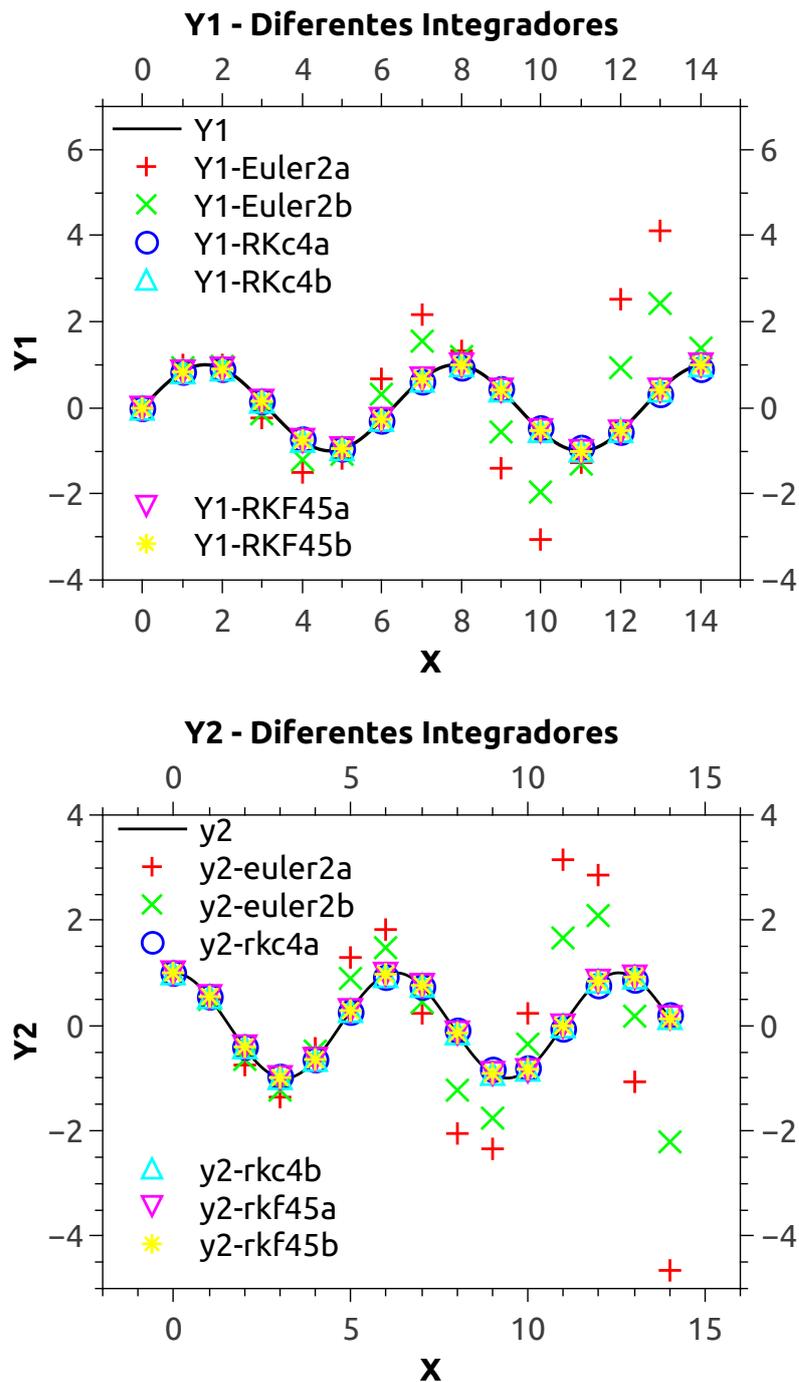


Figura 15: Resolução numérica da equação (4.75) para ambas as funções, y_1 acima e y_2 abaixo, podemos verificar o desempenho dos integradores de Euler de 2 ordem (euler2a,euler2b), dos integradores de Runge-Kutta de 4 ordem (rk4a,rk4b) e dos integradores de Runge-Kutta-Fehlberg de ordem 4-5 (rkf45a,rkf45b).

4.1.6 Seleção automática de integradores

Ao integrar uma equação diferencial ordinária é fundamental escolher adequadamente o integrador; isto é feito usando critérios de análise numérica. Na atualidade, o pacote de integração do software R, denominado deSolve [25], permite integrar equações diferenciais de forma automática, o próprio pacote escolhe qual integrador será utilizado, com base em critérios de integração previamente definidos.

Como exemplo da praticidade do uso deste pacote, vamos integrar algumas EDOs abaixo:

- Equação diferencial:

$$\frac{d}{dx}y + 2yx = 0, \quad (4.78)$$

- Solução analítica conhecida:

$$y(x) = e^{-x^2}, \quad (4.79)$$

- Condição inicial:

$$y(0) = 0.0183, \quad (4.80)$$

```
# Criando a sequência de valores no eixo x:
x<-seq(-2, 2, 0.1 )
# Criando o vetor y, que guarda as condições iniciais a serem usadas nas equações dife-
renciais:
y<-c(y=0.0183)
# Relacionando as equações do modelo:
model<-function(x,y,parms){
y=y[1]
dy<-2*y*x
list(dy)
}
# Obtendo a solução da EDO:
# carregando os integradores: lsoda, vode, daspk, euler, rk4, ode23, ode45...;
library(deSolve)
out<-ode(y, x, model, parms)
```

O integrador lsoda é assumido como padrão, este integrador automaticamente escolhe um dos integradores do pacote deSolve baseado no regime de integração para que não haja

divergência na integração, conforme pode ser observado após a integração digitando-se no console o comando `diagnostics(out)`:

```

> diagnostics(out)

----- lsoda return code -----
return code (idid) = 2 Integration was successful.
----- INTEGER values -----
1 The return code : 2
2 The number of steps taken for the problem so far: 49
3 The number of function evaluations for the problem so far: 101
5 The method order last used (successfully): 6
6 The order of the method to be attempted on the next step: 6
7 If return flag =-4,-5: the largest component in error vector 0
8 The length of the real work array actually required: 36
9 The length of the integer work array actually required: 21
14 The number of Jacobian evaluations and LU decompositions so far: 0
15 The method indicator for the last succesful step, 1=adams (nonstiff), 2= bdf (stiff): 1
16 The current method indicator to be attempted on the next step, 1=adams (nonstiff),
2= bdf (stiff): 1
----- RSTATE values -----
1 The step size in t last used (successfully): 0.1
2 The step size to be attempted on the next step: 0.1
3 The current value of the independent variable which the solver has reached: 2.024218
4 Tolerance scale factor > 1.0 computed when requesting too much accuracy: 0 5
The value of t at the time of the last method switch, if any: -2

```

- Resultado da integração:

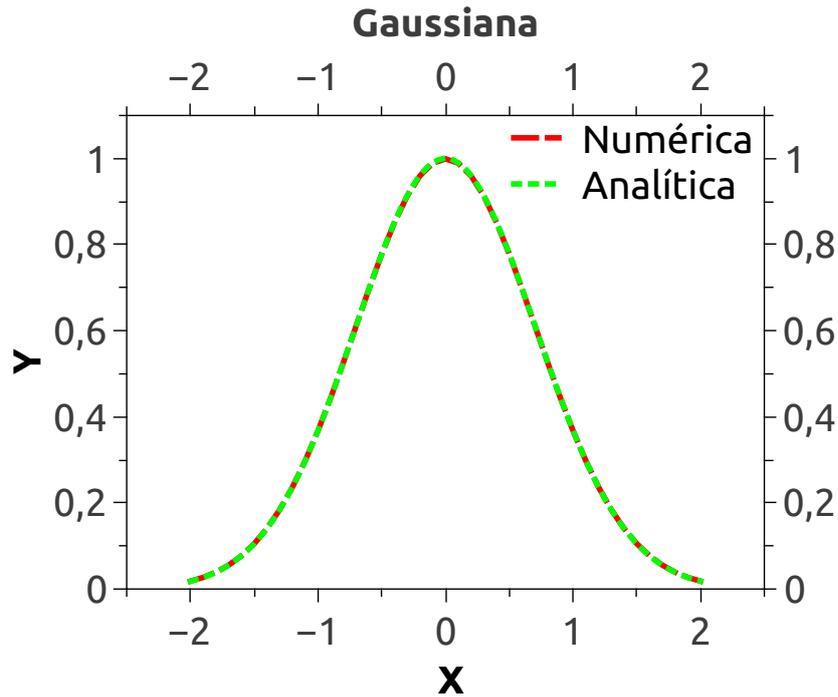


Figura 16: Solução numérica da equação (4.78), comparada com a solução analítica da equação (4.79), é possível notar que a solução numérica é perfeitamente compatível com a solução analítica.

4.2 Diferenças Finitas e Método de Linhas

Para resolver equações diferenciais parciais é necessário transformá-las em sistemas algébricos. Há várias maneiras de construir um sistema algébrico equivalente, dentre eles método de linhas, método de diferenças finitas, método de volumes finitos e método de elementos finitos.

Os métodos de diferenças finitas substituem as diferenciais parciais por diferenças finitas convenientemente escolhidas. Um outro tipo de discretização, também da classe de diferenças finitas, é o método de linhas, onde todas as derivadas parciais menos uma são discretizadas por diferenças finitas. Esta última derivada não discretizada pode ser resolvida automaticamente pelos integradores de EDO previamente discutidos.

As seguintes aproximações para derivação são frequentemente utilizadas nos mais diferentes esquemas de diferenças finitas, [24], [16]:

- Aproximação de derivada para frente: “Forward”

$$\frac{\partial y(x)}{\partial x} \approx \frac{y(x+h) - y(x)}{h} = \frac{y_{i+1} - y_i}{h}, \quad (4.81)$$

- Aproximação de derivada para trás: “Backward”

$$\frac{\partial y(x)}{\partial x} \approx \frac{y(x) - y(x-h)}{h} = \frac{y_i - y_{i-1}}{h}, \quad (4.82)$$

- Aproximação de derivada central: “Leap Frog”

$$\frac{\partial y(x)}{\partial x} \approx \frac{y(x+h) - y(x-h)}{2h} = \frac{y_{i+1} - y_{i-1}}{2h}, \quad (4.83)$$

- Aproximação de segunda ordem central: “Central Difference”

$$\frac{\partial^2 y(x)}{\partial x^2} \approx \frac{y(x+h) - 2y(x) + y(x-h)}{h^2} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}, \quad (4.84)$$

Vamos ilustrar estes procedimentos com o uso da equação do calor, onde u é a temperatura e α é a difusividade térmica, conforme segue,

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad u(0, t) = 0, \quad \begin{cases} u(x, 0) = \sin(\pi x), \\ u(0, t) = 0, \quad T(1, t) = 0. \end{cases} \quad (4.85)$$

a equação (4.85) possui a seguinte solução analítica: $u(x, t) = \sin(\pi x)e^{-\pi^2 t}$, assim podemos descontar a solução numérica da analítica e observar o erro durante a solução, conforme segue.

- Discretização pelo método de linhas: (neste caso, apenas o lado direito será discretizado)

$$\frac{\partial u_i}{\partial t} = \alpha \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}, \quad (4.86)$$

$$\text{ sistema } \left\{ \begin{array}{l} \vdots \\ \vdots \\ \frac{\partial u_2}{\partial t} = \alpha \frac{u_3 - 2u_2 + u_1}{h^2} \\ \frac{\partial u_3}{\partial t} = \alpha \frac{u_4 - 2u_3 + u_2}{h^2} \\ \frac{\partial u_4}{\partial t} = \alpha \frac{u_5 - 2u_4 + u_3}{h^2} \\ \vdots \\ \vdots \end{array} \right. \quad (4.87)$$

rodando o programa `heat.equation.1.R` do apêndice A, que implementa um sistema como o referido na equação (4.87) para um total de 20 equações obtemos

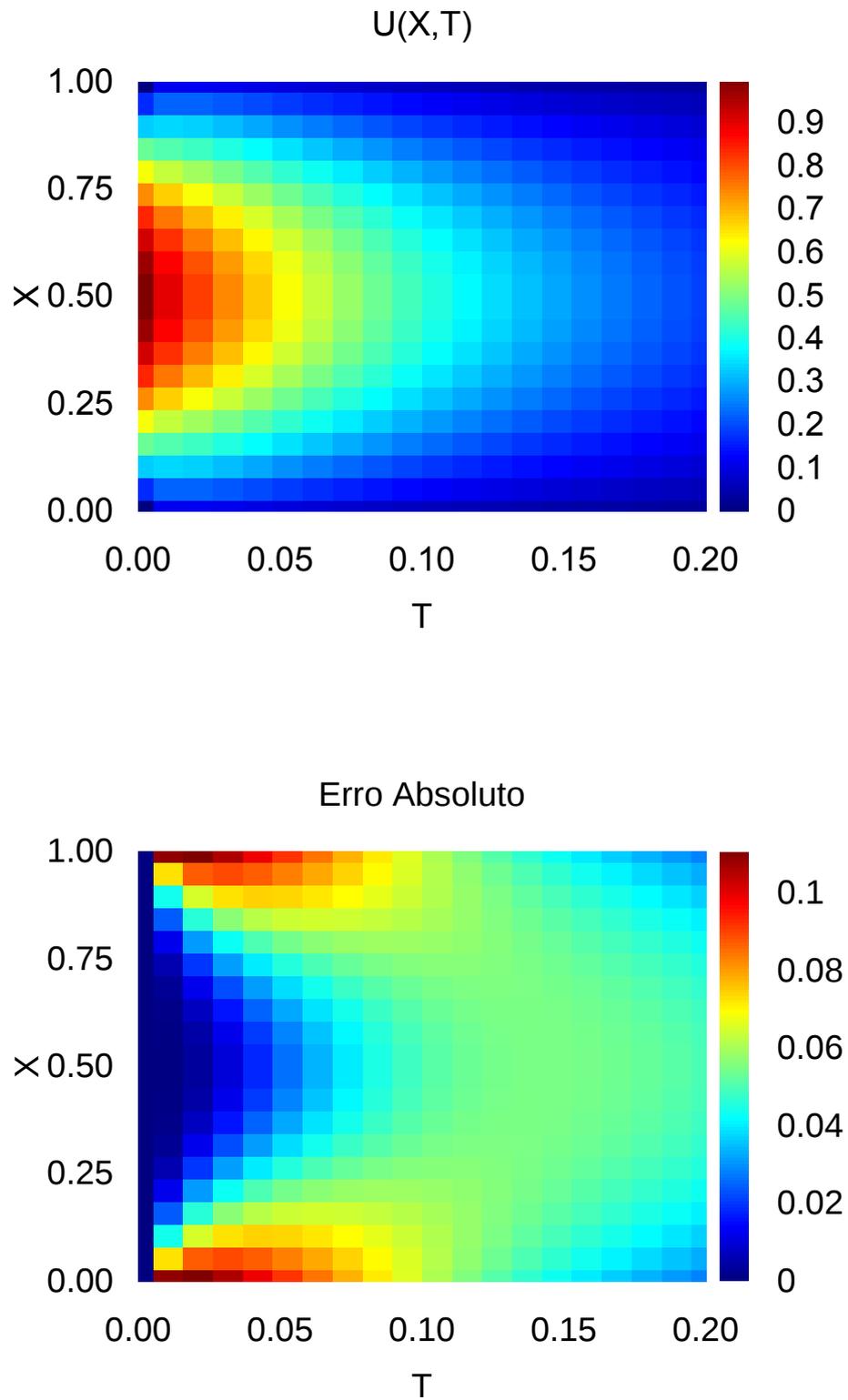


Figura 17: Superior: solução numérica obtida pelo programa `heat.equation.1.R`, inferior: erro absoluto associado, obtido descontando-se a solução numérica da analítica.

No programa `heat.equation.1.R` (ver apêndices) a inserção das 20 equações de forma manual é exaustiva e é possível notar que quanto mais equações maior a precisão da solução. No R podemos criar N equações através de um loop apresentado no programa `heat.equation.2.R`. Para uma quantidade maior de equações, podemos comparar a qualidade do resultado com o anterior baseando-se nas imagens das soluções apresentadas na figura (18).

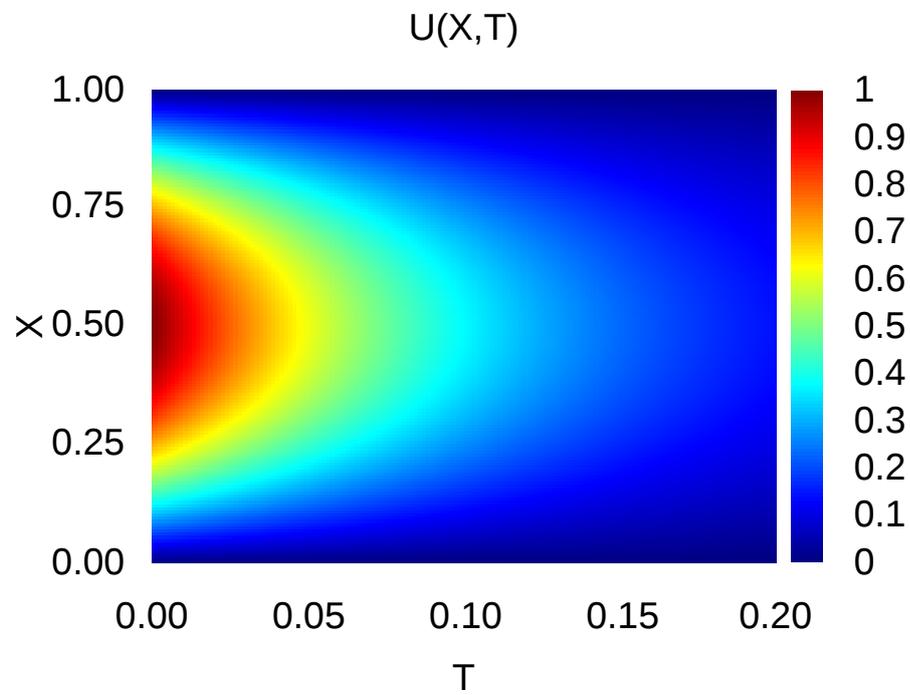
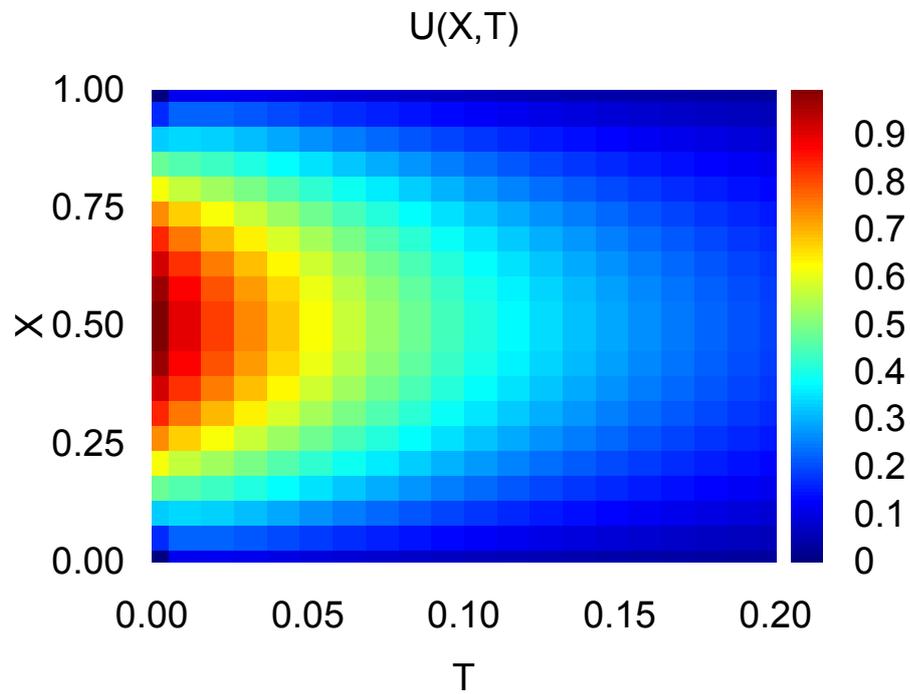


Figura 18: podemos notar uma baixa resolução na figura superior, correspondente ao método de linhas aplicado para as 20 equações que foram resolvidas conforme (4.87); contra uma alta resolução na figura inferior, onde foram resolvidas 150 equações.

Conforme é possível notar no procedimento de discretização pelo método de linhas executado acima, uma das derivadas (na ocasião, a derivada temporal), é resolvida pelo integrador automatizado, onde o algoritmo de integração foi escolhido internamente.

4.3 Método de Galerkin (Elementos Finitos)

Os problemas estudados na área de engenharia de estruturas deram origem ao método de elementos finitos e levou cerca de uma década antes deste método ser reconhecido como uma forma do problema de Rayleigh-Ritz [14]. A relação entre estas duas técnicas surge ao considerar a forma variacional do problema. Por exemplo, o funcional quadrático, [14],

$$\mathcal{F}(u) = \int_0^1 \left[p(x) (u'(x))^2 + q(x) (u(x))^2 - 2f(x) u(x) \right] dx, \quad (4.88)$$

possui um mínimo com respeito a variação em $u(x)$ dado pela equação de Euler. Isto pode ser visto tomando um incremento na equação (4.88), ou seja obtendo o termo $\mathcal{F}(u + \delta u)$ conforme abaixo

$$\int_0^1 \left[p(x) (u'(x) + \delta u'(x))^2 + q(x) (u(x) + \delta u(x))^2 - 2f(x) (u(x) + \delta u(x)) \right] dx,$$

em seguida calcula-se a variação $\delta\mathcal{F}(u) = \mathcal{F}(u + \delta u) - \mathcal{F}(u)$, levando em conta a seguinte aproximação: $(\delta u'(x))^2 \approx 0$, $(\delta u(x))^2 \approx 0$, logo

$$\delta\mathcal{F}(u) = \int_0^1 \left[p(x) 2u'(x) \delta u'(x) + q(x) 2u(x) \delta u(x) - 2f(x) \delta u(x) \right] dx. \quad (4.89)$$

Um caminho natural para comparar os termos em (4.89) é reescrever $\delta u'(x) = \frac{d}{dx} \delta u(x)$ explicitando assim a importância do fator $\delta u(x)$,

$$\frac{\delta\mathcal{F}(u)}{2} = \underbrace{\int_0^1 p(x) u'(x) \frac{d}{dx} \delta u(x) dx}_{\text{Integrar por partes}} + \int_0^1 [q(x) u(x) - f(x)] \delta u(x) dx. \quad (4.90)$$

Integrando o primeiro termo do segundo membro da equação (4.90) vem,

$$I(x) = \underbrace{p(x) u'(x) \delta u(x)}_{\substack{0 \\ \text{(extremidades fixas)}}} \Big|_0^1 - \int_0^1 \delta u(x) \frac{d}{dx} (p(x) u'(x)) dx; \quad (4.91)$$

na sequência, ao substituir (4.91) em (4.90) e minimizar o funcional assumindo $\delta \mathcal{F}(u) = 0$ segue finalmente

$$0 = \int_0^1 \underbrace{\left[-\frac{d}{dx} (p(x) u'(x)) + q(x) u(x) - f(x) \right]}_{\text{nulo}} \delta u(x) dx. \quad (4.92)$$

De (4.92) é possível notar que

$$-\frac{d}{dx} \left(p(x) \frac{d}{dx} u(x) \right) + q(x) u(x) = f(x). \quad (4.93)$$

O processo de obtenção da equação (4.93) torna claro que ao invés de resolver a equação diferencial (4.93) um processo alternativo consiste em encontrar $u(x)$ que minimiza o funcional $\mathcal{F}(u)$ da equação (4.88). A ideia de Rayleigh-Ritz é aproximar a solução por um número finito de funções,

$$u(x) = \sum_i^N q_i \Phi_i(x), \quad (4.94)$$

e determinar os pesos q_i que minimizam o funcional $\mathcal{F}(u)$ da equação (4.88). No método de elementos finitos a solução é aproximada por um número finito de funções tipicamente locais, em essência opostas às funções globais usadas no método de Rayleigh-Ritz [14].

O ponto de partida do método de elementos finitos é a equação diferencial, que pode ser colocada numa formulação integral (formulação de Galerkin), de tal modo que este problema pode ser reduzido a um sistema algébrico que pode ser resolvido numericamente. A conexão entre os dois métodos fica clara ao perceber que a forma integral do método de elementos finitos é a mesma do funcional do método de Rayleigh-Ritz.

O método dos resíduos ponderados ilustra como a escolha de diferentes pesos na forma integral ou forma fraca da equação pode ser usada para construir muitos métodos numéricos. Para descrever o método dos resíduos ponderados vamos considerar a equação

diferencial linear num domínio Ω denotada por

$$\mathbb{L}(u) = 0, \quad (4.95)$$

submetido a condições iniciais e de contorno apropriadas. Supondo que a solução $u(x, t)$ pode ser representada com precisão pela seguinte aproximação, [14],

$$u^\delta(x) = u_0(x) + \sum_{i=1}^{N_{dof}} u_i \Phi_i(x), \quad (4.96)$$

onde $\Phi_i(x)$ são funções analíticas denominadas funções de expansão, enquanto u_i são os N_{dof} ³ coeficientes desconhecidos e $u_0(x)$ é selecionado de maneira a satisfazer as condições de contorno. A equação (4.96) é uma aproximação finita, logo, ao ser substituída em (4.95), surge naturalmente um resíduo não nulo, como consequência do truncamento em (4.96), tal que

$$\mathbb{L}(u^\delta) = R(u^\delta). \quad (4.97)$$

O formato da aproximação é dado por (4.96), mas não existe uma única forma de determinar os coeficientes u_i . Para fazê-lo podemos impor restrições sobre o resíduo R , gerando equações integrais denominadas de formulação fraca. Uma forma de introduzir este tipo de restrição é através do conceito de produto interno (f, g) definido no domínio Ω como

$$(f, g) = \int_{\Omega} f(x) g(x) dx. \quad (4.98)$$

A restrição colocada sobre o resíduo é que seu produto interno por uma função peso $v_j(x)$ seja nulo,

$$(v_j(x), R) = 0 = \int_{\Omega} v_j(x) R dx, \quad j = 1, \dots, N_{dof}. \quad (4.99)$$

Este tipo de restrição gera a partir de (4.97) um sistema de equações diferenciais ordinárias em u_i , e quando a equação original (4.96) não possui dependência no tempo os coeficientes u_i podem ser determinados diretamente da solução do sistema algébrico de equações.

O resíduo $R(x)$ tende a zero na medida que $u^\delta(x)$ se aproxima da solução exata $u(x)$, assim como $N_{dof} \rightarrow \infty$. Todavia, a natureza do esquema é determinada pela escolha das funções de expansão $\Phi_i(x)$ e das funções peso v_j . Uma lista das funções peso mais usadas e o respectivo método que as mesmas produzem pode ser visto na tabela abaixo:

³A constante N_{dof} representa o número de graus de liberdade (number of degrees of freedom).

Funções peso	Tipo de método
$v_j(x) = \delta(x - x_j)$	Colocação
$v_j(x) = \begin{cases} 1 & \text{no interior de } \Omega^j \\ 0 & \text{no exterior de } \Omega^j \end{cases}$	Volumes Finitos
$v_j(x) = \frac{\partial R}{\partial u_j}$	Mínimos-Quadrados
$v_j(x) = x^j$	Método dos Momentos
$v_j(x) = \Phi_j$	Galerkin
$v_j(x) = \Psi_j (\neq \Phi_j)$	Petrov-Galerkin

Finalmente vamos considerar o método de Galerkin. Neste método as funções de expansão escolhidas são as mesmas funções peso $v_j = \Phi_j$. Em geral os métodos espectrais usam um conjunto de funções de expansão globais, tal que as $\Phi_i(x)$ estão definidas através do domínio da solução (por exemplo as funções $\sin(x)$ e $\cos(x)$). O método de elementos finitos usa um conjunto de funções de expansão $\Phi_i(x)$ que são definidas localmente, numa região finita. Na expansão de elementos finitos essas regiões são construídas por um pavimento que não se sobrepõe, sobre toda a região do domínio, tal como um quadrado subdividido em vários triângulos onde cada triângulo compartilha suas fronteiras com seus vizinhos não sendo permitida a sobreposição de triângulos.

Para descrever como formular o problema de Galerkin, vamos considerar como ponto de partida uma equação diferencial parcial e ilustrar as propriedades deste tipo de formulação. Nosso problema exemplo será a equação de Poisson, [14],

$$\mathbb{L}(u) \equiv \nabla^2 u + f = 0. \quad (4.100)$$

Esta equação surge em muitas áreas da física, como fluxo irrotacional de fluido e condução de calor, e ainda problemas envolvendo potenciais gravitacional e elétrico. Em uma dimensão a equação (4.100) fica

$$\mathbb{L}(u) \equiv \frac{\partial^2}{\partial x^2} u + f = 0. \quad (4.101)$$

Para que este problema esteja adequadamente colocado e tenha soluções únicas, precisamos especificar condições de contorno, se considerarmos uma solução num domínio $\Omega = \{x | 0 < x < 1\}$, então temos que considerar as seguintes condições de contorno,

$$u(0) = g_{\mathcal{D}}, \quad \frac{\partial u}{\partial x}(1) = g_{\mathcal{N}}. \quad (4.102)$$

Onde $g_{\mathcal{D}}$ e $g_{\mathcal{N}}$ são constantes para o problema em uma dimensão; a condição de Dirich-

let $g_{\mathcal{D}}$ é referida como condição de contorno essencial, já a condição de Neumann $g_{\mathcal{N}}$ é referida como condição de contorno natural, como veremos na formulação de Galerkin as condições necessitam ser especificadas explicitamente enquanto as condições de Neumann são tratadas implicitamente como parte da formulação.

Efetuando o produto interno de (4.100) por $v(x)$ (nula em todos os contornos de Dirichlet $\partial\Omega_{\mathcal{D}}$), com a integração efetuada no domínio Ω , segue

$$(v, \mathbb{L}(u)) = \int_0^1 v \left(\frac{\partial^2}{\partial x^2} u + f \right) dx = 0, \quad (4.103)$$

se a função u for aproximada por u^δ então nota-se facilmente com auxílio de (4.97) que a equação (4.103) é equivalente a equação (4.99); estudando a integração em (4.103), obtemos

$$I \equiv \underbrace{\int_0^1 v \frac{\partial^2}{\partial x^2} u dx}_{I_A} + \underbrace{\int_0^1 v f dx}_{I_B} = 0, \quad (4.104)$$

resolvendo I_A por partes vem

$$I_A = \int_0^1 v \frac{\partial^2}{\partial x^2} u dx = \left[v \frac{\partial}{\partial x} u \right]_0^1 - \int_0^1 \frac{\partial}{\partial x} u \frac{\partial}{\partial x} v dx. \quad (4.105)$$

O próximo passo é reunir (4.103), (4.104) e (4.105) para obter

$$\int_0^1 \frac{\partial}{\partial x} u \frac{\partial}{\partial x} v dx = \underbrace{\left[v \frac{\partial}{\partial x} u \right]_0^1}_{K_0} + \int_0^1 v f dx, \quad (4.106)$$

em seguida podemos estudar melhor o termo K_0 em (4.106),

$$K_0 = \left[v \frac{\partial}{\partial x} u \right]_0^1 = v(1) \underbrace{\left. \frac{\partial}{\partial x} u \right|_1}_{g_{\mathcal{N}}} - \underbrace{v(0)}_{0 \text{ em } \partial\Omega_{\mathcal{D}}} \left. \frac{\partial}{\partial x} u \right|_0 = v(1) g_{\mathcal{N}}, \quad (4.107)$$

de (4.107) em (4.106) vem

$$\int_0^1 \frac{\partial}{\partial x} u \frac{\partial}{\partial x} v dx = \int_0^1 v f dx + v(1) g_{\mathcal{N}}; \quad (4.108)$$

nestes últimos passos ficou claro como a condição de contorno de Neumann foi naturalmente incluída na formulação. A aproximação de Galerkin do problema dado pela equação (4.101) é a solução da forma fraca dada por (4.108) quando a solução exata $u(x)$ é aproximada por uma expansão finita denotada por $u^\delta(x)$. A função $v(x)$ em (4.108) também

é trocada por uma expansão finita, denotada por $v^\delta(x)$, e assim a equação (4.108) fica

$$\int_0^1 \frac{\partial}{\partial x} u^\delta \frac{\partial}{\partial x} v^\delta dx = \int_0^1 v^\delta f dx + v^\delta(1) g_N, \quad (4.109)$$

as funções u^δ são referidas como funções de expansão, enquanto v^δ é denominada função peso. Na aproximação de Galerkin o mesmo conjunto de funções que é usado para representar v^δ é também usado para representar a solução u^δ . Mas como a função peso é definida como zero num contorno de Dirichlet, fica clara então a necessidade da função u^δ ser composta por mais de uma função, deste modo

$$u^\delta = u^H + u^D, \quad (4.110)$$

onde

$$u^H(\partial\Omega_D) = 0, \quad u^D(\partial\Omega_D) = g_D, \quad (4.111)$$

substituindo (4.111) em (4.109) obtemos

$$\int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^H dx = \int_0^1 v^\delta f dx + v^\delta(1) g_N - \int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^D dx. \quad (4.112)$$

Esta última equação pode ser resolvida como um sistema algébrico, desde que todos os termos da direita sejam conhecidos e u^H e v^δ tenham um número finito de funções. Notadamente a formulação de Galerkin reduz um problema diferencial a um sistema algébrico que pode ser resolvido em um computador.

No método de elementos finitos para equações diferenciais de segunda ordem, construímos u^δ a partir de uma classe de funções que é dita ser C^0 , nesta classe as funções são contínuas em todo o domínio Ω , não valendo o mesmo para as derivadas destas funções. O domínio da solução Ω é subdividido em elementos finitos denotados por Ω^e , nestes domínios a derivada das funções é contínua, apesar de não ser contínua necessariamente na fronteira entre os elementos, uma vez que as funções sobre o domínio todo são C^0 contínuas.

Observando o integrando da equação (4.109) podemos nos valer de uma série sobre todos os elementos Ω^e para calcular a integração, se houverem N_{el} elementos então obtemos

$$\int_0^1 \frac{\partial}{\partial x} u^\delta \frac{\partial}{\partial x} v^\delta dx = \sum_{e=1}^{N_{el}} \int_{\Omega^e} \frac{\partial}{\partial x} u^\delta \frac{\partial}{\partial x} v^\delta dx, \quad (4.113)$$

é interessante notar que isto equivale a trocar a área abaixo de uma curva por uma soma

de segmentos de área ao longo desta mesma curva (conforme o significado geométrico de uma integral). Integrando por partes a integral da série em (4.113) vem,

$$\sum_{e=1}^{N_{el}} \int_{\Omega^e} \frac{\partial}{\partial x} u^\delta \frac{\partial}{\partial x} v^\delta dx = \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e}^{\Omega_R^e} - \sum_{e=1}^{N_{el}} \int_{\Omega^e} v^\delta \frac{\partial^2}{\partial x^2} u^\delta dx, \quad (4.114)$$

ou seja, de (4.113) e (4.114) obtemos

$$\int_0^1 \frac{\partial}{\partial x} u^\delta \frac{\partial}{\partial x} v^\delta dx = - \int_0^1 v^\delta \frac{\partial^2}{\partial x^2} u^\delta dx + \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e}^{\Omega_R^e}, \quad (4.115)$$

onde os símbolos Ω_L^e e Ω_R^e denotam respectivamente o valor de x na extremidade esquerda e direita do domínio Ω^e . Conforme a figura abaixo:

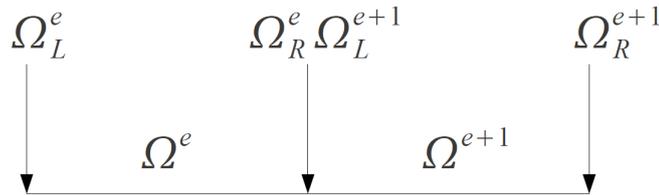


Figura 19: Decomposição de domínio por elementos: nota-se que a direita “Right” do elemento e coincide com a esquerda “Left” do elemento $e + 1$.

Se a função for do tipo C^1 , ou seja pertencente a uma classe cuja função e sua derivada primeira são contínuas na região da interface, conforme ilustrado pela figura acima, é necessário que os valores das derivadas de primeira ordem sejam iguais,

$$v^\delta \frac{\partial}{\partial x} u^\delta \Big|_{\Omega_R^e} = v^\delta \frac{\partial}{\partial x} u^\delta \Big|_{\Omega_L^{e+1}}, \quad (4.116)$$

neste contexto o uso de (4.116) em (4.115) recuperaria uma integração por partes convencional. Todavia se os elementos finitos forem globalmente C^0 , todos os termos internos permanecem.

Substituindo a equação (4.115) em (4.109) vem

$$- \int_0^1 v^\delta \frac{\partial^2}{\partial x^2} u^\delta dx + \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e}^{\Omega_R^e} = \int_0^1 v^\delta f dx + v^\delta(1) g_N, \quad (4.117)$$

reagrupando os termos de (4.117) segue

$$-\int_0^1 v^\delta \left(\frac{\partial^2}{\partial x^2} u^\delta + f \right) dx + \underbrace{\sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e}^{\Omega_R^e}}_{K_1} - v^\delta(1) g_N = 0, \quad (4.118)$$

vamos expandir K_1 , especificamente o primeiro termo na expansão relacionada com a extremidade esquerda e o último termo relacionado com a extremidade direita,

$$\begin{aligned} K_1 &= \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e}^{\Omega_R^e} = \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_R^e} - \sum_{e=1}^{N_{el}} \left[\frac{\partial}{\partial x} u^\delta v^\delta \right]_{\Omega_L^e} \\ &= \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^{N_{el}}} + \sum_{e=1}^{N_{el}-1} \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^e} - \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^1} - \sum_{e=2}^{N_{el}} \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^e}, \end{aligned} \quad (4.119)$$

para ajustar a soma relacionada com a extremidade esquerda, num mesmo intervalo da soma da direita, devemos fazer a seguinte mudança

$$\sum_{e=2}^{N_{el}} \rightarrow \sum_{e=1}^{N_{el}-1}, \quad \Omega_L^e \rightarrow \Omega_L^{e+1}, \quad (4.120)$$

preservando o número de termos e a posição do próximo termo a ser somado, como é de costume. Logo, de (4.120) em (4.119) é possível notar

$$K_1 = \sum_{e=1}^{N_{el}-1} \left[\frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^e} - \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^{e+1}} \right] + \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^{N_{el}}} - \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^1}, \quad (4.121)$$

substituindo a equação (4.121) em (4.118), podemos obter o seguinte arranjo para os termos

$$\begin{aligned} &-\int_0^1 v^\delta \left(\frac{\partial^2}{\partial x^2} u^\delta + f \right) dx - \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^1} \\ &+ \sum_{e=1}^{N_{el}-1} \left[\frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^e} - \frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_L^{e+1}} \right] \\ &+ \left[\frac{\partial}{\partial x} u^\delta v^\delta \Big|_{\Omega_R^{N_{el}}} - v^\delta(1) g_N \right] = 0, \end{aligned} \quad (4.122)$$

de onde é possível notar que o primeiro termo é o produto interno do resíduo pelo peso (v^δ, R) . O segundo termo é nulo pois Ω_L^1 é uma condição de Dirichlet, logo, $v^\delta(\Omega_L^1) = 0$, de acordo com a definição de v . O terceiro termo representa um salto na derivada, com

relação a continuidade da mesma sobre todo o domínio; já no último termo, apresenta-se uma diferença entre a condição de Neumann aproximada e da exata. Fica então claro que se a classe de funções em questão for do tipo C^1 , e se a condição de Neumann for satisfeita, então o resíduo será minimizado, $(v^\delta, R) = 0$.

Vamos construir a aproximação de Galerkin para uma equação diferencial parcial linear, similar ao discutido anteriormente, com uma estruturação mais matemática. Primeiramente vamos considerar a equação de Helmholtz

$$\mathbb{L}(u) \equiv \frac{\partial^2}{\partial x^2} u - \lambda u + f = 0, \quad (4.123)$$

onde λ é uma constante real positiva. A equação é presumida com as seguintes condições de contorno suplementares

$$u(0) = g_D, \quad \frac{\partial u}{\partial x}(l) = g_N, \quad (4.124)$$

Como indicado pelas condições de contorno, vamos tentar determinar a solução no intervalo $0 < x < l$ o qual será denotado por Ω ; o domínio Ω , excluindo-se os pontos finais é denominado aberto,

$$\Omega = \{x | 0 < x < l\} =]0, l[, \quad (4.125)$$

enquanto o domínio $\bar{\Omega}$, incluindo os pontos finais é denominado domínio fechado,

$$\bar{\Omega} = \{x | 0 \leq x \leq l\} = [0, l]. \quad (4.126)$$

multiplicando a equação (4.123) pela função peso $v(x)$, e integrando sobre o domínio Ω , vem

$$\int_0^l v \frac{\partial^2}{\partial x^2} u dx - \int_0^l \lambda v u dx + \int_0^l v f dx = 0, \quad (4.127)$$

integrando por partes o primeiro termo do primeiro membro, e reorganizando vem

$$\int_0^l \frac{\partial v}{\partial x} \frac{\partial u}{\partial x} dx + \int_0^l \lambda v u dx = \int_0^l v f dx + \left[v \frac{\partial u}{\partial x} \right]_0^l, \quad (4.128)$$

vamos agora introduzir a seguinte notação

$$a(v, u) \equiv \int_0^l \left(\frac{\partial v}{\partial x} \frac{\partial u}{\partial x} + \lambda v u \right) dx, \quad (4.129)$$

$$f(v) \equiv \int_0^l v f dx + \left[v \frac{\partial u}{\partial x} \right]_0^l, \quad (4.130)$$

assim a equação (4.127) pode ser escrita como

$$a(v, u) = f(v). \quad (4.131)$$

Em mecânica estrutural, $a(u, u)$ é referida como energia de deformação e o espaço de todas as funções que possuem energia de deformação finita em Ω , é chamado espaço de energia, e denotado por $E(\Omega)$, [14], onde

$$E(\Omega) = \{u | a(u, u) < \infty\}, \quad (4.132)$$

associada com o espaço de energia existe também a norma de energia $\|u\|_E$, definida como segue

$$\|u\|_E = \sqrt{a(u, u)}. \quad (4.133)$$

Funções que pertencem ao espaço de energia são chamadas funções H^1 e satisfazem a condição de que a integral do quadrado da função e de sua derivada são limitados. Vamos considerar soluções do problema representado em (4.123) onde $f(v)$ seja bem comportada, no sentido que $f(v)$ seja finita. Portanto, nós apenas consideramos candidatas a funções de expansão que morem no espaço de energia e satisfaçam as condições de contorno de Dirichlet. Este espaço será chamado espaço de expansão e será denotado por

$$\mathcal{X} = \{u | u \in H^1, u(0) = g_D\}. \quad (4.134)$$

Similarmente vamos definir o espaço de todas as funções peso denotadas por \mathcal{V} , as quais são homogêneas para todos os contornos de Dirichlet,

$$\mathcal{V} = \{v | v \in H^1, v(0) = 0\}, \quad (4.135)$$

o espaço das funções peso \mathcal{V} pode ser denotado por H_0^1 para explicitar que este é um espaço homogêneo. Podemos agora definir a formulação generalizada ou fraca da (4.123)

como: Encontrar $u \in \mathcal{X}$ tal que

$$a(v, u) = f(v) \quad \forall v \in \mathcal{V}. \quad (4.136)$$

O problema considerado, em sua formulação fraca, é ainda um problema de dimensão infinita, devido ao fato dos espaços \mathcal{X} e \mathcal{V} conterem um número infinito de funções. Todavia vamos selecionar subespaços \mathcal{X}^δ ($\mathcal{X}^\delta \subset \mathcal{X}$) e \mathcal{V}^δ ($\mathcal{V}^\delta \subset \mathcal{V}$) que contenham um número finito de funções. No método de elementos espectrais/hp temos dois tipos de discretizações a considerar uma do tipo h (tamanho do elemento) e a outra do tipo p (ordem do polinômio), ambos os casos podem ser considerados na aproximação. Uma aproximação para a solução da forma fraca pode ser estabelecida como segue

$$a(v^\delta, u^\delta) = f(v^\delta) \quad \forall v^\delta \in \mathcal{V}^\delta. \quad (4.137)$$

Na aproximação de Galerkin o mesmo conjunto de funções é utilizado para ambas as funções: de expansão e peso; para isso precisamos tornar a solução homogênea. Podemos notar que $u^\delta \in \mathcal{X}^\delta$ pode ser decomposto em duas outras funções, que obedecem as relações $u^{\mathcal{D}} \in \mathcal{X}^\delta$ e $u^{\mathcal{H}} \in \mathcal{V}^\delta$, ou seja,

$$u^\delta = u^{\mathcal{H}} + u^{\mathcal{D}}, \quad (4.138)$$

onde

$$u^{\mathcal{H}}(0) = 0, \quad u^{\mathcal{D}}(0) = g_{\mathcal{D}}. \quad (4.139)$$

Neste momento poderíamos definir a aproximação de Galerkin, mas primeiramente vamos mencionar que $a(v, u)$ é uma forma bilinear, simétrica, isso significa que

$$a(v, u) = a(u, v), \quad (4.140)$$

$$a(c_1v + c_2w, u) = c_1a(v, u) + c_2a(w, u), \quad (4.141)$$

onde c_1 e c_2 são constantes e u , v e w são funções. Além disso o operador $a(v, u)$ é dito ser contínuo ou limitado se

$$|a(v, u)| \leq C_1 \|v\|_1 \|u\|_1, \quad (4.142)$$

onde $C_1 < \infty$ e o subscrito denota norma em H^1 , é elíptico ou coercivo se

$$a(u, u) \geq C_2 \|v\|_1^2, \quad (4.143)$$

onde $C_2 > 0$. A forma do problema de Galerkin pode ser agora estabelecida como: encontrar

$$u^\delta = u^{\mathcal{H}} + u^{\mathcal{D}}, \quad (4.144)$$

onde

$$u^{\mathcal{H}} \in \mathcal{V}^\delta, \quad (4.145)$$

tal que

$$a(v^\delta, u^{\mathcal{H}}) = f(v^\delta) - a(v^\delta, u^{\mathcal{D}}) \quad \forall v^\delta \in \mathcal{V}^\delta, \quad (4.146)$$

se $a(v^\delta, u^\delta)$ é uma forma elíptica, bilinear, contínua, não necessariamente simétrica e $f(v^\delta)$ é um espaço dual de \mathcal{V}^δ , então o teorema de Lax-Milgram garante ambos: existência e unicidade da solução do problema de Galerkin (4.146). Um outro jeito de construir a solução do problema de Galerkin para esta equação linear é a partir do ponto de vista variacional; a equação (4.123) é o resultado equivalente ao minimizar o seguinte funcional

$$\mathcal{F}(v) = \int_0^l \left[\left(\frac{\partial v}{\partial x} \right)^2 + \lambda(v)^2 - 2vf \right] dx.$$

Todavia ao minimizar $\mathcal{F}(v)$ sobre o espaço de dimensão infinita \mathcal{V} vamos encontrar a solução da equação (4.123) que é a equação de Euler deste funcional, substituindo então o espaço por \mathcal{V}^δ obtemos o método de Ritz-Galerkin.

4.3.1 Propriedades da aproximação de Galerkin

Neste momento vamos introduzir algumas propriedades importantes da aproximação de Galerkin, considerando a aproximação u^δ da solução u onde $u^\delta \in \mathcal{X}^\delta$ e satisfaz, [14],

$$a(v^\delta, u^\delta) = f(v^\delta), \quad \forall v^\delta \in \mathcal{V}^\delta, \quad (4.147)$$

note que (4.147) é equivalente a equação (4.146) desde que $a(v^\delta, u^\delta) = a(v^\delta, u^{\mathcal{H}} + u^{\mathcal{D}})$ e seja usada a bilinearidade de $a(v, u)$ dada pela equação (4.141).

Para mostrar que a solução u^δ é única, vamos assumir que hajam duas soluções distintas u_1 e u_2 ($u_1, u_2 \in \mathcal{X}^\delta$) que satisfazem

$$a(v^\delta, u_1) = f(v^\delta), \quad \forall v^\delta \in \mathcal{V}^\delta, \quad (4.148)$$

e

$$a(v^\delta, u_2) = f(v^\delta), \quad \forall v^\delta \in \mathcal{V}^\delta, \quad (4.149)$$

subtraindo então a equação (4.149) de (4.148) vem

$$a(v^\delta, u_1) - a(v^\delta, u_2) = a(v^\delta, u_1 - u_2) = 0, \quad (4.150)$$

para $u_1 - u_2 \in \mathcal{V}^\delta$ e além disso podemos ajustar $v^\delta = u_1 - u_2$, então a equação (4.149) fica

$$a(u_1 - u_2, u_1 - u_2) = 0, \quad (4.151)$$

todavia isso implica que a norma de energia $\|u_1 - u_2\|_E = 0$, o que torna possível apenas $u_1 = u_2$ se $\lambda \neq 0$, mas se $\lambda = 0$ a solução fica $u_1 - u_2 = C$. A constante C é necessariamente nula se a condição de Dirichlet for especificada, embora a norma $\|u_1 - u_2\|_E$ não possa distinguir entre funções que diferem de uma constante arbitrária quando $\lambda = 0$.

O erro entre as soluções exata e aproximada pode ser definido como $\varepsilon = u - u^\delta$, e é ortogonal a todas as funções do espaço de dimensão finita das funções peso \mathcal{V}^δ , ou seja,

$$a(v^\delta, \varepsilon) = 0, \quad \forall v^\delta \in \mathcal{V}^\delta. \quad (4.152)$$

Para provar esta propriedade, vamos recorrer ao fato da solução exata satisfazer a forma fraca, equação (4.137), ou seja,

$$a(v, u) = f(v), \quad \forall v \in \mathcal{V}, \quad (4.153)$$

e a aproximação satisfaz a equação (4.146). O espaço de dimensão finita das funções peso \mathcal{V}^δ é um sub-espaço do espaço de \mathcal{V} , então a solução exata também satisfaz

$$a(v^\delta, u) = f(v^\delta), \quad \forall v^\delta \in \mathcal{V}^\delta; \quad (4.154)$$

podemos agora subtrair a equação (4.146) da (4.153), e utilizar a bilinearidade

$$\begin{aligned} a(v^\delta, u) &= f(v^\delta) \\ -a(v^\delta, u^\delta) &= -f(v^\delta) \\ \hline a(v^\delta, u - u^\delta) &= 0, \end{aligned}$$

como queríamos demonstrar. também podemos mostrar que a solução de elementos finitos u^δ é uma solução em \mathcal{X}^δ , a qual minimiza o erro na norma de energia

$$\|u - u^\delta\|_E = \min_{w^\delta \in \mathcal{X}^\delta} \|u - w^\delta\|_E, \quad (4.155)$$

onde podemos utilizar a existência do elemento neutro da adição, resultando em

$$\|u - w^\delta\|_E^2 = \|u - u^\delta + u^\delta - w^\delta\|_E^2 = \|\varepsilon + v^\delta\|_E^2. \quad (4.156)$$

Na ocasião definimos $v^\delta = u^\delta - w^\delta$, a seguir vamos utilizar a definição da norma de energia e a propriedade da bilinearidade, então

$$\|u - w^\delta\|_E^2 = a(\varepsilon + v^\delta, \varepsilon + v^\delta) = a(\varepsilon, \varepsilon) + 2a(v^\delta, \varepsilon) + a(v^\delta, v^\delta), \quad (4.157)$$

como vimos o erro é ortogonal a função peso, então para que a norma de energia do erro seja mínima deveríamos ter $v^\delta = 0$, isto implica que $w^\delta = u^\delta$, c.q.d.

Após o estudo da unicidade nas aproximações de Galerkin, fica claro que se há duas expansões independentes, por exemplo: $u_1^\delta(x) = \sum_{i=0}^P \alpha_i \psi_i(x)$, $u_2^\delta(x) = \sum_{i=0}^P \beta_i h_i(x)$, onde estas expansões estão num espaço polinomial de ordem P , então fica claro que

$$u_1^\delta(x) = u_2^\delta(x) \Rightarrow \sum_{i=0}^P \alpha_i \psi_i(x) = \sum_{i=0}^P \beta_i h_i(x). \quad (4.158)$$

Tradicionalmente o método de elementos finitos sempre usa expansões polinomiais e isso pode ser atribuído ao uso histórico da expansão em série de Taylor, a qual permite escrever uma função analítica em termo de polinômios. Funções polinomiais tem a vantagem adicional de possuir regras de integração discreta de fácil implementação computacional.

No método do tipo h um polinômio de grau fixo é usado em cada elemento, e a

convergência é garantida ao se reduzir o tamanho característico de um elemento (h), já no método tipo p uma malha fixa é usada e a convergência é garantida aumentando a ordem do polinômio em cada elemento; se todo o domínio da solução for tratado como um único elemento então o método tipo p se torna um método espectral; o método de elementos finitos do tipo hp combina os atributos de ambos os métodos (tipo h e tipo p).

Vamos demonstrar o processo de decomposição da expansão em contribuições elementares, ou seja, processo de extensões do tipo h . É interessante notar que a decomposição elementar é parte integrante de ambos os métodos tipo h ou tipo p , já que o método do tipo p é baseado numa malha inicial ou discretização do tipo h .

Como será mostrado abaixo, o principal uso da representação elementar é habilitar o tratamento das operações numa base elementar local. Isto não apenas simplifica a implementação mas também permite que muitas operações sejam efetuadas de forma mais eficiente. Para o caso unidimensional, a decomposição pode parecer desnecessária, todavia, os mesmos princípios são aplicados na decomposição em múltiplas dimensões. Ao usar métodos do tipo h , o domínio da solução é subdividido ou particionado em subdomínios ou elementos que não se sobrepõem dentro do qual uma expansão polinomial é usada. Considerando um domínio de solução Ω , nós podemos particioná-lo em uma malha contendo N_{el} elementos, denotados por Ω^e , tal que a união dos elementos que não se sobrepõem se iguala ao domínio original, ou seja,

$$\Omega = \bigcup_{e=1}^{N_{el}} \Omega^e \quad \text{onde} \quad \bigcap_{e=1}^{N_{el}} \Omega^e = \emptyset, \quad (4.159)$$

para o domínio $\Omega = \{x | 0 < x < l\}$ uma malha específica pode ser representada pelos pontos

$$0 = x_0 < x_1 < \dots < x_{N_{el}-1} < x_{N_{el}} = l, \quad (4.160)$$

assim o e -ésimo elemento é definido como

$$\Omega^e = \{x | x_{e-1} < x < x_e\}. \quad (4.161)$$

Como um exemplo podemos considerar o caso mostrado pela figura abaixo⁴:

⁴A figura (20) foi retirada da referência [14].

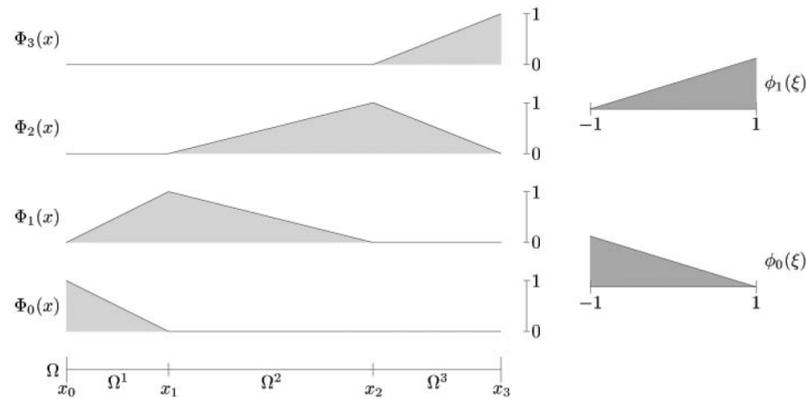


Figura 20: Funções a serem utilizadas na subdivisão em 3 elementos distintos e mapeamento correspondente da coordenada x para coordenada ξ .

Na figura (20) notamos o domínio da solução $\Omega = \{x | 0 < x < l\}$, subdividido em 3 subdomínios distintos. A malha fica definida pelos pontos x_0, x_1, x_2, x_3 onde o número de elementos padrão será sempre o número de subdomínios adicionado de uma unidade; como exemplo definimos o primeiro elemento como

$$\Omega^1 = \{x | x_0 < x < x_1\}, \quad (4.162)$$

Na figura (20), os modos de expansão global para a expansão de elementos finitos linear sobre os $N_{el} = 3$ domínios elementares também estão mostrados. Como é típico em uma expansão linear de elementos finitos, cada modo tem um valor unitário no final de um domínio elementar e decai linearmente a zero enquanto cruza o elemento vizinho. Portanto existem quatro graus de liberdade $N_{dof} = 4$ nesta expansão, que são, $\Phi_0(x), \Phi_1(x), \Phi_2(x)$ e $\Phi_3(x)$. Os modos globais são não-nulos em no máximo dois elementos. Seria portanto muito pouco econômico considerar uma expansão em termos dos modos globais, particularmente ao usar um número muito grande de elementos.

Podemos em nível elementar notar que cada modo global possui apenas duas funções que variam linearmente. Assim se introduzirmos o elemento padrão, Ω_{st} tal que

$$\Omega_{st} = \{\xi | -1 < \xi < 1\},$$

onde podemos definir funções similares que variem linearmente no domínio Ω_{st} em termos

das coordenadas locais ξ como segue

$$\phi_0(\xi) = \begin{cases} \frac{1-\xi}{2} & \xi \in \Omega_{st} \\ 0 & \text{outras regiões} \end{cases} \quad \phi_1(\xi) = \begin{cases} \frac{1+\xi}{2} & \xi \in \Omega_{st} \\ 0 & \text{outras regiões} \end{cases}. \quad (4.163)$$

O elemento padrão Ω_{st} pode ser mapeado para qualquer domínio elementar Ω^e via a transformação $\chi^e(\xi)$ que expressa a coordenada global x em termos da coordenada local ξ como

$$x(\xi) = \chi^e(\xi) = \frac{(1-\xi)}{2}x_{e-1} + \frac{(1+\xi)}{2}x_e, \quad \xi \in \Omega_{st}, \quad (4.164)$$

tal relação pode ser obtida levando em conta que o elemento padrão será um segmento colocado dentro de outro ou seja, por um processo de ajuste de tamanho através da multiplicação por uma constante é possível escrever

$$dx = cte.d\xi, \quad \Delta x = cte.\Delta\xi, \quad (4.165)$$

que equivale a seguinte proposta

$$dx = \frac{\Delta x}{\Delta\xi}d\xi. \quad (4.166)$$

Vamos colocar o elemento padrão inteiro $\Delta\xi = 1 - (-1) = 2$ dentro de um domínio elementar genérico $\Delta x = x_e - x_{e-1}$, substituindo estas considerações em (4.165) obtemos

$$dx = \frac{d\xi}{2}x_e - \frac{d\xi}{2}x_{e-1}; \quad (4.167)$$

como o domínio elementar possui o elemento padrão montado sobre si, as variáveis variam harmoniosamente, de forma que enquanto no domínio elementar ocorre uma variação de x_{e-1} até x , no elemento padrão a variação será de -1 até ξ ; integrando vem

$$\int_{x_{e-1}}^x dx = \int_{-1}^{\xi} \frac{d\xi}{2}x_e - \int_{-1}^{\xi} \frac{d\xi}{2}x_{e-1}, \quad (4.168)$$

$$x - x_{e-1} = \frac{(\xi+1)}{2}x_e - \frac{(\xi+1)}{2}x_{e-1}, \quad (4.169)$$

finalmente, isolando a coordenada x em (4.61) vem

$$x = \frac{(\xi+1)}{2}x_e + \frac{-\xi x_{e-1} - x_{e-1} + 2x_{e-1}}{2}, \quad (4.170)$$

de onde segue finalmente a equação para a coordenada x

$$x(\xi) = \frac{(1-\xi)}{2}x_{e-1} + \frac{(1+\xi)}{2}x_e \equiv \chi^e(\xi), \quad \xi \in \Omega_{st}. \quad (4.171)$$

Conforme foi estabelecida em (4.56), este mapeamento tem uma inversa analítica da forma

$$\xi(x) = (\chi^e)^{-1}(x) = 2 \frac{(x - x_{e-1})}{(x_e - x_{e-1})} - 1, \quad x \in \Omega^e. \quad (4.172)$$

Para obter esta forma podemos partir de (4.167), primeiramente isolando $d\xi$

$$\frac{2}{(x_e - x_{e-1})} dx = d\xi,$$

depois integrando nos intervalos já discutidos anteriormente,

$$\frac{2}{(x_e - x_{e-1})} \int_{x_{e-1}}^x dx = \int_{-1}^{\xi} d\xi, \quad (4.173)$$

$$\frac{2(x - x_{e-1})}{(x_e - x_{e-1})} = \xi + 1, \quad (4.174)$$

e finalmente, ao isolar ξ podemos estabelecer a inversa do mapeamento conforme enunciado em (4.171),

$$\xi(x) \equiv (\chi^e)^{-1} = \frac{2(x - x_{e-1})}{(x_e - x_{e-1})} - 1, \quad x \in \Omega^e. \quad (4.175)$$

Os modos globais $\Phi_i(x)$ podem agora ser representados em termos dos modos de expansão elementar local $\phi_p(\xi)$ pelo mapeamento do elemento padrão Ω_{st} , para cada domínio elementar Ω^e . Isto produzirá os quatro modos globais abaixo:

$$\Phi_0(x) = \begin{cases} \phi_0(\xi(x)) = \frac{1 - \xi(x)}{2} & x \in \Omega^1 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_0([\chi^1]^{-1}(x)) & x \in \Omega^1 \\ 0 & \text{exterior} \end{cases}, \quad (4.176)$$

$$\Phi_1(x) = \begin{cases} \phi_1(\xi(x)) = \frac{1 + \xi(x)}{2} & x \in \Omega^1 \\ \phi_0(\xi(x)) = \frac{1 - \xi(x)}{2} & x \in \Omega^2 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_1([\chi^1]^{-1}(x)) & x \in \Omega^1 \\ \phi_0([\chi^2]^{-1}(x)) & x \in \Omega^2 \\ 0 & \text{exterior} \end{cases}, \quad (4.177)$$

$$\Phi_2(x) = \begin{cases} \phi_1(\xi(x)) = \frac{1 + \xi(x)}{2} & x \in \Omega^2 \\ \phi_0(\xi(x)) = \frac{1 - \xi(x)}{2} & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_1([\chi^2]^{-1}(x)) & x \in \Omega^2 \\ \phi_0([\chi^3]^{-1}(x)) & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases}, \quad (4.178)$$

$$\Phi_3(x) = \begin{cases} \phi_1(\xi(x)) = \frac{1 + \xi(x)}{2} & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_0([\chi^3]^{-1}(x)) & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases}. \quad (4.179)$$

Fica claro então que o uso do mapeamento inverso (4.175) nas equações (4.176)-(4.179), produz a representação dos modos globais com a variável x explícita, para ilustrar tal fato, vamos substituir o mapeamento inverso em ambas as funções locais ϕ_0 e ϕ_1 , isto permite notar, que a consequência do uso de (4.175) implica no fato do elemento padrão ter sempre as seguintes equações:

retas decrescentes

$$\phi_0(x) = \left[\frac{(x - x_e)}{(x_{e-1} - x_e)} \right]_{e=i} \text{ no domínio elementar } \Omega^i, \quad (4.180)$$

retas crescentes

$$\phi_1(x) = \left[\frac{(x - x_{e-1})}{(x_e - x_{e-1})} \right]_{e=i} \text{ no domínio elementar } \Omega^i. \quad (4.181)$$

As equações (4.180) e (4.181) podem então ser utilizadas para escrever as funções “chapéu” dos quatro modos globais conforme abaixo

$$\Phi_0(x) = \begin{cases} \frac{(x - x_1)}{(x_0 - x_1)} & x \in \Omega^1 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_0([\chi^1]^{-1}(x)) & x \in \Omega^1 \\ 0 & \text{exterior} \end{cases}, \quad (4.182)$$

$$\Phi_1(x) = \begin{cases} \frac{(x - x_0)}{(x_1 - x_0)} & x \in \Omega^1 \\ \frac{(x - x_2)}{(x_1 - x_2)} & x \in \Omega^2 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_1([\chi^1]^{-1}(x)) & x \in \Omega^1 \\ \phi_0([\chi^2]^{-1}(x)) & x \in \Omega^2 \\ 0 & \text{exterior} \end{cases}, \quad (4.183)$$

$$\Phi_2(x) = \begin{cases} \frac{(x - x_1)}{(x_2 - x_1)} & x \in \Omega^2 \\ \frac{(x - x_3)}{(x_2 - x_3)} & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_1([\chi^2]^{-1}(x)) & x \in \Omega^2 \\ \phi_0([\chi^3]^{-1}(x)) & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases}, \quad (4.184)$$

$$\Phi_3(x) = \begin{cases} \frac{(x-x_2)}{(x_3-x_2)} & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases} = \begin{cases} \phi_0([\chi^3]^{-1}(x)) & x \in \Omega^3 \\ 0 & \text{exterior} \end{cases}. \quad (4.185)$$

4.3.2 Exemplo de Elementos Finitos: Dois Subdomínios em 1D

Vamos agora resolver a equação de Poisson em uma dimensão,

$$\mathbb{L}(u) \equiv \frac{\partial^2}{\partial x^2} u + f = 0, \quad (4.186)$$

onde $f(x)$ é uma função conhecida e as condições de contorno são:

$$u(0) = g_{\mathcal{D}} = 1, \quad \frac{\partial u}{\partial x}(1) = g_{\mathcal{N}} = 1. \quad (4.187)$$

Vamos considerar a forma fraca, lembrando a equação (4.112),

$$\int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^{\mathcal{H}} dx = \int_0^1 v^\delta f dx + v^\delta(1) g_{\mathcal{N}} - \int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^{\mathcal{D}} dx. \quad (4.188)$$

A solução será aproximada por funções seccionalmente lineares sob dois sub-domínios Ω^1 e Ω^2 . Este tipo de aproximação é conhecida como aproximação do tipo h, onde o parâmetro h representa o tamanho característico de um sub-domínio (para 1-D h é o comprimento). A convergência da solução aproximada para a solução exata ocorre quando o domínio Ω for sub-dividido em subdomínios cada vez menores, ou seja, fazendo $h \rightarrow 0$. Para o caso de dois sub-domínios a expansão aproximada toma a forma

$$u^\delta = \sum_{i=0}^2 u_i \Phi_i(x). \quad (4.189)$$

Para obter as funções $\Phi(x)$ basta utilizar o padrão de distribuição de funções chapéu (par de retas: uma crescente e outra decrescente) sobre cada interface, conforme a ilustração abaixo:

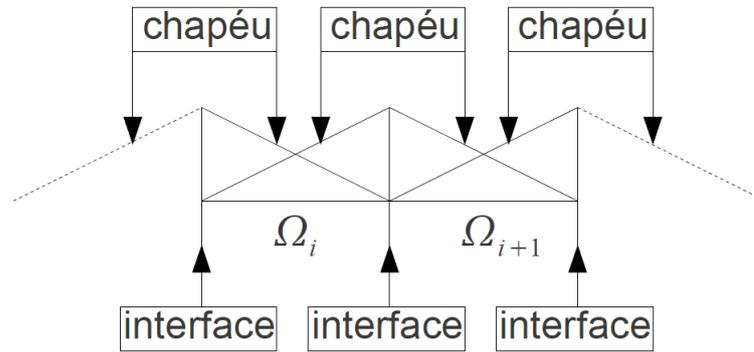


Figura 21: Par de funções uma crescente e outra decrescente inseridas em cada interface.

Vale notar que na extremidade a região externa ao domínio é descartada, ou seja, a reta pontilhada na ilustração da figura (21) não é levada em conta. Assim sendo cada função está associada com um ponto no domínio.

Na ocasião estamos dividindo o domínio em duas sub-regiões Ω^1 e Ω^2 , o que traz três pontos x_0 , x_1 e x_2 , permitindo obter as funções Φ_0 , Φ_1 e Φ_2 como o ponto x_0 é uma região da fronteira esquerda repousa sobre ele uma função decrescente, de acordo com a figura acima; é padrão obter as equações de reta num sistema cartesiano 2-D seguindo os dois pontos $P_i(x_i, y_i)$ e $P_f(x_f, y_f)$ já que por dois pontos passam uma única reta, o coeficiente angular será dado por $m = \frac{y_f - y_i}{x_f - x_i}$ e a equação da reta será do tipo $y = m(x - x_i) + y_i$, logo podemos escrever a reta para Φ_0 pois esta terá os pontos $\{x_i, y_i, x_f, y_f\} = \{x_0, 1, x_1, 0\}$, $y = -\frac{1}{(x_1 - x_0)}(x - x_0) + 1$, definindo $\Omega = \{x | 0 \leq x \leq 1\}$ vem $y = -\frac{1}{(2^{-1} - 0)}(x - 0) + 1$, então $y = 1 - 2x$ para a região de $0 \leq x \leq \frac{1}{2}$ e 0 para $\frac{1}{2} \leq x \leq 1$, com este mesmo processo onde são utilizadas equações de reta podem ser deduzidas as funções Φ_0 , Φ_1 e Φ_2 , conforme abaixo:

$$\begin{array}{ccc} \Phi_0(x) & \Phi_1(x) & \Phi_2(x) \\ \left\{ \begin{array}{ll} 1-2x & 0 \leq x \leq \frac{1}{2} \\ 0 & \frac{1}{2} \leq x \leq 1 \end{array} \right. & \left\{ \begin{array}{ll} 2x & 0 \leq x \leq \frac{1}{2} \\ 2(1-x) & \frac{1}{2} \leq x \leq 1 \end{array} \right. & \left\{ \begin{array}{ll} 0 & 0 \leq x \leq \frac{1}{2} \\ 2x-1 & \frac{1}{2} \leq x \leq 1 \end{array} \right. \end{array}$$

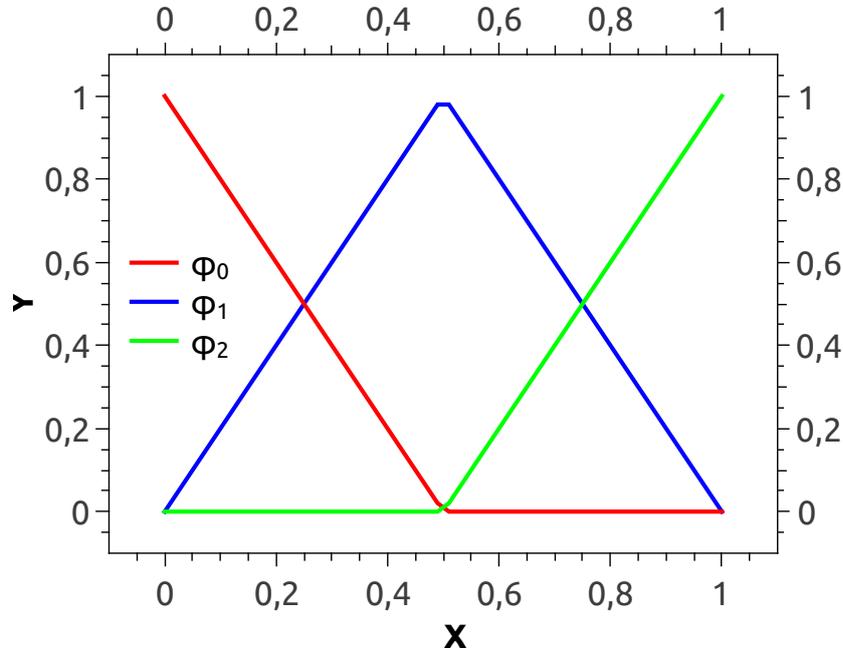


Figura 22: Funções tipo “chapéu” a serem introduzidas nos subdomínios.

Das funções podemos notar a seguinte expansão:

$$u^\delta(x) = u_0\Phi_0 + u_1\Phi_1 + u_2\Phi_2, \quad (4.190)$$

cuja avaliação da condição de Dirichlet ($u^\delta(0) = g_{\mathcal{D}}$) nos remete ao seguinte estudo

$$u^\delta(0) = u_0(1-2x)|_{x=0} + u_1(2x)|_{x=0} + u_2(0)|_{x=0} = u_0, \quad (4.191)$$

não restando outra opção a não ser ajustar $u_0 = g_{\mathcal{D}}$, de (4.191) e (4.190) notamos que é conveniente denominar os termos como segue

$$u^\delta(x) = \underbrace{g_{\mathcal{D}}\Phi_0}_{u^{\mathcal{D}}} + \underbrace{u_1\Phi_1 + u_2\Phi_2}_{u^{\mathcal{H}}},$$

pois $u^{\mathcal{D}}$ é a parte estudada em (4.190) que garante a condição de Dirichlet, enquanto $u^{\mathcal{H}}$ é a parte homogênea (que se anula em $x = 0$).

Tanto as funções peso quanto a função $f(x)$ também podem ser aproximadas com uso das funções de expansão, ou seja,

$$v^\delta(x) = v_1\Phi_1(x) + v_2\Phi_2(x), \quad (4.192)$$

enquanto

$$f(x) = \sum_{i=0}^2 f_i\Phi_i(x) = f_0\Phi_0(x) + f_1\Phi_1(x) + f_2\Phi_2(x). \quad (4.193)$$

Lembrando da equação (4.188),

$$\int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^{\mathcal{H}} dx = \int_0^1 v^\delta f dx + v^\delta(1) g_{\mathcal{N}} - \int_0^1 \frac{\partial}{\partial x} v^\delta \frac{\partial}{\partial x} u^{\mathcal{D}} dx, \quad (4.194)$$

cujos termos podem ser calculados como segue:

$$\int_0^1 \frac{\partial v^\delta}{\partial x} \frac{\partial u^{\mathcal{H}}}{\partial x} dx = \int_0^{\frac{1}{2}} 2v_1 2u_1 dx + \int_{\frac{1}{2}}^1 (-2v_1 + 2v_2)(-2u_1 + 2u_2) dx, \quad (4.195)$$

$$\int_0^1 \frac{\partial v^\delta}{\partial x} \frac{\partial u^{\mathcal{H}}}{\partial x} dx = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad (4.196)$$

$$\begin{aligned} \int_0^1 v^\delta f dx &= \int_0^{\frac{1}{2}} (v_1 2x) [f_0(1-2x) + f_1(2x)] dx \\ &+ \int_{\frac{1}{2}}^1 [v_1 2(1-x) + v_2(2x-1)] [f_1 2(1-x) + f_2(2x-1)] dx \\ &= \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} \frac{1}{12}f_0 + \frac{1}{3}f_1 + \frac{1}{12}f_2 \\ \frac{1}{12}f_1 + \frac{1}{6}f_2 \end{bmatrix}, \end{aligned} \quad (4.197)$$

$$v^\delta(1) g_{\mathcal{N}} = [v_1\Phi_1(1) + v_2\Phi_2(1)] g_{\mathcal{N}} = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} g_{\mathcal{N}}, \quad (4.198)$$

$$\int_0^1 \frac{\partial v^\delta}{\partial x} \frac{\partial u^{\mathcal{D}}}{\partial x} dx = \int_0^{\frac{1}{2}} (2v_1)(-2g_{\mathcal{D}}) dx = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} -2g_{\mathcal{D}} \\ 0 \end{bmatrix}. \quad (4.199)$$

Logo, ao substituirmos os termos calculados nas equações (4.196)-(4.199) obtemos a seguinte equação matricial

$$\begin{bmatrix} v_1 & v_2 \end{bmatrix} \left\{ \begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - \begin{bmatrix} \frac{1}{12}f_0 + \frac{1}{3}f_1 + \frac{1}{12}f_2 \\ \frac{1}{12}f_1 + \frac{1}{6}f_2 \end{bmatrix} - \begin{bmatrix} 0 \\ g_N \end{bmatrix} + \begin{bmatrix} -2g_D \\ 0 \end{bmatrix} \right\} = 0, \quad (4.200)$$

onde podemos utilizar a arbitrariedade de v_1 e v_2 para inferir a nulidade do termo entre chaves em (4.200), de onde segue,

$$\begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 2 + \frac{1}{12}f_0 + \frac{1}{3}f_1 + \frac{1}{12}f_2 \\ 1 + \frac{1}{12}f_1 + \frac{1}{6}f_2 \end{bmatrix}. \quad (4.201)$$

Em (4.201) foram utilizadas as condições de contorno; finalmente, basta multiplicar a equação (4.201) pela matriz inversa de M , onde

$$M \equiv \begin{bmatrix} 4 & -2 \\ -2 & 2 \end{bmatrix}, \quad (4.202)$$

feito isso, obtemos finalmente o valor dos coeficientes u_1 e u_2 ,

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \frac{3}{2} + \frac{1}{24}f_0 + \frac{5}{24}f_1 + \frac{1}{8}f_2 \\ 2 + \frac{1}{24}f_0 + \frac{1}{4}f_1 + \frac{5}{24}f_2 \end{bmatrix}, \quad (4.203)$$

a aproximação de elementos finitos $u^\delta(x) = g_D\Phi_0(x) + u_1\Phi_1 + u_2\Phi_2$, será então dada por

$$u^\delta = \begin{cases} 1 + x + \frac{x}{12}f_0 + \frac{5x}{12}f_1 + \frac{x}{4}f_2 & 0 \leq x \leq \frac{1}{2} \\ 1 + x + \frac{1}{24}f_0 + \frac{2+x}{12}f_1 + \frac{1+4x}{24}f_2 & \frac{1}{2} \leq x \leq 1 \end{cases}. \quad (4.204)$$

Finalmente podemos confirmar que a equação residual (4.122) é nula. A expansão linear $u^\delta(x)$ implica que $\frac{\partial^2 u^\delta}{\partial x^2}(x)$ é zero e $\frac{\partial u^\delta}{\partial x}(x)$ é uma constante, então da equação (4.122) para dois sub-domínios é possível notar a seguinte relação

$$-\int_0^1 v^\delta \left(\frac{\partial^2 u^\delta}{\partial x^2} + f \right) dx + \left[v^\delta \frac{\partial u^\delta}{\partial x} \right]_{\Omega_L^1}^{\Omega_R^1} + \left[v^\delta \frac{\partial u^\delta}{\partial x} \right]_{\Omega_L^2}^{\Omega_R^2} - v^\delta(1) g_N \equiv R_e, \quad (4.205)$$

onde o parâmetro R_e é um resíduo a ser calculado, assim para saber o valor de R_e devemos calcular os termos do primeiro membro (esquerda) em (4.205); seguem então as relações abaixo

$$\frac{\partial u^\delta}{\partial x} = \begin{cases} 1 + \frac{1}{12}f_0 + \frac{5}{12}f_1 + \frac{1}{4}f_2 & 0 \leq x \leq \frac{1}{2} \\ 1 + \frac{1}{12}f_1 + \frac{1}{6}f_2 & \frac{1}{2} \leq x \leq 1 \end{cases}, \quad (4.206)$$

$$\frac{\partial^2}{\partial x^2} u^\delta = \begin{cases} 0, & 0 \leq x \leq \frac{1}{2} \\ 0, & \frac{1}{2} \leq x \leq 1 \end{cases}, \quad (4.207)$$

$$v^\delta = \begin{cases} v_1(2x) + v_2(0), & \text{para } 0 \leq x \leq \frac{1}{2} \\ v_1(2-2x) + v_2(2x-1), & \text{para } \frac{1}{2} \leq x \leq 1 \end{cases}, \quad (4.208)$$

$$\Omega_L^1 = 0, \quad \Omega_R^1 = \frac{1}{2}, \quad \Omega_L^2 = \frac{1}{2}, \quad \Omega_R^2 = 1, \quad (4.209)$$

com estes termos em mão o parâmetro R_e ficará

$$R_e = - \int_0^1 v^\delta f dx + \left[v_1 \left(\frac{f_0}{12} + \frac{f_1}{3} + \frac{f_2}{12} \right) + v_2 \left(\frac{f_1}{12} + \frac{f_2}{6} \right) \right], \quad (4.210)$$

de onde notamos facilmente que R_e é nulo, pois a integral $-\int_0^1 v^\delta f dx$ foi calculada em (4.197).

4.4 Método de Galerkin Descontínuo com Pênalti

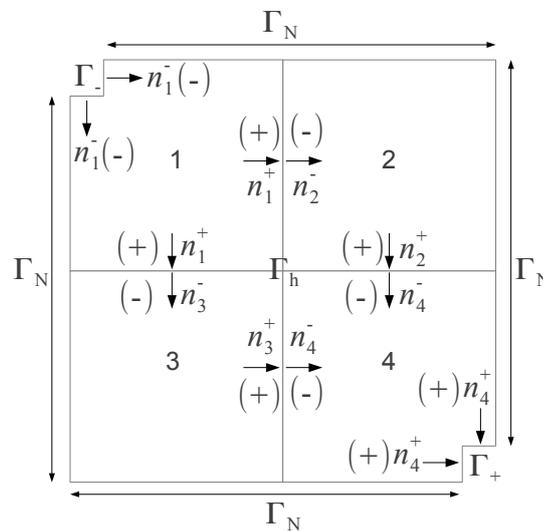


Figura 23: Diagrama ilustrando o fluxo das funções entre os elementos, o uso dos vetores orientados serve apenas para não causar equívoco entre as muitas faces internas. Podemos notar que os vetores que apontam para fora do volume de um elemento foram adotados como vetores unitários de valor +1, enquanto os vetores que apontam para dentro do volume foram adotados como vetores unitários de valor -1.

O uso de operadores diferenciais permite a representação de uma equação diferencial como uma equação de operadores, por exemplo, é possível trocar a equação

$$\nabla^2 u + f = 0, \quad (4.211)$$

pela representação

$$\hat{L}u = 0, \quad (4.212)$$

A representação através do método de operadores é especialmente útil no método de resíduos ponderados, através do qual uma dada função de interesse u , que sofre a atuação do operador \hat{L} , pode ser expandida e representada por uma função aproximada,

$$\begin{aligned} u_{aprox.} &= \underbrace{u_0}_{\text{c.i.}} + \sum_{i=1}^N u_i \phi_i = u_0 + u_1 \phi_1 + u_2 \phi_2 + \dots, \\ u &\approx u_{aprox.}, \end{aligned} \quad (4.213)$$

Se o operador \hat{L} quando atua na função u produz valor resultante nulo ($\hat{L}u = 0$), então, é de se esperar que o uso da aproximação ($u \approx u_{aprox.}$) produza um valor residual não nulo, denominado resíduo,

$$\hat{L}u_{aprox.} \equiv R, \quad (4.214)$$

quanto mais o valor de $u_{aprox.}$ se aproxima do valor de u , mais próximo de zero fica o valor do resíduo R .

Uma outra função denominada função peso v pode ser utilizada para realizar produto interno com o resíduo (R, v) , de modo que

$$(R, v) = \int_{\Omega} Rv = \int_{\Omega} \hat{L}u_{aprox.}v. \quad (4.215)$$

É possível impor que o produto interno entre R e v seja nulo, este tipo de restrição permite definir o método de resíduos ponderados, através do qual podem ser construídos diferentes métodos numéricos usando diferentes valores de v : colocação, volumes finitos, mínimos quadrados, Galerkin, Petrov-Galerkin...

O método de volumes finitos é obtido quando $v_j = 1$, no interior do sub-domínio Ω_j e $v_j = 0$ no exterior deste subdomínio, o método de Galerkin é obtido quando as funções de v_j são escolhidas como sendo as funções da base usadas para expandir u , ou seja, as

funções peso são as funções de expansão $v_j = \phi_j$.

$$(R, v) = \int_{\Omega} Rv, \quad (4.216)$$

O domínio Ω é subdividido em uma partição $\varepsilon_h = \{E\}_E$, consistindo de N_h elementos, quadriláteros ou triângulos, de diâmetro máximo h . Seja Γ_h a união dos conjuntos abertos que coincidem com o interior dos elementos da partição ε_h , onde e simboliza um segmento de Γ_h compartilhado entre dois elementos E^k e E^l de ε_h , associado com e , há um vetor unitário normal n_e , direcionado de E^k para E^l , [9],

$$\Gamma_h = \{E_1^k|_e E_2^l, E_2^k|_e E_3^l, E_3^k|_e E_4^l, \dots, E_{i-1}^k|_e E_i^l\}, \quad (4.217)$$

a fronteira do domínio, é subdividida em Γ_N , Γ_- e Γ_+ , de modo que

$$\partial\Omega = \Gamma_N \cup \Gamma_+ \cup \Gamma_-, \quad (4.218)$$

onde Γ_N é o contorno onde não há fluxo, Γ_+ é o contorno onde o fluxo é de saída e Γ_- é o contorno onde o fluxo é de entrada.

Definimos o salto da função ψ como segue,

$$[\psi] = (\psi|_{E^k})|_e - (\psi|_{E^l})|_e, \quad (4.219)$$

analogamente é possível definir a média da função ψ ,

$$\{\psi\} = \frac{1}{2} (\psi|_{E^k})|_e + \frac{1}{2} (\psi|_{E^l})|_e. \quad (4.220)$$

A equação da pressão para o escoamento bifásico, incluindo os termos de fonte, é dada por,

$$-\nabla \cdot (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) = (q_w + q_n); \quad (4.221)$$

seja a função vetorial \vec{f} e a função escalar q definidas como segue,

$$\begin{aligned} \vec{f} &\equiv \lambda_t k \nabla p_w + \lambda_n k \nabla p_c, \\ q &\equiv q_w + q_n, \end{aligned} \quad (4.222)$$

então a equação da pressão pode ser reescrita em termos de \vec{f} e q , usando (4.222) em (4.221),

$$\nabla \cdot \vec{f} + q = 0. \quad (4.223)$$

Um operador \hat{L} pode ser usado para representar a equação diferencial (4.223),

$$\hat{L}\vec{f} \equiv \nabla \cdot \vec{f} + q, \quad (4.224)$$

logo,

$$\hat{L}\vec{f} = 0; \quad (4.225)$$

as funções de interesse de \vec{f} são expandidas em termos de uma base, isso gera uma aproximação, como consequência, $\vec{f} \approx \vec{f}_{approx.}$, ao substituir esta aproximação na equação acima, um resíduo aparecerá,

$$\hat{L}\vec{f}_{approx.} = R. \quad (4.226)$$

O próximo passo consiste em efetuar o produto interno do resíduo por uma função peso, z

$$(R, z) = \int_{\Omega} Rz = \int_{\Omega} \left(\hat{L}\vec{f}_{approx.} \right) z, \quad (4.227)$$

ao impor que o produto interno entre R e z é nulo, condição para construir métodos numéricos via resíduos ponderados, obtemos

$$\begin{aligned} (R, z) &= 0, \\ \int_{\Omega} \left(\hat{L}\vec{f}_{approx.} \right) z &= 0. \end{aligned} \quad (4.228)$$

Especificamente, para o método de Galerkin, há a necessidade de que a função peso z seja escrita na mesma base usada para expandir a função \vec{f} . Recordando da equação representada por \hat{L} operador é possível escrever,

$$\int_{\Omega} z \left(\hat{L}\vec{f}_{approx.} \right) = \int_{\Omega} \left(\nabla \cdot \vec{f}_{approx.} + q \right) z = 0, \quad (4.229)$$

ou seja,

$$- \int_{\Omega} z \nabla \cdot \vec{f}_{approx.} = \int_{\Omega} qz; \quad (4.230)$$

por comodidade, vamos denotar a função aproximada $\vec{f}_{approx.}$ apenas por \vec{f} , ou seja,

$$- \int_{\Omega} z \nabla \cdot \vec{f} = \int_{\Omega} qz. \quad (4.231)$$

Substituindo \vec{f} em (4.231) vem

$$- \int_{\Omega} z \nabla \cdot (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) = \int_{\Omega} (q_w + q_n) z. \quad (4.232)$$

Em (4.232), a identidade vetorial seguinte é útil para a integral dos termos do lado esquerdo da igualdade, pois torna possível a aplicação do teorema da divergência de Gauss,

$$\nabla \cdot (\vec{f}z) = z\nabla \cdot \vec{f} + \nabla z \cdot \vec{f}, \quad (4.233)$$

de forma equivalente,

$$z\nabla \cdot \vec{f} = \nabla \cdot (\vec{f}z) - \nabla z \cdot \vec{f}, \quad (4.234)$$

substituindo a identidade acima na integral em (4.232), obtemos

$$\int_{\Omega} -(\nabla \cdot (\vec{f}z) - \nabla z \cdot \vec{f}) = \int_{\Omega} (q_w + q_n) z. \quad (4.235)$$

Na equação (4.235) surge a possibilidade de separar as integrais, para aplicar na integral que possui o termo $\nabla \cdot (\vec{f}z)$ o teorema da divergência de Gauss (TDG),

$$\int_{\Omega} \nabla z \cdot \vec{f} - \underbrace{\int_{\Omega} \nabla \cdot (\vec{f}z)}_{\text{(TDG)}} = \int_{\Omega} (q_w + q_n) z, \quad (4.236)$$

é possível trocar a integral de volume do domínio Ω pela soma das integrais dos volumes dos elementos E que pertencem à partição ε_h , já a integral que foi modificada pelo teorema da divergência de Gauss, ao invés de ser trocada pela soma dos volumes dos elementos E é trocada pela soma das faces dos volumes E , ou seja, e pertencente a $\Gamma_h U \partial \Omega$, pois o teorema de Gauss relaciona a divergência de uma função vetorial com relação a um volume com o fluxo desta função vetorial através da superfície que enclausura o referido volume; deste modo a integral acima quando escrita em termos dos elementos fica:

$$\sum_{E \in \varepsilon_h} \int_E \nabla z \cdot \vec{f} - \sum_{e \in \Gamma_h \cup \partial \Omega} \int_e [\vec{f}z \cdot \hat{n}_e] = \int_{\Omega} (q_w + q_n) z. \quad (4.237)$$

É interessante notar que o fluxo através das faces dos elementos está relacionado com o salto da função na face entre os elementos distintos, o termo que aparece na integral acima, $[\vec{f}z \cdot \hat{n}_e]$, é o salto da função $\vec{f}z \cdot \hat{n}_e$; substituindo de volta a função vetorial \vec{f} na integral acima obtemos

$$\begin{aligned} \int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) \cdot \nabla z + \\ &- \sum_{e \in \Gamma_h \cup \partial \Omega} \int_e [(\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) z \cdot \hat{n}_e]. \end{aligned} \quad (4.238)$$

para estudar o salto de um produto, por exemplo $[\vec{f}z \cdot \hat{n}_e]$, onde aparece em produto no interior do salto a função z e a função $\vec{f} \cdot \hat{n}_e$, é útil investigar a identidade que relaciona o salto de um produto, por exemplo $[ab]$ em termos das quantidades $\{a\}$, $\{b\}$, $[a]$, $[b]$, que são: a média de a , média b , salto de a e salto de b , respectivamente; primeiramente define-se o salto de uma quantidade arbitrária x conforme segue,

$$[x] = \frac{1}{2} (x^- - x^+), \quad (4.239)$$

bem como a média de uma quantidade arbitrária x ,

$$\{x\} = \frac{1}{2} (x^- + x^+), \quad (4.240)$$

agora podemos arranjar em produto a média de a e o salto de b ,

$$\{a\} [b] = \frac{1}{2} a^- b^- - \frac{1}{2} a^- b^+ + \frac{1}{2} a^+ b^- - \frac{1}{2} a^+ b^+, \quad (4.241)$$

em seguida, o produto da média de b pelo salto de a ,

$$[a] \{b\} = \frac{1}{2} a^- b^- + \frac{1}{2} a^- b^+ - \frac{1}{2} a^+ b^- - \frac{1}{2} a^+ b^+; \quad (4.242)$$

somando as duas equações acima haverá cancelamento de termos simétricos, resultando apenas

$$\{a\} [b] + [a] \{b\} = a^- b^- - a^+ b^+. \quad (4.243)$$

Mas $a^- b^- - a^+ b^+$, é equivalente a $[ab]$, portanto é possível escrever a seguinte identidade,

$$[ab] = \{a\} [b] + [a] \{b\}. \quad (4.244)$$

Usando a identidade de (4.244) em (4.238) a seguinte equação é obtida,

$$\begin{aligned} \int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) \cdot \nabla z \\ &- \sum_{e \in \Gamma_h \cup \partial\Omega} \int_e \hat{n}_e \cdot \{ \lambda_t k \nabla p_w + \lambda_n k \nabla p_c \} [z] + \\ &- \sum_{e \in \Gamma_h \cup \partial\Omega} \int_e \{z\} [\lambda_t k \nabla p_w + \lambda_n k \nabla p_c] \cdot \hat{n}_e, \end{aligned} \quad (4.245)$$

É interessante notar para o terceiro termo do segundo membro de (4.245) a seguinte identidade,

$$\begin{aligned} \nabla (\{z\} [\lambda_t k p_w + \lambda_n k p_c]) \cdot \hat{n}_e &= \{\nabla z\} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e + \\ &+ \{z\} [\lambda_t k \nabla p_w + \lambda_n k \nabla p_c] \cdot \hat{n}_e. \end{aligned} \quad (4.246)$$

Ao impor que as funções p_w e p_c sejam contínuas em (4.246), o termo de salto $[\lambda_t k p_w + \lambda_n k p_c]$ deve se anular, restando apenas

$$0 = \{\nabla z\} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e + \{z\} [\lambda_t k \nabla p_w + \lambda_n k \nabla p_c] \cdot \hat{n}_e, \quad (4.247)$$

ou simplesmente,

$$\{z\} [\lambda_t k \nabla p_w + \lambda_n k \nabla p_c] \cdot \hat{n}_e = - \{\nabla z\} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e; \quad (4.248)$$

substituindo a equação (4.248) em (4.245) ficamos com,

$$\begin{aligned} \int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) \cdot \nabla z + \\ &- \sum_{e \in \Gamma_h \cup \partial\Omega} \int_e \hat{n}_e \cdot \{\lambda_t k \nabla p_w + \lambda_n k \nabla p_c\} [z] + \\ &+ \sum_{e \in \Gamma_h \cup \partial\Omega} \int_e \{\nabla z\} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e. \end{aligned} \quad (4.249)$$

As condições de contorno nas fronteiras estão associadas com o termo que possui a integral $\int_e \{\nabla z\} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e$ de (4.249). Para aplicar as condições de contorno na fronteira do domínio Ω basta notar as seguintes propriedades do salto, que serão demonstradas mais adiante,

$$[f] = f_{int.} - f_{ext.}, \quad (4.250)$$

para funções vetoriais a propriedade é semelhante,

$$[\vec{f}] = (\vec{f}_{int.} - \vec{f}_{ext.}) \cdot \hat{n}_e, \quad (4.251)$$

onde o índice *int.* se refere ao elemento interno, enquanto o índice *ext.* se refere ao elemento externo, substituindo (4.250) em (4.249) obtém-se

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \hat{n}_e \cdot \{ \lambda_t k \nabla p_w + \lambda_n k \nabla p_c \} [z] + \\
&+ \sum_{e \in \Gamma_h} \int_e \{ \nabla z \} [\lambda_t k p_w + \lambda_n k p_c] \cdot \hat{n}_e + \\
&+ \sum_{e \in \Gamma_+} \int_e \nabla z \cdot \hat{n}_e (\lambda_t k p_{int.}^+ + \lambda_n k p_c (s_{int.}^+)) + \\
&- \sum_{e \in \Gamma_+} \int_e \{ \nabla z \cdot \hat{n}_e \} (\lambda_t k p_{dir.}^+ + \lambda_n k p_c (s_{dir.}^+)) + \\
&+ \sum_{e \in \Gamma_-} \int_e \{ \nabla z \cdot \hat{n}_e \} (\lambda_t k p_{int.}^- + \lambda_n k p_c (s_{int.}^-)) + \\
&- \sum_{e \in \Gamma_-} \int_e \{ \nabla z \cdot \hat{n}_e \} (\lambda_t k p_{dir.}^- + \lambda_n k p_c (s_{dir.}^-)). \tag{4.252}
\end{aligned}$$

As condições de contorno do modelo de escoamento bifásico em questão são as seguintes, [9],

$$\begin{aligned}
p_w &= p_{dir.}^-, \quad s_n = s_{dir.}, \quad \text{em } \Gamma_- \text{- contorno onde há fluxo de entrada,} \\
p_w &= p_{dir.}^+, \quad \lambda_n K \nabla p_c \cdot n = 0, \quad \text{em } \Gamma_+ \text{- contorno onde há fluxo de saída,} \\
\lambda_w K \nabla p_w \cdot n &= 0, \quad \lambda_n K \nabla p_n \cdot n = 0, \quad \text{em } \Gamma_N \text{- contorno onde não há fluxo.} \tag{4.253}
\end{aligned}$$

Aplicando as condições de (4.253) em (4.252), desmembrando os termos das integrais e ainda nomeando os termos: $p_{int.}^+ = [p_w]$ em Γ_+ , $p_{int.}^- = [p_w]$ em Γ_- , $p_c (s_{int.}^-) = [p_c]$ em

Γ_- , $p_{dir}^- = p_{dir}$ em Γ_- e $p_{dir}^+ = p_{dir}$ em Γ_+

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t k \nabla p_w \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n k \nabla p_c \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_t k \nabla p_w\} [z] - \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_n k \nabla p_c\} [z] + \\
&+ \sum_{e \in \Gamma_h} \int_e \{\nabla z\} [\lambda_t k p_w] \cdot \hat{n}_e + \sum_{e \in \Gamma_h} \int_e \{\nabla z\} [\lambda_n k p_c] \cdot \hat{n}_e + \\
&+ \sum_{e \in \Gamma_+} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} \underbrace{p_{int.}^+}_{[p_w]} + \sum_{e \in \Gamma_+} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} \underbrace{p_c(s_{int.}^+)}_0 + \\
&- \sum_{e \in \Gamma_+} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} \underbrace{p_{dir.}^+}_{p_{dir}} - \sum_{e \in \Gamma_+} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} \underbrace{p_c(s_{dir.}^+)}_0 + \\
&+ \sum_{e \in \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} \underbrace{p_{int.}^-}_{[p_w]} + \sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} \underbrace{p_c(s_{int.}^-)}_{[p_c]} + \\
&- \sum_{e \in \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} \underbrace{p_{dir.}^-}_{p_{dir}} - \sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} \underbrace{p_c(s_{dir.}^-)}_{p_c(s_{dir.})}, \quad (4.254)
\end{aligned}$$

realizando as alterações marcadas na equação acima obtemos

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t k \nabla p_w \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n k \nabla p_c \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_t k \nabla p_w\} [z] - \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_n k \nabla p_c\} [z] + \\
&+ \underbrace{\sum_{e \in \Gamma_h} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} [p_w]}_1 + \underbrace{\sum_{e \in \Gamma_h} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} [p_c]}_2 + \\
&+ \underbrace{\sum_{e \in \Gamma_+} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} [p_w]}_1 + 0 - \underbrace{\sum_{e \in \Gamma_+} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} p_{dir.}}_3 + \\
&- 0 + \underbrace{\sum_{e \in \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} [p_w]}_1 + \underbrace{\sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} [p_c]}_2 + \\
&- \underbrace{\sum_{e \in \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} p_{dir.}}_3 - \sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} p_c(s_{dir.}). \quad (4.255)
\end{aligned}$$

Reunindo na equação acima os termos sublinhados com mesmo valor: 1, 2 e 3, obtemos a seguinte equação

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t k \nabla p_w \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n k \nabla p_c \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_t k \nabla p_w\} [z] - \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_n k \nabla p_c\} [z] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} [p_w] + \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} [p_c] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} p_{dir.} - \sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} p_c (s_{dir.}) \quad (4.256)
\end{aligned}$$

Dois tipos de fatores de pênalti podem ser introduzidos, um deles refere-se aos saltos da função que é solução da equação diferencial, enquanto o outro se refere à derivada desta função; sejam duas funções genéricas $f(x)$ e $g(x)$, os fatores de pênalti das funções, J_0 , e das derivadas das funções, J_1 , são dados como segue:

$$J_0(f(x), g(x)) \equiv \sum_{n=0}^N \frac{\sigma^n}{|e|^\beta} [f(x)] [g(x)], \quad (4.257)$$

$$J_1(f, g) \equiv \sum_{n=1}^{N-1} \frac{\sigma^n}{|e|^\beta} [\nabla f \cdot \hat{n}_e] [\nabla g \cdot \hat{n}_e]. \quad (4.258)$$

Na equação (4.256) vamos introduzir parâmetros de pênalti apenas do tipo J_0 . A introdução destes parâmetros não afeta a igualdade pois estes termos geralmente se anulam, mas pode melhorar a estabilidade do método.

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t k \nabla p_w \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n k \nabla p_c \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_t k \nabla p_w\} [z] - \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \hat{n}_e \cdot \{\lambda_n k \nabla p_c\} [z] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} [p_w] + \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} [p_c] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_t k \nabla z \cdot \hat{n}_e\} p_{dir.} - \sum_{e \in \Gamma_-} \int_e \{\lambda_n k \nabla z \cdot \hat{n}_e\} p_c (s_{dir.}) + \\
&+ J_0(z, p_w) + J_0(z, p_c), \quad (4.259)
\end{aligned}$$

onde os termos introduzidos em (4.259), $J_0(z, p_w)$ e $J_0(z, p_c)$ são, [22],

$$J_0(z, p_w) + J_0(z, p_c) = J_0(z, p_w + p_c) \equiv \sum_{e \in \Gamma_h \cup \partial\Omega} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w + p_c]. \quad (4.260)$$

Uma análise das estimativas de erro para este tipo de problema pode ser encontrada na referência [10].

Desmembrando os termos em (4.260) e analisando a aplicação dos termos de fronteira em Γ_- e Γ_+ vem

$$\begin{aligned}
J_0(z, p_w + p_c) &\equiv \sum_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w] + \sum_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \int_e [z] [p_c] + \\
&+ \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e \underbrace{z}_{[z]} \underbrace{p_{int}^-}_{[p_w]} - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z \underbrace{p_{dir}^-}_{p_{dir}} + \\
&+ \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e \underbrace{z}_{[z]} \underbrace{p_c(s_{int}^-)}_{[p_c]} - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z p_c(s_{dir}^-) + \\
&+ \sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e \underbrace{z}_{[z]} \underbrace{p_{int}^+}_{[p_w]} - \sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e z \underbrace{p_{dir}^+}_{p_{dir}} + \\
&+ \sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e \underbrace{z}_{[z]} \underbrace{p_c(s_{int}^+)}_0 - \sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e z \underbrace{p_c(s_{dir}^+)}_0. \quad (4.261)
\end{aligned}$$

Aplicando as alterações marcadas na equação acima, obtemos

$$\begin{aligned}
J_0(z, p_w + p_c) &\equiv \underbrace{\sum_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w]}_1 + \underbrace{\sum_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \int_e [z] [p_c]}_2 + \\
&+ \underbrace{\sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w]}_1 - \underbrace{\sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z p_{dir}^-}_3 + \\
&+ \underbrace{\sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [z] [p_c]}_2 - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z p_c(s_{dir}^-) + \\
&+ \underbrace{\sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w]}_1 - \underbrace{\sum_{e \in \Gamma_+} \frac{\sigma}{|e|^\beta} \int_e z p_{dir}^+}_3. \quad (4.262)
\end{aligned}$$

Reunindo na equação acima os termos sublinhados com mesmo valor (1, 2 e 3), obtemos a seguinte equação

$$\begin{aligned}
J_0(z, p_w + p_c) &\equiv \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [z] [p_w] + \sum_{e \in \Gamma_h \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [z] [p_c] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z p_{dir} - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e z p_c(s_{dir}^-). \quad (4.263)
\end{aligned}$$

Substituindo a equação (4.263) em (4.259) obtemos finalmente a equação para a pressão,

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t k \nabla p_w \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n k \nabla p_c \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_t k \nabla p_w \cdot \hat{n}_e \} [z] - \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{ \lambda_n k \nabla p_c \cdot \hat{n}_e \} [z] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_t k \nabla z \cdot \hat{n}_e \} [p_w] + \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{ \lambda_n k \nabla z \cdot \hat{n}_e \} [p_c] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_t k \nabla z \cdot \hat{n}_e p_{dir} - \sum_{e \in \Gamma_-} \int_e \lambda_n k \nabla z \cdot \hat{n}_e p_c (s_{dir}) + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [p_w] [z] + \sum_{e \in \Gamma_h \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [p_c] [z] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e p_{dir} z - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e p_c (s_{dir}) z. \tag{4.264}
\end{aligned}$$

Para implementar um método implícito, as quantidades que evoluem temporalmente serão avaliadas num tempo $i + 1$, conforme segue

$$\begin{aligned}
\int_{\Omega} (q_w + q_n) z &= \sum_{E \in \varepsilon_h} \int_E \lambda_t^{i+1} k \nabla p_w^{i+1} \cdot \nabla z + \sum_{E \in \varepsilon_h} \int_E \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot \nabla z + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_t^{i+1} k \nabla p_w^{i+1} \cdot \hat{n}_e \} [z] + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{ \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot \hat{n}_e \} [z] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_t^{i+1} k \nabla z \cdot \hat{n}_e \} [p_w^{i+1}] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_-} \int_e \{ \lambda_n^{i+1} k \nabla z \cdot \hat{n}_e \} [p_c^{i+1}] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_t^{i+1} k \nabla z \cdot \hat{n}_e p_{dir} - \sum_{e \in \Gamma_-} \int_e \lambda_n^{i+1} k \nabla z \cdot \hat{n}_e p_c (s_{dir}) + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [p_w^{i+1}] [z] + \sum_{e \in \Gamma_h \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [p_c^{i+1}] [z] + \\
&- \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e p_{dir} z - \sum_{e \in \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e p_c (s_{dir}) z. \tag{4.265}
\end{aligned}$$

Na sequência vamos obter a equação da saturação usando o método de Galerkin descontínuo. A equação que dá a evolução temporal da saturação pode ser escrita como

segue

$$-\phi \frac{\partial}{\partial t} s_n - \nabla \cdot (\lambda_w k \nabla p_w) = q_w, \quad (4.266)$$

primeiramente, escrevendo a equação (4.266) em termos de operadores vem

$$\begin{aligned} \nabla \cdot \vec{f} + S + Q &\equiv \hat{L}\vec{f}, \\ \hat{L}\vec{f} &= 0, \end{aligned} \quad (4.267)$$

onde foram definidas as quantidades

$$\vec{f} \equiv \lambda_w k \nabla p_w, \quad S \equiv \phi \frac{\partial}{\partial t} s_n, \quad Q \equiv q_w, \quad (4.268)$$

observando a equação (4.267) vemos que o operador \hat{L} atua em \vec{f} e o resultado é nulo, mas se ao invés de usar \vec{f} usarmos uma aproximação para \vec{f} , denotada por $\vec{f}_{approx.}$, que é uma expansão através de uma base, então ao invés de \hat{L} atuando em $\vec{f}_{approx.}$ produzir um resultado nulo o resultado será um resíduo R , tão menor quanto melhor for a aproximação. Deste modo, baseado em (4.267) podemos escrever

$$\hat{L}\vec{f}_{approx.} = R. \quad (4.269)$$

Fazendo o produto interno do resíduo com uma função peso v obtemos

$$(R, v) = \int_{\Omega} \hat{L}\vec{f}_{approx.} v, \quad (4.270)$$

e por comodidade, vamos denotar a função aproximada $\vec{f}_{approx.}$ apenas por \vec{f} , ou seja,

$$(R, v) = \int_{\Omega} \hat{L}\vec{f} v = \int_{\Omega} (\nabla \cdot \vec{f} + S + Q) v. \quad (4.271)$$

Para construir vários métodos da classe dos métodos de resíduos ponderados basta considerar o produto interno (R, v) como nulo,

$$\begin{aligned} (R, v) &= 0, \\ \int_{\Omega} (\nabla \cdot \vec{f} + S + Q) v &= 0. \end{aligned} \quad (4.272)$$

Para o caso em que a função peso v seja escolhida como funções da mesma base usada para expandir \vec{f} , o método obtido é o método de Galerkin.

A equação (4.272) pode ser reescrita usando as definições de (4.268), conforme segue

$$\begin{aligned} 0 &= \int_{\Omega} \left(\nabla \cdot \vec{f} + S + Q \right) v, \\ \int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} - \int_{\Omega} v \nabla \cdot \vec{f}. \end{aligned} \quad (4.273)$$

O uso da identidade vetorial seguinte será útil para aplicar o teorema de Gauss,

$$\begin{aligned} \nabla \cdot (v \vec{f}) &= \vec{f} \cdot \nabla v + v \nabla \cdot \vec{f}, \\ v \nabla \cdot \vec{f} &= \nabla \cdot (v \vec{f}) - \vec{f} \cdot \nabla v. \end{aligned} \quad (4.274)$$

Substituindo a identidade da equação (4.274) em (4.273) obtemos

$$\int_{\Omega} q_w v = - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} - \int_{\Omega} \nabla \cdot (v \vec{f}) + \int_{\Omega} \vec{f} \cdot \nabla v, \quad (4.275)$$

na equação (4.275) o termo $\int_{\Omega} \nabla \cdot (v \vec{f})$ pode ser trocado usando o teorema da divergência de Gauss, além disso o volume total do domínio Ω pode ser trocado pela soma dos volumes dos elementos E enquanto o fluxo através das faces dos volumes será através da soma de todas as faces e compartilhadas pelos elementos E_{i-1} e E_i vizinhos.

$$\int_{\Omega} q_w v = - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_{\Omega} \vec{f} \cdot \nabla v - \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e [v \vec{f}] \cdot \hat{n}_e, \quad (4.276)$$

conforme (4.244) é possível reescrever o fator $[v \vec{f}]$ de (4.276) como segue

$$[v \vec{f}] = \{v\} [\vec{f}] + [v] \{ \vec{f} \}. \quad (4.277)$$

Substituindo (4.277) em (4.276) vem

$$\begin{aligned} \int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_{\Omega} \vec{f} \cdot \nabla v + \\ &- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{v\} [\vec{f} \cdot \hat{n}_e] - \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e [v] \{ \vec{f} \cdot \hat{n}_e \}, \end{aligned} \quad (4.278)$$

o próximo passo é substituir a definição de \vec{f} conforme a equação (4.268) em (4.278), logo

$$\begin{aligned}
\int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_{\Omega} \lambda_w k \nabla p_w \cdot \nabla v + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e] + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e [v] \{ \lambda_w k \nabla p_w \cdot \hat{n}_e \}. \tag{4.279}
\end{aligned}$$

O termo de (4.279) que possui salto no gradiente de pressão, $\int_e \{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e]$, pode ser melhor analisado conforme a identidade seguinte

$$\hat{n}_e \cdot \nabla \left(\underbrace{\{v\} [\lambda_w k p_w]}_0 \right) = \{ \nabla v \cdot \hat{n}_e \} [\lambda_w k p_w] + \{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e], \tag{4.280}$$

na equação (4.280) o salto da pressão p_w está sendo considerado nulo, ou seja a função p_w é contínua, isto implica em

$$\begin{aligned}
0 &= \{ \nabla v \cdot \hat{n}_e \} [\lambda_w k p_w] + \{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e], \\
\{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e] &= - \{ \nabla v \cdot \hat{n}_e \} [\lambda_w k p_w], \\
\int_e \{v\} [\lambda_w k \nabla p_w \cdot \hat{n}_e] &= - \int_e \{ \lambda_w k \nabla v \cdot \hat{n}_e \} [p_w]. \tag{4.281}
\end{aligned}$$

Substituindo (4.281) em (4.279) vem

$$\begin{aligned}
\int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_{\Omega} \lambda_w k \nabla p_w \cdot \nabla v + \\
&+ \underbrace{\sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_w k \nabla v \cdot \hat{n}_e \} [p_w]}_I + \\
&- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e [v] \{ \lambda_w k \nabla p_w \cdot \hat{n}_e \}, \tag{4.282}
\end{aligned}$$

mas em (4.282) o termo sublinhado com I pode ser descrito como segue

$$\begin{aligned}
I &\equiv \sum_{e \in \Gamma_h} \int_e \{ \lambda_w k \nabla v \cdot \hat{n}_e \} [p_w] + \\
&+ \sum_{e \in \Gamma_+} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{w.int.}^+ - \sum_{e \in \Gamma_+} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}^+ + \\
&+ \sum_{e \in \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{w.int.}^- - \sum_{e \in \Gamma_+} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}^-, \tag{4.283}
\end{aligned}$$

e de maneira similar ao que foi feito de (4.252)-(4.254), podemos reescrever a equação acima como

$$\begin{aligned}
I &\equiv \underbrace{\sum_{e \in \Gamma_h} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w]}_1 + \\
&+ \underbrace{\sum_{e \in \Gamma_+} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w]}_1 - \underbrace{\sum_{e \in \Gamma_+} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}}_2 + \\
&+ \underbrace{\sum_{e \in \Gamma_-} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w]}_1 - \underbrace{\sum_{e \in \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}}_2; \tag{4.284}
\end{aligned}$$

em (4.284) devemos notar que $[p_w]$ assume o valor $[p_w] = p_{w.int}^+$ em Γ_+ , $[p_w]$ assume o valor $[p_w] = p_{w.int}^-$ em Γ_- , p_{dir} assume o valor $p_{dir} = p_{dir}^-$ em Γ_- e p_{dir} assume o valor $p_{dir} = p_{dir}^+$ em Γ_+ .

Nota-se ainda a possibilidade de reunir os termos sublinhados com 1, bem como os termos sublinhados com 2, em (4.284) ficando apenas

$$I \equiv \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}, \tag{4.285}$$

ao trocar (4.285) em (4.282) obtemos,

$$\begin{aligned}
\int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_E \lambda_w k \nabla p_w \cdot \nabla v - \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_w k \nabla p_w \cdot \hat{n}_e\} [v] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir}. \tag{4.286}
\end{aligned}$$

Finalmente vamos introduzir um termo de pênalti em (4.286), este termo não afeta equação mas é útil na estabilidade do método, sendo nulo quando a solução é exata, logo

$$\begin{aligned}
\int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_E \lambda_w k \nabla p_w \cdot \nabla v - \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_w k \nabla p_w \cdot \hat{n}_e\} [v] + \\
&+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{\lambda_w k \nabla v \cdot \hat{n}_e\} [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir} + \\
&+ J_0(v, p_w), \tag{4.287}
\end{aligned}$$

onde o termo J_0 é definido do mesmo modo que em (4.260),

$$J_0(v, p_w) \equiv \sum_{e \in \Gamma_h \cup \partial\Omega} \frac{\sigma}{|e|^\beta} \int_e [v] [p_w]. \quad (4.288)$$

Similar ao efetuado de (4.260)-(4.263) a equação (4.287) pode ser reescrita da seguinte forma

$$J_0(v, p_w) \equiv \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [v] [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e v p_{dir}, \quad (4.289)$$

substituindo (4.289) em (4.287) ficamos com

$$\begin{aligned} \int_{\Omega} q_w v &= - \int_{\Omega} v \phi \frac{\partial s_n}{\partial t} + \sum_{E \in \varepsilon_h} \int_E \lambda_w k \nabla p_w \cdot \nabla v - \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_w k \nabla p_w \cdot \hat{n}_e \} [v] + \\ &+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_w k \nabla v \cdot \hat{n}_e \} [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_w k \nabla v \cdot \hat{n}_e p_{dir} + \\ &+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [v] [p_w] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e v p_{dir}. \end{aligned} \quad (4.290)$$

finalmente a discretização temporal do tipo Euler forward em (4.290) e a análise das quantidades que evoluem no tempo no instante $i + 1$, permite obter a discretização na forma implícita da equação da saturação, como segue

$$\begin{aligned} \int_{\Omega} q_w v &= - \int_{\Omega} \frac{\phi}{\Delta t} (s_n^{i+1} - s_n^i) v + \sum_{E \in \varepsilon_h} \int_E \lambda_w^{i+1} k \nabla p_w^{i+1} \cdot \nabla v + \\ &- \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_w^{i+1} k \nabla p_w^{i+1} \cdot \hat{n}_e \} [v] + \\ &+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \int_e \{ \lambda_w^{i+1} k \nabla v \cdot \hat{n}_e \} [p_w^{i+1}] + \\ &- \sum_{e \in \Gamma_+ \cup \Gamma_-} \int_e \lambda_w^{i+1} k \nabla v \cdot \hat{n}_e p_{dir} + \\ &+ \sum_{e \in \Gamma_h \cup \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e [p_w^{i+1}] [v] - \sum_{e \in \Gamma_+ \cup \Gamma_-} \frac{\sigma}{|e|^\beta} \int_e p_{dir} v. \end{aligned} \quad (4.291)$$

Para construir o Jacobiano que permitirá resolver o sistema de equações discretizadas vamos substituir as expansões para a pressão e saturação, bem como escolher as funções peso, como segue

$$(p_w^{i+1}, s_n^{i+1}, z, v) \equiv \left(\sum_{l_p=1}^{m_p} p_E^{l_p} \varphi_E^{l_p}, \sum_{l_s=1}^{m_s} s_E^{l_s} \varphi_E^{l_s}, \varphi_E^r, \varphi_E^r \right), \quad (4.292)$$

com $1 \leq r_p \leq m_p$, e $1 \leq r_s \leq m_s$; as derivações serão de acordo com os coeficientes, conforme segue,

$$\left(\frac{\partial p_w^{i+1}}{\partial p_E^{l_p}}, \frac{\partial s_n^{i+1}}{\partial p_E^{l_p}} \right) \equiv \left(\frac{\partial}{\partial p_E^{l_p}} \sum_{l_p=1}^{m_p} p_E^{l_p} \varphi_E^{l_p}, \frac{\partial}{\partial p_E^{l_p}} \sum_{l_s=1}^{m_s} s_E^{l_s} \varphi_E^{l_s} \right) = (\varphi_E^{l_p}, 0), \quad (4.293)$$

$$\left(\frac{\partial p_w^{i+1}}{\partial s_E^{l_s}}, \frac{\partial s_n^{i+1}}{\partial s_E^{l_s}} \right) \equiv \left(\frac{\partial}{\partial s_E^{l_s}} \sum_{l_p=1}^{m_p} p_E^{l_p} \varphi_E^{l_p}, \frac{\partial}{\partial s_E^{l_s}} \sum_{l_s=1}^{m_s} s_E^{l_s} \varphi_E^{l_s} \right) = (0, \varphi_E^{l_s}). \quad (4.294)$$

Vamos calcular as contribuições das integrais de volume da equação da pressão,

$$c.i.v. (p_E^{l_p}) = \frac{\partial}{\partial p_E^{l_p}} \left(\int_E \lambda_t^{i+1} k \nabla p_w^{i+1} \cdot \nabla z + \int_E \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot \nabla z \right), \quad (4.295)$$

substituindo a função z de acordo com (4.292) vem

$$c.i.v. (p_E^{l_p}) = \frac{\partial}{\partial p_E^{l_p}} \underbrace{\left(\int_E \lambda_t k \nabla p_w^{i+1} \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot \nabla \varphi_E^{r_p} \right)}_{G^{r_p}} \equiv \frac{\partial G^{r_p}}{\partial p_E^{l_p}}, \quad (4.296)$$

em seguida, atuando com o operador de derivação na equação (4.296),

$$\frac{\partial G^{r_p}}{\partial p_E^{l_p}} = \left(\int_E \lambda_t k \nabla \underbrace{\frac{\partial p_w^{i+1}}{\partial p_E^{l_p}}}_{\text{eq. (83)}} \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n k \nabla \frac{\partial p_c}{\partial s_n^{i+1}} \underbrace{\frac{\partial s_n^{i+1}}{\partial p_E^{l_p}}}_{\text{eq. (83)}} \cdot \nabla \varphi_E^{r_p} \right). \quad (4.297)$$

Substituindo (4.293) em (4.297) obtemos

$$\begin{aligned} \frac{\partial G^{r_p}}{\partial p_E^{l_p}} &= \int_E \lambda_t k \nabla \varphi_E^{l_p} \cdot \nabla \varphi_E^{r_p} + 0, \\ \frac{\partial G^{r_p}}{\partial p_E^{l_p}} &= \int_E \lambda_t k \begin{pmatrix} \nabla \varphi_E^{l_p=1} \cdot \nabla \varphi_E^{r_p=1} & \nabla \varphi_E^{l_p=2} \cdot \nabla \varphi_E^{r_p=1} & \dots \\ \nabla \varphi_E^{l_p=1} \cdot \nabla \varphi_E^{r_p=2} & \nabla \varphi_E^{l_p=2} \cdot \nabla \varphi_E^{r_p=2} & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}. \end{aligned} \quad (4.298)$$

De maneira similar é obtida a contribuição da integral do volume devido à saturação, como segue

$$c.i.v. (s_E^{l_s}) = \frac{\partial}{\partial s_E^{l_s}} \underbrace{\left(\int_E \lambda_t k \nabla p_w^{i+1} \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot \nabla \varphi_E^{r_p} \right)}_{G^{r_p}} \equiv \frac{\partial G^{r_p}}{\partial s_E^{l_s}}, \quad (4.299)$$

$$\begin{aligned} \frac{\partial G^{r_p}}{\partial s_E^{l_s}} &= \int_E \frac{\partial \lambda_t}{\partial s_n^{i+1}} \underbrace{\frac{\partial s_n^{i+1}}{\partial s_E^{l_s}}}_{\text{eq. (84)}} k \nabla p_w^{i+1} \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_t k \nabla \underbrace{\frac{\partial p_w^{i+1}}{\partial s_E^{l_s}}}_{\text{eq. (84)}} \cdot \nabla \varphi_E^{r_p} + \\ &+ \int_E \frac{\partial \lambda_n}{\partial s_n^{i+1}} \underbrace{\frac{\partial s_n^{i+1}}{\partial s_E^{l_s}}}_{\text{eq. (84)}} k \nabla p_c \cdot \nabla \varphi_E^{r_p} + \\ &+ \int_E \lambda_n k \nabla \left(\underbrace{\frac{\partial p_c}{\partial s_n^{i+1}} \frac{\partial s_n^{i+1}}{\partial s_E^{l_s}}}_{\text{eq. (84)}} \right) \cdot \nabla \varphi_E^{r_p}, \end{aligned} \quad (4.300)$$

substituindo (4.294) em (4.300) vem

$$\begin{aligned} \frac{\partial G^{r_p}}{\partial s_E^{l_s}} &= \int_E \frac{\partial \lambda_t}{\partial s_n^{i+1}} \varphi_E^{l_s} k \nabla p_w^{i+1} \cdot \nabla \varphi_E^{r_p} + 0 + \\ &+ \int_E \frac{\partial \lambda_n}{\partial s_n^{i+1}} \varphi_E^{l_s} k \nabla p_c \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n k \nabla \left(\frac{\partial p_c}{\partial s_n^{i+1}} \varphi_E^{l_s} \right) \cdot \nabla \varphi_E^{r_p}, \end{aligned} \quad (4.301)$$

em seguida, mudando a notação de ∂s_n^{i+1} para $\partial s_E^{l_s}$, p_w^{i+1} para p_w

$$\begin{aligned} \frac{\partial G^{r_p}}{\partial s_E^{l_s}} &= \int_E \frac{\partial \lambda_t}{\partial s_E^{l_s}} \varphi_E^{l_s} k \nabla p_w \cdot \nabla \varphi_E^{r_p} + \int_E \frac{\partial \lambda_n}{\partial s_E^{l_s}} \varphi_E^{l_s} k \nabla p_c \cdot \nabla \varphi_E^{r_p} + \\ &+ \int_E \lambda_n k \varphi_E^{l_s} \nabla \left(\frac{\partial p_c}{\partial s_E^{l_s}} \right) \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n k \frac{\partial p_c}{\partial s_E^{l_s}} \nabla \varphi_E^{l_s} \cdot \nabla \varphi_E^{r_p}. \end{aligned} \quad (4.302)$$

A seguinte identidade será útil

$$\nabla \left(\frac{\partial p_c}{\partial s_E^{l_s}} \right) = \frac{\partial}{\partial s_E^{l_s}} \left(\frac{\partial p_c}{\partial s_E^{l_s}} \right) \nabla s_E^{l_s} = \frac{\partial^2 p_c}{\partial s_E^{l_s^2}} \nabla s_E^{l_s}, \quad (4.303)$$

finalmente, substituindo (4.303) em (4.302) obtemos

$$\begin{aligned}
\frac{\partial G^{r_p}}{\partial s_E^{l_s}} &= \int_E \frac{\partial \lambda_t}{\partial s_E^{l_s}} \varphi_E^{l_s} k \nabla p_w \cdot \nabla \varphi_E^{r_p} + \int_E \frac{\partial \lambda_n}{\partial s_E^{l_s}} \varphi_E^{l_s} k \nabla p_c \cdot \nabla \varphi_E^{r_p} + \\
&+ \int_E \lambda_n k \varphi_E^{l_s} \frac{\partial^2 p_c}{\partial s_E^{l_s^2}} \nabla s_n \cdot \nabla \varphi_E^{r_p} + \int_E \lambda_n k \frac{\partial p_c}{\partial s_E^{l_s}} \nabla \varphi_E^{l_s} \cdot \nabla \varphi_E^{r_p}. \quad (4.304)
\end{aligned}$$

para as fronteiras internas a contribuição é dada por

$$\begin{aligned}
c.f.i. \left(p_E^{l_p} \right) &= -\frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \{ \lambda_t^{i+1} k \nabla p_w^{i+1} \cdot n_e \} [z] + \frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \{ \lambda_t^{i+1} k \nabla z \cdot n_e \} [p_w^{i+1}] + \\
&+ \frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} [p_w^{i+1}] [z] - \frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \{ \lambda_n^{i+1} k \nabla p_c^{i+1} \cdot n_e \} [z] + \\
&+ \frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \{ \lambda_n^{i+1} k \nabla z \cdot n_e \} [p_c^{i+1}] + \frac{\partial}{\partial p_E^{l_p}} \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} [p_c^{i+1}] [z]; \quad (4.305)
\end{aligned}$$

para o elemento E_1 , mudando z por $\varphi_{E_1}^{r_p}$, λ_n^{i+1} por λ_n , λ_t^{i+1} por λ_t vem

$$\begin{aligned}
\frac{\partial G_{E_1}^{r_p}}{\partial p_{E_1}^{l_p}} &= - \int_{e \in \Gamma_h} \left\{ \lambda_t^1 k^1 \nabla \frac{\partial p_w^{i+1}}{\partial p_{E_1}^{l_p}} \cdot n_e \right\} [\varphi_{E_1}^{r_p}] + \int_{e \in \Gamma_h} \{ \lambda_t^1 k^1 \nabla \varphi_{E_1}^{r_p} \cdot n_e \} \left[\frac{\partial p_w^{i+1}}{\partial p_{E_1}^{l_p}} \right] + \\
&+ \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \left[\frac{\partial p_w^{i+1}}{\partial p_{E_1}^{l_p}} \right] [\varphi_{E_1}^{r_p}] - \int_{e \in \Gamma_h} \left\{ \lambda_n^1 k^1 \nabla \frac{\partial p_c^{i+1}}{\partial p_{E_1}^{l_p}} \cdot n_e \right\} [z] + \\
&+ \int_{e \in \Gamma_h} \{ \lambda_n^1 k^1 \nabla z \cdot n_e \} \left[\frac{\partial p_c^{i+1}}{\partial p_{E_1}^{l_p}} \right] + \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \left[\frac{\partial p_c^{i+1}}{\partial p_{E_1}^{l_p}} \right] [z], \quad (4.306)
\end{aligned}$$

ou simplesmente

$$\begin{aligned}
\frac{\partial G_{E_1}^{r_p}}{\partial p_{E_1}^{l_p}} &= - \int_{e \in \Gamma_h} \left\{ \lambda_t^1 k^1 \nabla \varphi_{E_1}^{l_p} \cdot n_e \right\} [\varphi_{E_1}^{r_p}] + \\
&+ \int_{e \in \Gamma_h} \{ \lambda_t^1 k^1 \nabla \varphi_{E_1}^{r_p} \cdot n_e \} [\varphi_{E_1}^{l_p}] + \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} [\varphi_{E_1}^{l_p}] [\varphi_{E_1}^{r_p}], \quad (4.307)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial G_{E_1}^{r_p}}{\partial p_{E_1}^{l_p}} &= -\frac{1}{2} \int_{e \in \Gamma_h} \lambda_t^1 k^1 \nabla \varphi_{E_1}^{l_p} \cdot n_e \varphi_{E_1}^{r_p} + \\
&+ \frac{1}{2} \int_{e \in \Gamma_h} \lambda_t^1 k^1 \nabla \varphi_{E_1}^{r_p} \cdot n_e \varphi_{E_1}^{l_p} + \int_{e \in \Gamma_h} \frac{\sigma}{|e|^\beta} \varphi_{E_1}^{l_p} \varphi_{E_1}^{r_p}. \quad (4.308)
\end{aligned}$$

4.4.1 Considerações gerais sobre o salto

Para discussão das propriedades abaixo, a figura (23) será usada como base; o salto de uma função f através da face compartilhada pelos elementos 1 e 2, pode ser definido

como segue

$$[f] = f_1 n_1^+ + f_2 n_2^+, \quad (4.309)$$

mas observando a orientação dos vetores unitários na figura (1), podemos notar que,

$$n_2^+ = -n_1^+, \quad (4.310)$$

substituindo (4.310) em (4.309) vem,

$$[f] = (f_1 - f_2) n_1^+, \quad (4.311)$$

trocando n_1^+ por $+1$, conforme a convenção do sinal na fronteira que pode ser novamente observado na figura (23), temos apenas

$$[f] = f_1 - f_2, \quad (4.312)$$

De uma forma genérica e equivalente, podemos definir o salto entre dois elementos que compartilham uma borda interna,

$$[f] = f_{i-1} n_{i-1} + f_i n_i, \quad (4.313)$$

da mesma forma que fizemos anteriormente é possível notar que

$$n_i^+ = -n_{i-1}^+. \quad (4.314)$$

Substituindo (4.314) em (4.313) obtemos

$$[f] = (f_{i-1} - f_i) n_{i-1}^+, \quad (4.315)$$

mas trocando n_{i-1}^+ por $+1$, ficamos apenas com

$$[f] = f_{i-1} - f_i, \quad (4.316)$$

4.4.2 Salto através do contorno Γ_-

O contorno Γ_- é a fronteira entre o elemento 1 e o exterior, como o salto está definido para dois elementos, vamos aplicar o salto supondo a existência de um elemento 0 que compartilharia a borda Γ_- com o elemento 1, sabendo que este elemento 0 não existe. Deste modo, o salto como definido em (4.313) fica

$$[f] = f_{i-1} n_{i-1}^+ + f_i n_i^+ = f_0 n_0^+ + f_1 n_1^+, \quad (4.317)$$

mas neste caso é possível fazer em (4.317) a substituição $n_0^+ = -n_1^+$, ou seja,

$$[f] = (f_1 - f_0) n_1^+, \quad (4.318)$$

para o caso $n_1^+ = 1$,

$$[f] = f_1 - f_0, \quad (4.319)$$

ou seja o salto através do contorno Γ_- é o valor da função no interior do elemento 1 subtraído do valor externo ao elemento 1, no caso o elemento fictício 0.

4.4.3 Salto através do contorno Γ_+

Analogamente ao discutido no cálculo do salto através do contorno Γ_- , temos

$$[f] = f_4 n_4^+ + f_5 n_5^+, \quad (4.320)$$

substituindo $n_5^+ = -n_4^+$ em (4.320), obtemos

$$[f] = (f_4 - f_5) n_4^+, \quad (4.321)$$

fazendo $n_4^+ = 1$ vem

$$[f] = f_4 - f_5, \quad (4.322)$$

ou seja o salto através do contorno Γ_+ é o valor da função no interior do elemento 4 subtraído do valor externo ao elemento 4, no caso o elemento fictício 5.

As discretizações abrem a possibilidade de implementação dos diferentes métodos numéricos discutidos nesta secção, através de códigos computacionais, primeiramente serão implementados códigos para equações diferenciais simples e posteriormente, serão realizadas aplicações dos códigos para simular as equações do escoamento bifásico.

5 ALGUMAS APLICAÇÕES

Neste capítulo apresentamos a implementação computacional de alguns casos estudados usando o programa R: escoamento bifásico em 1D com método de linhas, difusão do calor em 1D e 2D. Em seguida apresentamos também o estudo do escoamento bifásico usando o OpenFOAM.

5.1 Fluxo bifásico em meios porosos com método de linhas

É possível implementar as equações básicas do escoamento usando o método de linhas discutido no capítulo anterior. As equações da pressão e saturação foram deduzidas previamente no capítulo 3. O modelo fica estabelecido pelas equações e relações seguintes:

- Relações fundamentais:

$$P_c = P_n - P_w, \quad S_w + S_n = 1, \quad \lambda_n = \frac{k_{rn}}{\mu_n}, \quad \lambda_w = \frac{k_{rw}}{\mu_w}. \quad (5.1)$$

- Modelo de Brooks-Corey:

$$P_c(S_n) = P_d(1 - S_n)^{-\frac{1}{\theta}}, \quad k_{rw}(S_n) = (1 - S_n)^{\frac{2+3\theta}{\theta}}, \quad k_{rn}(S_n) = S_n^2 \left[1 - (1 - S_n)^{\frac{2+\theta}{\theta}} \right]. \quad (5.2)$$

- Equações para pressão e saturação:

$$\begin{aligned} -\vec{\nabla} \cdot \left(\lambda_t k \vec{\nabla} P_w + \lambda_n k \vec{\nabla} P_c \right) &= q_w + q_n, \\ -\phi \frac{\partial}{\partial t} S_n - \vec{\nabla} \cdot \left(k \lambda_w \vec{\nabla} P_w \right) &= q_w. \end{aligned} \quad (5.3)$$

- Saturação inicial

$$S_n = \begin{cases} 0.15, & \{x = x_i\}, \\ 0.80, & \{x > x_i\}. \end{cases} \quad (5.4)$$

- Equação da pressão em 1D

$$0 = - [k\partial_x (\lambda_n + \lambda_w) \partial_x p_w + k (\lambda_n + \lambda_w) \partial_x^2 p_w + k\partial_x (\lambda_n) \partial_x p_c + k\lambda_n \partial_x^2 p_c], \quad (5.5)$$

$$0 = - \underbrace{k\partial_x (\lambda_n + \lambda_w)}_{I(x)} p_{wII} - \underbrace{k (\lambda_n + \lambda_w)}_{II(x)} \partial_x p_{wII} - \underbrace{k\partial_x (\lambda_n)}_{III(x)} \underbrace{\partial_x p_c}_{IV(x)} - \underbrace{k\lambda_n}_{V(x)} \underbrace{\partial_x^2 p_c}_{VI(x)}. \quad (5.6)$$

- Equação da saturação em 1D

$$\frac{ds_n}{dt} = -\frac{1}{\phi} [k\partial_x (\lambda_w) \partial_x p_w + k\lambda_w \partial_x^2 p_w]. \quad (5.7)$$

As equações a serem implementadas pelo método de linhas são

$$p_w \equiv p_{WI}, \quad p_{wII} \equiv \frac{d}{dx} p_{WI}, \quad (5.8)$$

$$\frac{d}{dx} p_{wII} = - [I(x) p_{wII} + III(x) IV(x) + V(x) VI(x)] \frac{1}{II(x)}, \quad (5.9)$$

$$\frac{ds_n}{dt} = -\frac{1}{\phi} [k\partial_x (\lambda_w) \partial_x p_w + k\lambda_w \partial_x^2 p_w]. \quad (5.10)$$

É possível notar em (5.8) e (5.9) que a equação da pressão teve sua ordem reduzida, pois os integradores do método de linhas integram apenas sistemas de equações de primeira ordem. Adicionalmente, temos os seguintes parâmetros,

$$\begin{aligned} x &= [0, 100], h_x = 0.400, t = [0, 1], h_t = 0.600, \\ \mu_n &= 10^{-2}, P_{winj} = 3 \times 10^6, P_{wprod} = 10^6, \\ k &= 10^{-8}, \phi = 0.2, P_d = 10^3, \theta = 2, \mu_w = 10^{-3}, \end{aligned} \quad (5.11)$$

$$(x_i, y_i) = (0, 0.2); (x_j, y_j) = (0, 0.8), C_3 = 0.300, \quad (5.12)$$

$$C_1 = 0.267, C_2 = 0.333, \quad (5.13)$$

$$s_n(x) = \frac{0.267}{0.333 + e^{-0.300x}}. \quad (5.14)$$

O programa que implementa as equações acima, denominado 2phase.flow.1.R, consta no apêndice A e permite obter os seguintes resultados

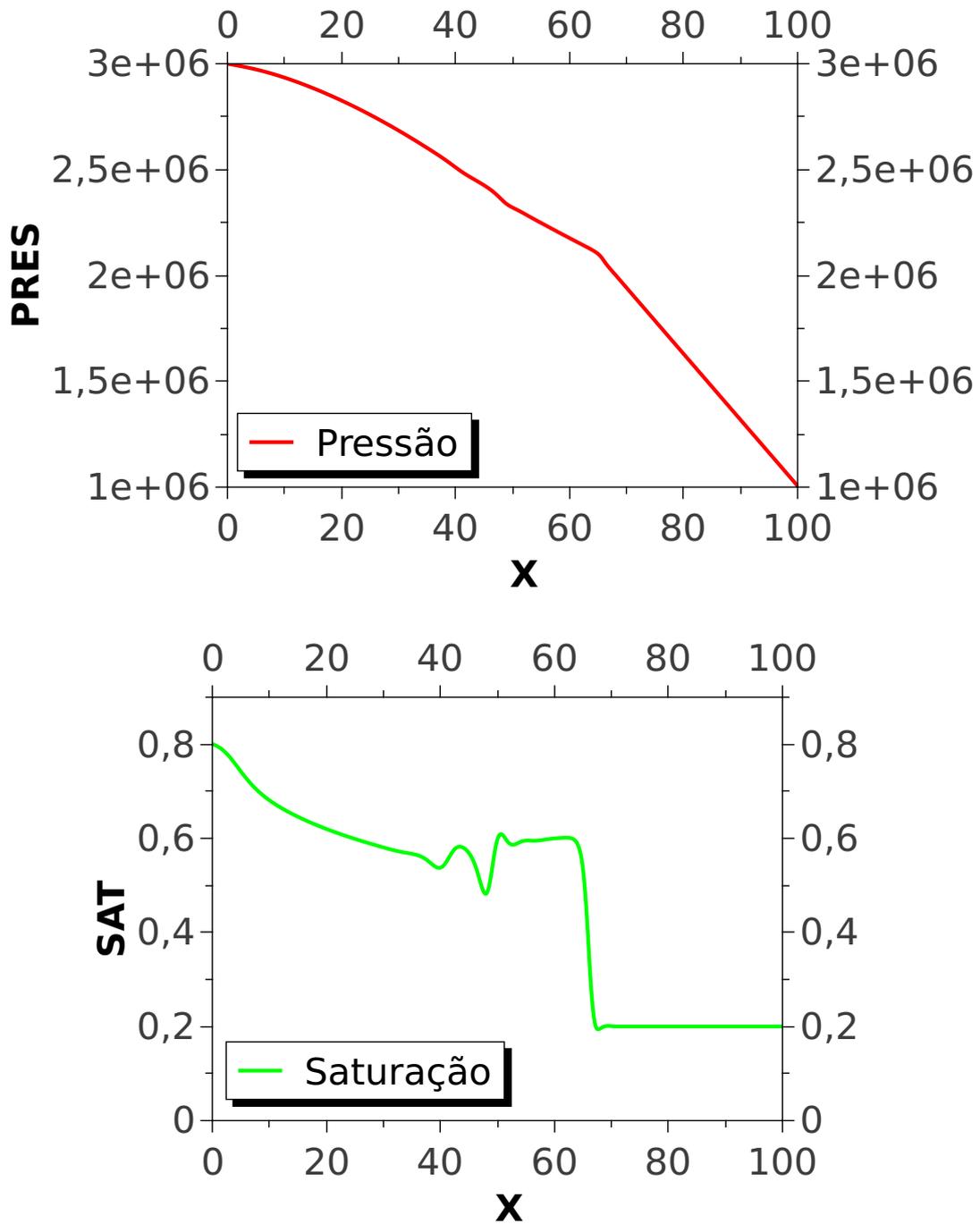


Figura 24: Curvas de pressão (superior) e de saturação (inferior) para simulação de 300 dias de escoamento de dois fluidos em um meio poroso utilizando o método de linhas. Foram usados $[\Delta t, \Delta x] = [0.01, 0.1]$. A pressão é dada em Pascal Pa , a posição é dada em metros m e a saturação é adimensional.

Em seguida são apresentados os valores de referência, para comparação, obtidos usando o esquema IMPES (Implicit Pressure Explicit Saturation) implementado no software livre OpenFOAM, listado no solver disponível no apêndice A.

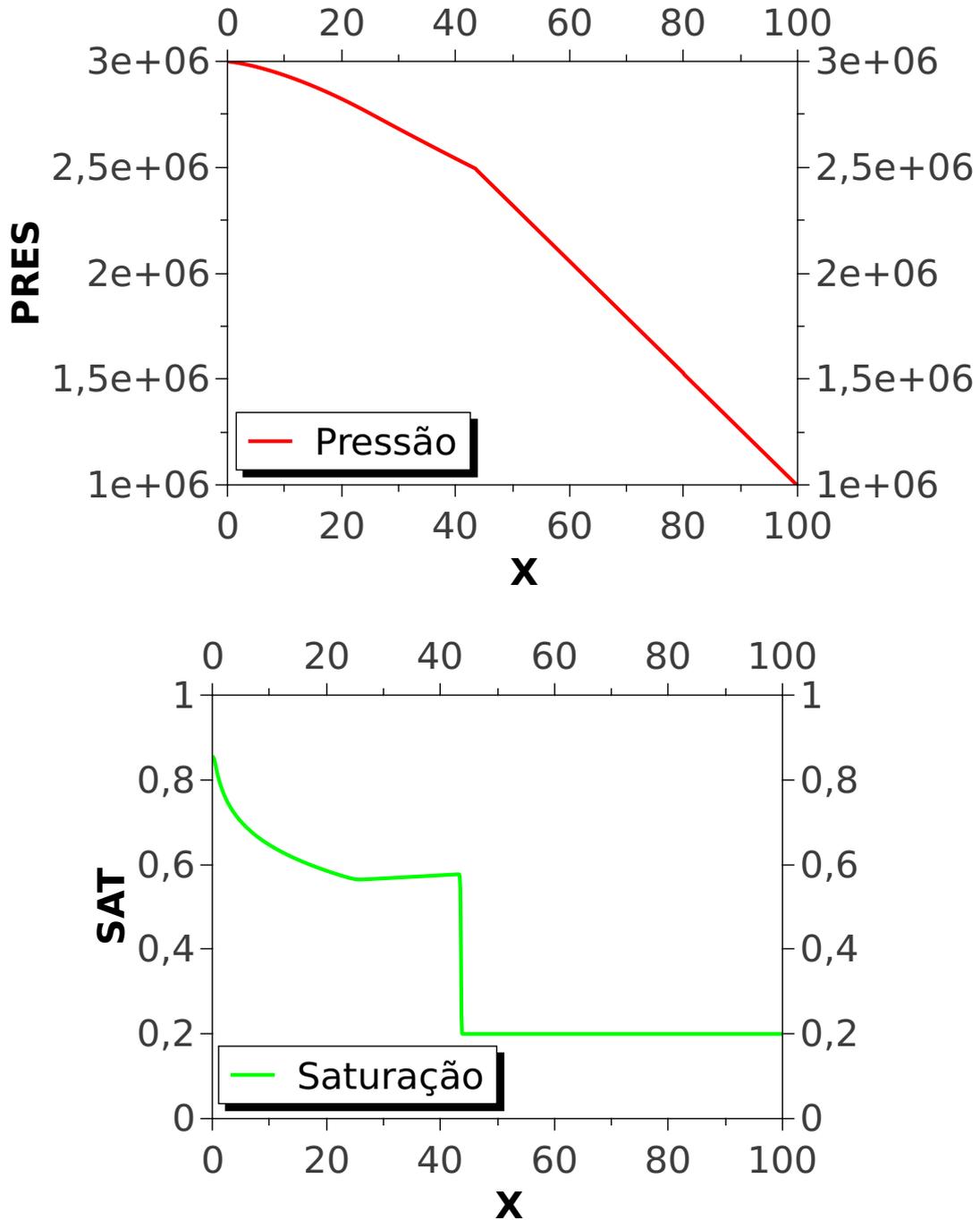


Figura 25: Curvas de pressão (superior) e de saturação (inferior) para simulação de 300 dias de escoamento de dois fluidos em um meio poroso utilizando o software OpenFOAM. A pressão é dada em Pascal Pa , a posição é dada em metros m e a saturação é adimensional.

Considerando as figuras (24) e (25) notamos que há uma diferença significativa entre as curvas de saturação. A saturação obtida pelo método de linhas apresenta uma oscilação indesejável (instabilidade), além disso a frente da referida curva de saturação está quase $20m$ adiantada com relação à curva de referência. O adiantamento é esperado, pois o método de linhas implementado não roda curvas abruptas (diverge), então a saturação foi suavizada para que a simulação pudesse rodar, e a frente de saturação implementada pelo método de linhas já partiu quase $20m$ adiantada, conforme a figura abaixo,

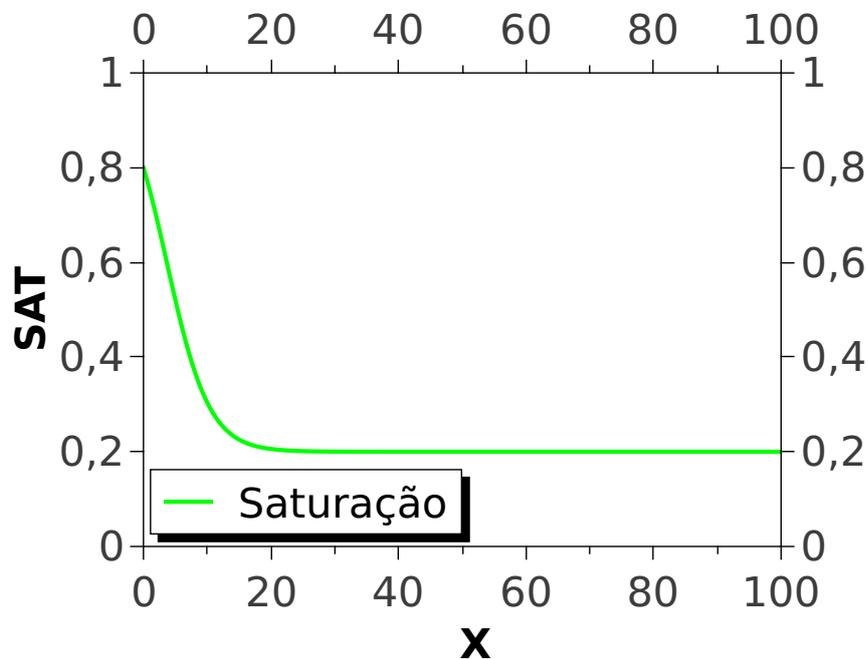


Figura 26: Campo de saturação inicial suavizado utilizado no método de linhas. A posição é dada em metros m e a saturação é adimensional.

Comparando as soluções obtidas pelo método de linhas e pelo método de volumes finitos (OpenFOAM), notamos que o método de linhas não é uma boa escolha. O método de linhas apresenta oscilações na solução da equação da saturação mesmo após a utilização de filtros do tipo “smooth interpolation”. Essas oscilações tendem a crescer com o avanço do tempo.

A estabilidade depende do método de discretização. No caso do método de linhas, o fato de não discretizarmos todas as derivadas não nos permite efetuar uma análise da

estabilidade, pois uma das derivadas ficou sob demanda do conjunto de integradores do R. No entanto observamos do experimento numérico que o método das linhas implementado é instável.

5.2 Aplicações do Método de Volumes Finitos

A seguir vamos realizar algumas aplicações do método de volumes finitos, com o objetivo de produzir códigos computacionais que possam ser reaproveitados para a simulação de reservatórios. Estes códigos serão construídos baseando-se na discretização da forma integral da equação geral do transporte, [27].

5.3 Equação Geral do Transporte

A equação geral do transporte pode ser escrita na seguinte forma, [27],

$$\underbrace{\frac{\partial}{\partial t}(\rho\phi)}_{\text{transiente}} + \underbrace{\vec{\nabla} \cdot (\rho\phi\vec{u})}_{\text{convecção}} = \underbrace{\vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi)}_{\text{difusão}} + \underbrace{S}_{\text{fontes}}, \quad (5.15)$$

Na equação (5.15) o termo transiente está associado com a taxa de variação temporal da propriedade estudada, o termo de convecção está associado com a parcela da propriedade que ao ser transportada sofre o efeito da corrente de deslocamento através do meio, podendo ser transportada mais rapidamente ou mais lentamente dependendo da intensidade da corrente, o termo da difusão está associado com a variação da propriedade em cada ponto, e por último, o termo de fonte corresponde a uma taxa de produção ou destruição de uma propriedade no meio.

Em seguida estudaremos os esquemas de discretização do termo de difusão da equação (5.15).

5.3.1 Difusão no Estado Estacionário

Esta análise será feita sobre um problema específico do fluxo de calor em uma dimensão, de forma estacionária, onde as temperaturas no início e final do domínio são dadas respectivamente por $T_A = 100$ e $T_B = 500$, sendo a distância entre dois pontos nodais

consecutivos¹ $\delta x = 0.1m$, mostrado na figura abaixo.

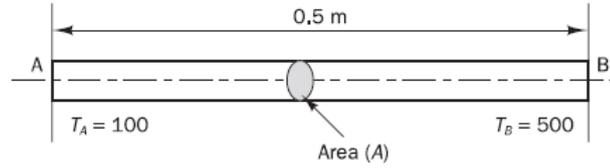


Figura 27: Ilustração do domínio onde a difusão do calor será resolvida, representando uma barra com área de secção transversal negligenciável.

A conservação do calor através de um volume é descrita pela lei da conservação da energia térmica. Já a forma do fluxo de calor com relação à temperatura é dada pela lei de Fourier. Reunindo estas considerações é possível formular o problema do fluxo de calor como segue,

$$\begin{aligned}\frac{\partial}{\partial t}T &= -\vec{\nabla} \cdot \vec{q}, \\ \vec{q} &= -k\vec{\nabla}T, \\ \frac{\partial}{\partial t}T &= \vec{\nabla} \cdot (k\vec{\nabla}T).\end{aligned}\quad (5.16)$$

É interessante notar que a equação (5.16) é um caso particular da equação (5.15) conforme pode ser observado abaixo,

$$\begin{aligned}\frac{\partial}{\partial t}(T) + 0 &= \vec{\nabla} \cdot (k\vec{\nabla}T) + 0, \\ \frac{\partial}{\partial t}(\rho\phi) + \underbrace{\vec{\nabla} \cdot (\rho\phi\vec{u})}_0 &= \vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi) + \underbrace{S}_0, \\ \frac{\partial}{\partial t}(\phi) + 0 &= \vec{\nabla} \cdot \left(\frac{\Gamma}{\rho}\vec{\nabla}\phi\right) + 0,\end{aligned}\quad (5.17)$$

ou seja, basta anular o termo de fonte e o de convecção para estabelecer a semelhança entre (5.16) e (5.15), conforme podemos observar em (5.17).

Para o caso estacionário basta desconsiderar em (5.17) a parte temporal, ou seja,

¹A figura (27) foi retirada da referência [27].

$$\underbrace{\frac{\partial}{\partial t}}_0(T) = \vec{\nabla} \cdot (k \vec{\nabla} T) \Rightarrow 0 = \vec{\nabla} \cdot (k \vec{\nabla} T),$$

$$\underbrace{\frac{\partial}{\partial t}}_0(\phi) = \vec{\nabla} \cdot \left(\frac{\Gamma}{\rho} \vec{\nabla} \phi \right) \Rightarrow 0\rho = \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi). \quad (5.18)$$

De (5.18) é possível notar a equivalência $\{\phi, \Gamma\} \equiv \{T, k\}$. Esta equivalência é importante, pois ao resolver o problema de difusão e discretizar o termo $\vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi)$ será possível em ocasiões futuras aproveitar a discretização deste termo, apenas declarando ϕ e Γ de acordo com a equação a ser resolvida.

Em seguida obteremos a forma discreta do termo de difusão. ²

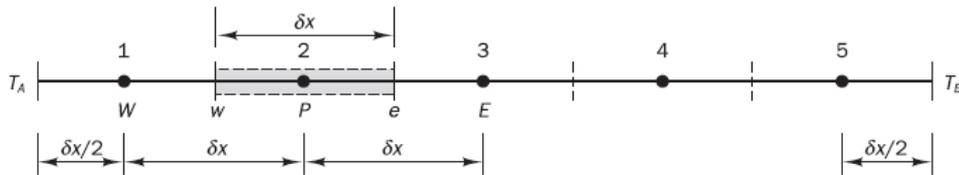


Figura 28: Discretização do domínio em volumes finitos, a quantidade de interesse deve ser obtida nos pontos contidos no centro de cada volume.

O domínio é subdividido em células, cada célula contendo em seu interior um ponto nodal que possui duas faces vizinhas encerrando a célula. Para cada ponto nodal, a localização dos seus vizinhos e das faces vizinhas segue a notação de bússola com letras maiúsculas e minúsculas respectivamente.

Devido ao uso do teorema da divergência os valores dos fluxos são calculados nas faces

²A figura (28) foi retirada da referência [27].

$$\begin{aligned}
\int_V \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi) dV &= \int_S (\Gamma \vec{\nabla} \phi) \cdot d\vec{A}, \\
&= \int_e (\Gamma \vec{\nabla} \phi) \cdot d\vec{A}_e + \int_w (\Gamma \vec{\nabla} \phi) \cdot d\vec{A}_w, \\
&= A_e \left(\Gamma \frac{\partial \phi}{\partial x} \right)_e - A_w \left(\Gamma \frac{\partial \phi}{\partial x} \right)_w, \\
&\simeq A_e \Gamma_e \frac{\phi_E - \phi_P}{\delta x} - A_w \Gamma_w \frac{\phi_P - \phi_W}{\delta x}.
\end{aligned} \tag{5.19}$$

A aproximação em (5.19) pode ser simplificada considerando $A_e = A_w = A$, pois na ocasião a discretização de (5.19) será relacionada com a equação (5.18); não é interessante fazer o mesmo com Γ , pois em ocasiões posteriores será possível reutilizar Γ como um valor que muda em cada face (permeabilidade, permissividade, condutividade, etc...) .

Simplificando (5.19) ficamos com

$$\frac{0\rho}{A} = \int_V \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi) dV \simeq \Gamma_e \frac{\phi_E - \phi_P}{\delta x} - \Gamma_w \frac{\phi_P - \phi_W}{\delta x}. \tag{5.20}$$

De maneira equivalente, a equação (5.20) pode ser organizada em termos dos coeficientes

$$0 = \frac{\Gamma_w}{\delta x_{PW}} \phi_W + \left(-\frac{\Gamma_w}{\delta x_{PW}} - \frac{\Gamma_e}{\delta x_{EP}} \right) \phi_P + \frac{\Gamma_e}{\delta x_{EP}} \phi_E; \tag{5.21}$$

os coeficientes de (5.21) podem ser renomeados usando a seguinte notação,

$$\begin{aligned}
0 &= a_W \phi_W + a_P \phi_P + a_E \phi_E, \\
\{a_W, a_E, a_P\} &= \left\{ \frac{\Gamma_w}{\delta x}, \frac{\Gamma_e}{\delta x}, -a_W - a_E \right\}.
\end{aligned} \tag{5.22}$$

A aplicação de (5.21) e (5.22) no domínio da figura abaixo³ é para os pontos nodais internos. As condições de fronteira estão associadas com os pontos nodais inicial e final. Na ocasião do exemplo tratado, será possível construir o seguinte sistema linear

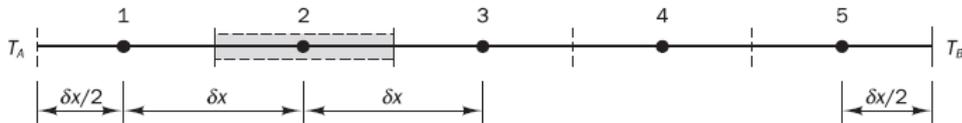


Figura 29: Distância entre pontos nodais consecutivos e distância entre os pontos nodais e as faces nas regiões de fronteira do domínio.

³A figura (29) foi retirada da referência [27].

$$A_{i \times i} \phi_{i \times 1} = C_{i \times 1}, \quad (5.23)$$

e o vetor solução será obtido resolvendo-se o sistema linear,

$$\phi_{i \times 1} = A_{i \times i}^{-1} C_{i \times 1}. \quad (5.24)$$

Para usar uma notação que seja compatível com a matricial é necessário converter a notação de bússola em notação indicial. Para isso vamos estabelecer uma correspondência entre $\{W, E, S, N, B, T\}$ e os índices $\{i, j, k\}$ conforme a figura seguinte

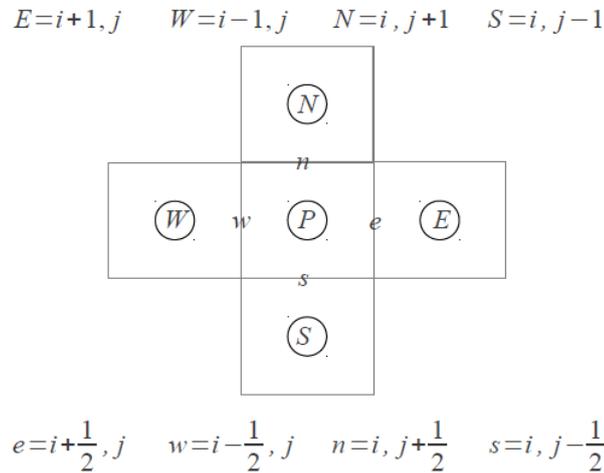


Figura 30: Equivalência entre a notação de bússola e a notação indicial, útil para montar os esquemas na forma matricial.

Vamos estabelecer um esquema para a construção da matriz partindo da equação (5.22) com os pontos nodais variando de 2 até 4. A seguir nos preocuparemos com os pontos 1 e 5, pois estes envolvem os valores na fronteira. Então

$$\begin{aligned} a_W \phi_W + a_P \phi_P + a_E \phi_E &= 0, \\ a_{i,i-1} \phi_{i-1} + a_{i,i} \phi_i + a_{i,i+1} \phi_{i+1} &= 0. \end{aligned} \quad (5.25)$$

Para os pontos nodais de 2 até 4 ficamos com as equações seguintes

ponto nodal	Equação
1	...
2	$a_{21}\phi_1 + a_{22}\phi_2 + a_{23}\phi_3 = 0$
3	$a_{32}\phi_2 + a_{33}\phi_3 + a_{34}\phi_4 = 0$
4	$a_{43}\phi_3 + a_{44}\phi_4 + a_{45}\phi_5 = 0$
5	...

Tabela 3: Equações discretizadas para os pontos nodais internos, estas equações serão úteis para construir o sistema linear a ser resolvido.

Das equações na tabela acima é possível notar que a equação matricial correspondente ao sistema em questão é da forma

$$\begin{pmatrix} \dots & \dots & \dots & \dots & \dots \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ \dots & \dots & \dots & \dots & \dots \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{pmatrix} = \begin{pmatrix} \dots \\ 0 \\ 0 \\ 0 \\ \dots \end{pmatrix}. \quad (5.26)$$

Em seguida é necessário retomar a equação (5.21) e reformular a discretização para o ponto 1, que possui como vizinhos o ponto nodal 2 e a fronteira esquerda,

$$0 = \Gamma_e \frac{\phi_E - \phi_P}{\delta x} - \lim_{\phi_W \rightarrow \phi_w} \lim_{\delta x \rightarrow \delta x/2} \Gamma_w \frac{\phi_P - \phi_W}{\delta x} = \Gamma_e \frac{\phi_E - \phi_P}{\delta x} - \Gamma_w \frac{\phi_P - \phi_w}{\delta x/2}. \quad (5.27)$$

Observe que o elemento de coordenada W à esquerda do ponto 1 não existe, assumimos então que este é igual ao valor na face w e esta está na metade do caminho até o ponto nodal externo. Usando a mesma notação definida em (5.22) temos

$$a_W \phi_w + a_P \phi_P + a_E \phi_E = 0, \quad (5.28)$$

$$\{a_W, a_E, a_P\} = \left\{ \frac{\Gamma_w}{\delta x/2}, \frac{\Gamma_e}{\delta x}, -a_W - a_E \right\}. \quad (5.29)$$

De maneira similar ao efetuado nas equações (5.27)-(5.29), a equação para o ponto nodal 5, pode ser escrita,

$$0 = \lim_{\phi_E \rightarrow \phi_e} \lim_{\delta x \rightarrow \delta x/2} \Gamma_e \frac{\phi_E - \phi_P}{\delta x} - \Gamma_w \frac{\phi_P - \phi_W}{\delta x}, \quad (5.30)$$

$$a_W \phi_W + a_P \phi_P + a_E \phi_e = 0, \quad (5.31)$$

$$\{a_W, a_E, a_P\} = \left\{ \frac{\Gamma_w}{\delta x}, \frac{\Gamma_e}{\delta x/2}, -a_W - a_E \right\}. \quad (5.32)$$

Reunindo estas informações podemos completar o sistema linear,

ponto nodal	Equação
1	$a_{10}\phi_0 + a_{11}\phi_1 + a_{12}\phi_2 = 0,$
5	$a_{54}\phi_4 + a_{55}\phi_5 + a_{56}\phi_6 = 0,$

Tabela 4: Equações discretizadas para os pontos nodais nas fronteiras, esta tabela complementa a tabela (3).

Finalmente o uso da tabela (3) permite completar a equação (5.26),

$$\begin{pmatrix} a_{11} & a_{21} & 0 & 0 & 0 \\ a_1 & a_2 & a_3 & 0 & 0 \\ 0 & a_2 & a_3 & a_4 & 0 \\ 0 & 0 & a_3 & a_4 & a_5 \\ 0 & 0 & 0 & a_5 & a_6 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{pmatrix} = \begin{pmatrix} -a_{10}\phi_0 \\ 0 \\ 0 \\ 0 \\ -a_{56}\phi_6 \end{pmatrix}, \quad (5.33)$$

ou seja,

$$A_{5 \times 5} \phi_{5 \times 1} = C_{5 \times 1}, \quad (5.34)$$

onde

$$A_{5 \times 5} = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{pmatrix}, \quad \phi_{5 \times 1} = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{pmatrix}, \quad c_{5 \times 1} = \begin{pmatrix} -a_{10}\phi_0 \\ 0 \\ 0 \\ 0 \\ -a_{56}\phi_6 \end{pmatrix}. \quad (5.35)$$

Para o ponto nodal 1, basta observar a equação (5.29) e converter da notação de bússola para notação matricial,

$$a_{10} = \frac{\Gamma_1}{\delta x/2}, \quad a_{12} = \frac{\Gamma_2}{\delta x}, \quad a_{11} = -a_{10} - a_{12}. \quad (5.36)$$

Para os pontos nodais internos, de 2 até 4, vem

$$a_{i,j} = \frac{\Gamma_i}{\delta x}, \quad \forall j = i - 1, \quad a_{i,j} = \frac{\Gamma_{i+1}}{\delta x}, \quad \forall j = i + 1, \quad a_{ij} = -a_{i,i-1} - a_{i,i+1}, \quad \forall j = i. \quad (5.37)$$

Finalmente, para o ponto nodal 5 os coeficientes são

$$a_{54} = \frac{\Gamma_5}{\delta x}, \quad a_{56} = \frac{\Gamma_6}{\delta x/2}, \quad a_{55} = -a_{54} - a_{56}. \quad (5.38)$$

Para concluir, este problema pode ser resolvido lembrando que os dados para este exemplo são

$$\{\Gamma_i = k = \Gamma_{i+1}, \phi_0 = T_A, \phi_6 = T_B\} = \left\{ 100 \frac{W}{mK}, 100K, 500K \right\}. \quad (5.39)$$

Para calcular o δx é necessário contar o número de pontos nodais e como a distância entre cada dois pontos nodais é δx , restará ainda a distância do primeiro ponto nodal até a borda inicial $\delta x/2$ e também do último ponto nodal até a borda final $\delta x/2$, ou seja, para n pontos nodais, teremos $(n - 1)\delta x + \delta x/2 + \delta x/2$, logo, $l = n\delta x$ onde l é o tamanho do domínio, e portanto $\delta x = l/n$.

Para este problema a matriz $A_{5 \times 5}$ assume a forma seguinte,

$$A_{5 \times 5} = 1000 \begin{pmatrix} -3 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -3 \end{pmatrix}, \quad (5.40)$$

enquanto a matriz C que está envolvidas com as constantes vale

$$C_{5 \times 1} = 1000 \begin{pmatrix} -200 \\ 0 \\ 0 \\ 0 \\ -1000 \end{pmatrix}. \quad (5.41)$$

Finalmente, basta resolver o sistema linear abaixo,

$$1000 \underbrace{\begin{pmatrix} -3 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -3 \end{pmatrix}}_{A_{5 \times 5}} \underbrace{\begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{pmatrix}}_{C_{5 \times 1}} = 1000 \underbrace{\begin{pmatrix} -200 \\ 0 \\ 0 \\ 0 \\ -1000 \end{pmatrix}}_{C_{5 \times 1}}; \quad (5.42)$$

de onde é possível obter as temperaturas nos pontos nodais,

$$\begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{pmatrix} = \begin{pmatrix} 140 \\ 220 \\ 300 \\ 380 \\ 460 \end{pmatrix}. \quad (5.43)$$

5.3.2 Implementação Computacional da Difusão no Estado Estacionário

Para uma quantidade pequena de pontos nodais é possível resolver o sistema de diferentes formas, mas quando uma grande quantidade de pontos nodais é utilizada, torna-se indispensável a implementação computacional do sistema. Na ocasião a implementação em R mostrou-se bastante eficaz, inclusive pela disponibilidade de pacotes para inversão de matrizes esparsas em poucos segundos.

Inicialmente, é necessário definir as constantes e o grid. Portanto, é necessário escrever no código R as seguintes linhas:

Trecho de código R: início

```
l=0.5; x0=0; xnf=0.5; n=100; interval=1; phi0=100
```

```
phin=500; Gamma=100; nf=n+1; delta.x.n=l/n; delta.x.f=delta.x.n/2
```

```
first.node=x0+delta.x.f; last.node=first.node+(n-1)*delta.x.n
```

```
grid1D<-seq(first.node,last.node,delta.x.n)
```

Trecho de código R: fim

Onde: “l” foi escolhido para representar o tamanho do domínio em 1D, “x0” é a posição inicial do grid, “xn” é a posição final do grid, o número de faces é representado por “n”,

a variável “interval” define o intervalo no qual o resultado deve ser gravado no arquivo de saída, “phi0” é o valor de ϕ na face do início do domínio, “phin” é o valor de ϕ na face do final do domínio, “Gamma” é o valor de Γ , “nf” é a quantidade de faces, “delta.x.n” é a distância entre pontos nodais consecutivos e “delta.x.f” é a distância entre uma face e um ponto nodal adjacente, “first.node” e “last.node” definem as posições do primeiro e último pontos nodais respectivamente e finalmente a variável “grid1D” guarda os valores das localizações dos pontos nodais.

Em seguida a matriz que guarda os valores de Γ nas faces é criada abaixo, lembrando que para o problema acima Γ é constante, ou seja, terá o mesmo valor em todas as faces, como segue

Trecho de código R: início

```
Gamma.x=matrix(data=0,nrow=1,ncol=nf)
for(j in 1:nf){Gamma.x[j]<-100}
```

Trecho de código R: fim.

A matriz A possui coeficientes associados aos valores dos pontos nodais e os valores devido ao contorno do domínio são definidos de maneira suplementar, como segue

Trecho de código R: início

```
A10=Gamma.x[1]/delta.x.f
A1nf=Gamma.x[nf]/delta.x.f
```

Trecho de código R: fim.

A matriz A é uma matriz tri-diagonal, portanto é esparsa. Para otimizar a solução do sistema linear é necessário carregar um pacote que habilite a solução de sistemas com matrizes esparsas, na ocasião o pacote Matrix,

Trecho de código R: início

```
library(Matrix)
```

Trecho de código R: fim.

Em seguida a matriz A é criada como uma matriz nula n por n, e seus valores são guardados conforme o loop à seguir

Trecho de código R: início

```

A=matrix(data=0,nrow=n,ncol=n)
for(i in 1:n){ for(j in 1:n){
if(i==j){
if(i==1){ A[i,j+1]=Gamma.x[j]/delta.x.n; A[i,j]=-A10 -A[i,j+1]}
if(i>1 & i<n){ A[i,j-1]=Gamma.x[j]/delta.x.n; A[i,j+1]=Gamma.x[j+1]/delta.x.n
A[i,j]=-A[i,j-1]-A[i,j+1] }
if(i==n){ A[i,j-1]=Gamma.x[j]/delta.x.n; A[i,j]=-A[i,j-1]-A1nf } } } }

```

Trecho de código R: fim.

Agora que a matriz A está criada, esta pode ser convertida para esparsa e em seguida vamos verificar a economia de espaço nas diferentes representações, esparsa ou densa, como segue,

Trecho de código R: início

```

Asparse<-as(A,"sparseMatrix")
object.size(A)
object.size(Asparse)

```

Trecho de código R: fim.

Na ocasião de uma matriz 20 por 20 a diferença fica pequena, 3,4kbytes na representação densa contra 2,2kbytes na representação esparsa, mas esta diferença cresce quando o tamanho da matriz A cresce, e para uma matriz 1000 por 1000 chega a 8000kbytes contra 41kbytes respectivamente nas representações densa e esparsa. Na figura (31)-superior é possível ver a ocupação da matriz, enquanto na (31)-inferior é exibido o custo de memória para as representações esparsa e densa da matriz A.

Em seguida a matriz das constantes C é criada como uma matriz nula de n linhas por 1 coluna, posteriormente são inseridos os valores inicial e final que correspondem às condições de contorno, enquanto os demais valores serão nulos, ou seja,

Trecho de código R: início

```

C=matrix(data=0,nrow=n,ncol=1)
for(i in 1:n){
if(i==1){ C[i]=-A10*phi0}

```

```
if(i==n){C[i]=-A1nf*phinf} }
```

Trecho de código R: fim.

Finalmente, o sistema pode ser resolvido de duas maneiras: a primeira delas usando os métodos do pacote base para resolver diretamente o sistema com as rotinas tipo LAPACK; a segunda maneira usa o fato da matriz A ser esparsa e portanto é mais facilmente resolvida pelo pacote Matrix do R, que inclui rotinas do tipo LAPACK e SUIESPARSE. A coleção de rotinas SUIESPARSE é ampla e inclui a rotina UMFPACK cuja eficiência na solução de sistemas lineares envolvendo matrizes esparsas é bastante conhecida. Na figura (32)-inferior é possível notar uma grande diferença no tempo gasto para resolver o sistema linear com as diferentes rotinas, já na figura (32)-superior podemos notar que a solução numérica está de acordo com a solução analítica.

Trecho de código R: início

```
system.time(( phi=solve(A,C) ))
```

```
system.time(( phi2=try(solve(Asparse, C, sparse=TRUE,tol = 1e-19)) ))
```

Trecho de código R: fim.

A solução analítica para o problema em questão é dada pela função abaixo

$$\begin{aligned} \phi(x) &= \frac{1}{\Gamma} (C_1 x + C_2), \quad C_1 \equiv \Gamma \left(\frac{\phi_n - \phi_0}{x_n - x_0} \right), \\ \text{e } C_2 &\equiv \Gamma \left(\frac{\phi_n + \phi_0}{2} \right) - C_1 \left(\frac{x_n + x_0}{2} \right), \end{aligned} \quad (5.44)$$

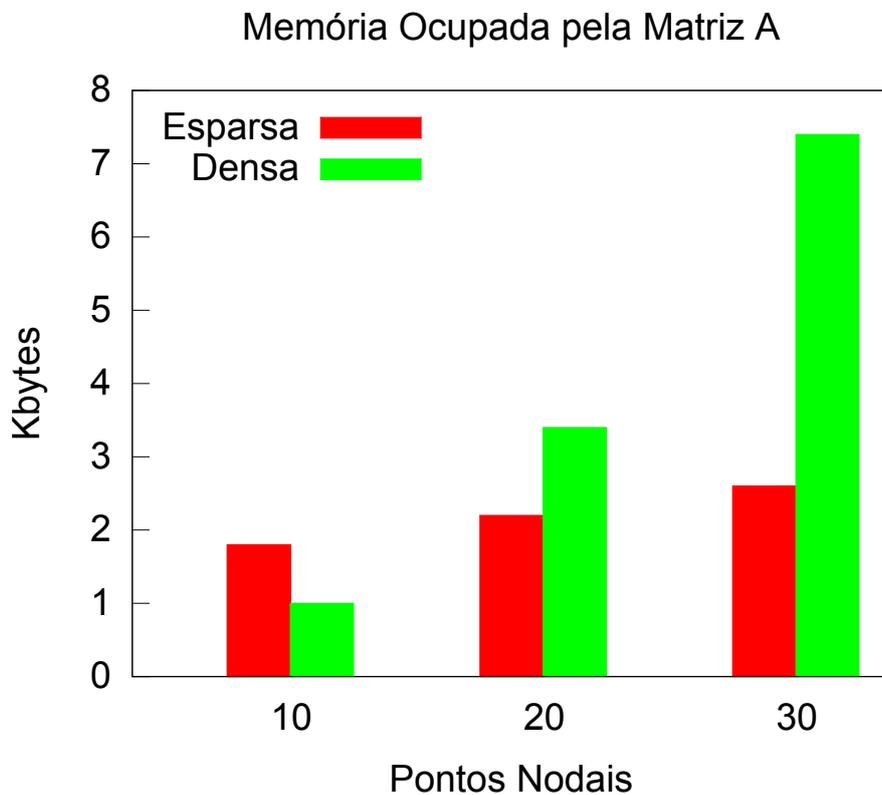
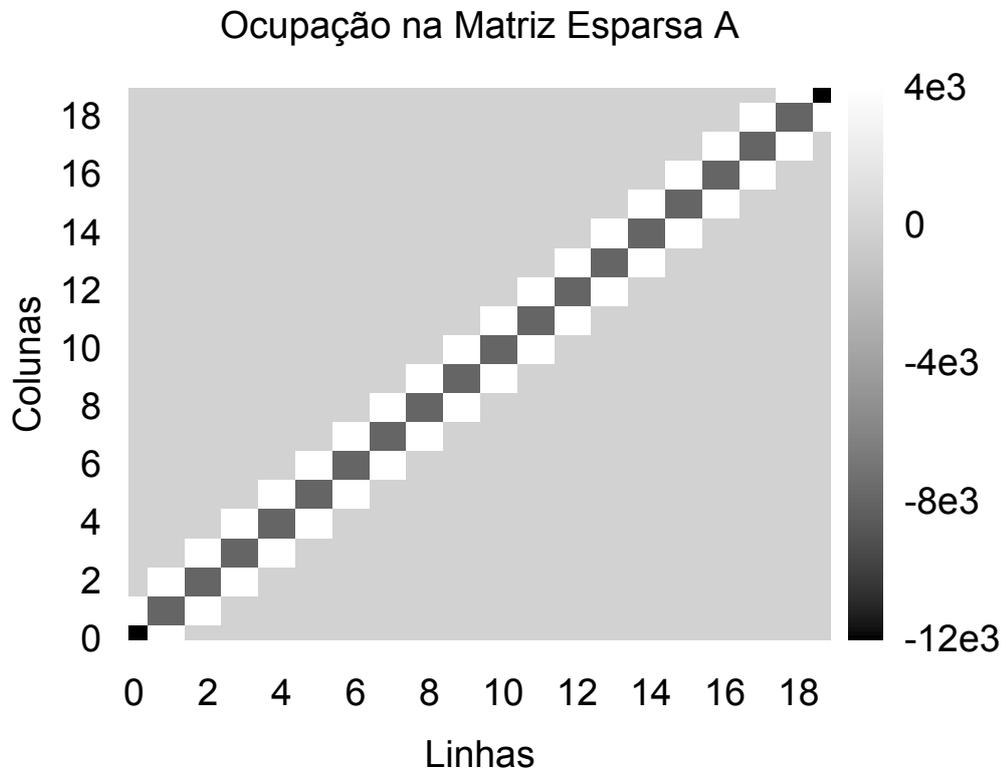


Figura 31: Superior: representação da matriz esparsa A para uma discretização com 20 pontos nodais; inferior: ocupação da memória pela matriz A nas formas densa e esparsa.

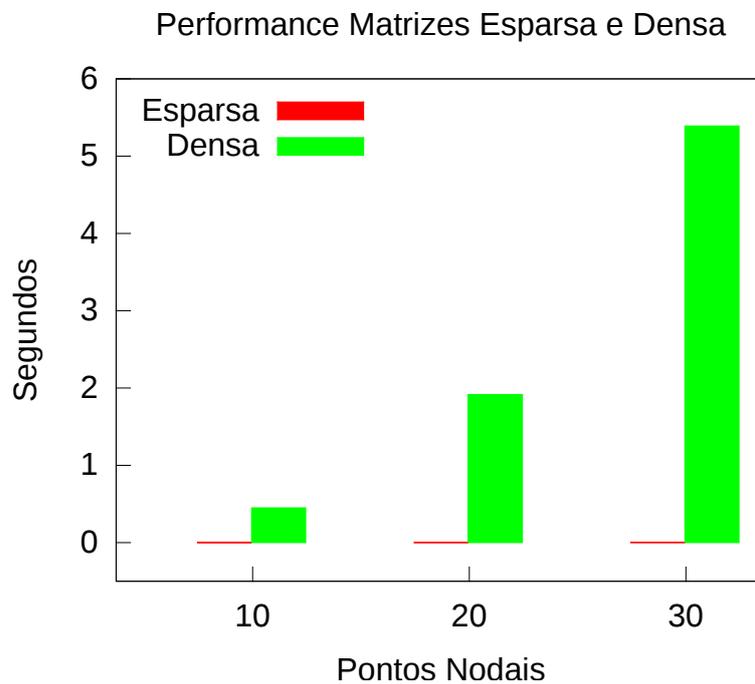
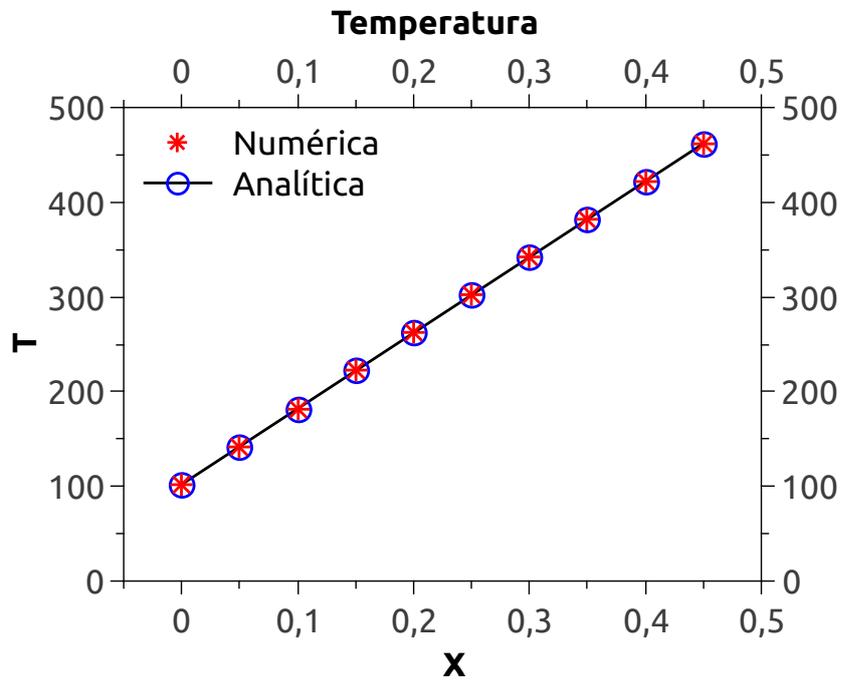


Figura 32: Superior: Solução do exemplo proposto usando 100 pontos nodais com intervalo de gravação de 10 em 10 pontos. Inferior: aumento do custo computacional quando a solução é obtida usando método para matrizes esparsas, contra a solução obtida usando método para matrizes densas.

5.3.3 Difusão com o Termo Transiente

Para a difusão com termo transiente vamos estudar a condução do calor através de uma barra submetida à condições iniciais, como segue

$$\frac{\partial T}{\partial t} = \vec{\nabla} \cdot (k \vec{\nabla} T), \quad \begin{cases} T(x, 0) = \sin(\pi x), \\ T(0, t) = 0, \quad T(1, t) = 0. \end{cases} \quad (5.45)$$

Adicionalmente a constante k está definida como $k = 1$. Para este caso a solução analítica é dada pela função,

$$T(x, t) = \sin(\pi x) e^{-\pi^2 t}. \quad (5.46)$$

É possível notar a semelhança da equação (5.45) com a seguinte forma da equação do transporte,

$$\frac{\partial \phi}{\partial t} + \underbrace{\vec{\nabla} \cdot (\phi \vec{u})}_0 = \vec{\nabla} \cdot \left(\frac{\Gamma}{\rho} \vec{\nabla} \phi \right) + \underbrace{\frac{S}{\rho}}_0, \quad (5.47)$$

desde que sejam declaradas as relações de equivalência entre os valores: $\Gamma/\rho \equiv \Gamma \equiv k$, $\phi \equiv T$. A discretização de (5.47) passa pelo procedimento de integração,

$$\int_t^{t+\Delta t} \int_V \frac{\partial \phi}{\partial t} dV dt = \int_t^{t+\Delta t} \int_V \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi) dV dt. \quad (5.48)$$

Comutando as integrais dentro das possibilidades em (5.48) vem

$$\left[\int_V dV \right] \int_t^{t+\Delta t} \frac{\partial \phi}{\partial t} dt = \left[\int_t^{t+\Delta t} dt \right] \int_V \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi) dV, \quad (5.49)$$

ou simplesmente,

$$\underbrace{\int_t^{t+\Delta t} \frac{\partial \phi}{\partial t} dt}_I = \frac{\left[\int_t^{t+\Delta t} dt \right]}{\left[\int_V dV \right]} \underbrace{\int_V \nabla \cdot (\Gamma \nabla \phi) dV}_{II}. \quad (5.50)$$

Em seguida é necessário proceder com a discretização dos termos I e II da equação (5.50). O termo I será discretizado no esquema conhecido como **explicit forward Euler**, o termo II terá a discretização reaproveitada da difusão no estado estacionário, em

conformidade com a equação (5.19), como segue,

$$\underbrace{\int_t^{t+\Delta t} \frac{\partial \phi}{\partial t} dt}_I \simeq \phi_P^{t+1} - \phi_P^t,$$

$$\underbrace{\int_V \nabla \cdot (\Gamma_\phi \nabla \phi) dV}_II \simeq A_e \Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - A_w \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x}. \quad (5.51)$$

Agora que a parte temporal aparece devido ao termo transiente, é necessário introduzir os índices t e $t + 1$, onde t refere-se ao instante de tempo atual enquanto $t + 1$ refere-se ao instante de tempo imediatamente posterior.

Considerando os coeficientes de (5.50), $[\int_t^{t+\Delta t} dt] = \delta t$ e $[\int_V dV] = \delta V = A\delta x$ e a regularidade da discretização com $A_e = A_w = A$ e reunindo (5.50) e (5.51), é possível obter

$$\phi_P^{t+1} - \phi_P^t \simeq \frac{A\delta t}{A\delta x} \left[\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right], \quad (5.52)$$

$$\phi_P^{t+1} \simeq \phi_P^t + \frac{\delta t}{\delta x} \left[\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right]. \quad (5.53)$$

Neste esquema a estabilidade dependerá do coeficiente

$$D = \frac{\Gamma \delta t}{\delta x^2}. \quad (5.54)$$

A condição de estabilidade é dada por $\frac{\Gamma \delta t}{\delta x^2} < 1$ [27]. Para construir a equação matricial do esquema dado em (5.53), basta notar que: para os pontos nodais 1 e n , os limites nas faces devem ser assumidos em conformidade com o que foi efetuado nas equações (5.27) e (5.30), enquanto para os pontos nodais de 2 até $n-1$ a equação (5.53) pode ser utilizada, conforme segue

- Ponto nodal 1

$$\phi_1^{t+1} \simeq \phi_1^t + \frac{\delta t}{\delta x} \left[\Gamma_e \frac{\phi_E^t - \phi_1^t}{\delta x} - \lim_{\phi_W^t \rightarrow \phi_1^t} \lim_{\delta x \rightarrow \delta x/2} \Gamma_w \frac{\phi_1^t - \phi_W^t}{\delta x} \right]. \quad (5.55)$$

- Pontos nodais de 2 até $n-1$

$$\phi_P^{t+1} \simeq \phi_P^t + \frac{\delta t}{\delta x} \left[\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right], \quad P = \{2, \dots, n-1\}. \quad (5.56)$$

- Ponto nodal n

$$\phi_n^{t+1} \simeq \phi_n^t + \frac{\delta t}{\delta x} \left[\lim_{\phi_E^t \rightarrow \phi_e^t} \lim_{\delta x \rightarrow \delta x/2} \Gamma_e \frac{\phi_E^t - \phi_n^t}{\delta x} - \Gamma_w \frac{\phi_n^t - \phi_W^t}{\delta x} \right]. \quad (5.57)$$

As equações de (5.55)-(5.57) permitem escrever as equações para 4 pontos nodais (como exemplo), no seguinte formato,

$$\phi_1^{t+1} \simeq \phi_1^t + \frac{\delta t}{\delta x} \left[\underbrace{\left(-\frac{\Gamma_2}{\delta x} - \frac{\Gamma_1}{\delta x/2} \right)}_{a_{11}} \phi_1^t + \underbrace{\left(\frac{\Gamma_2}{\delta x} \right)}_{a_{12}} \phi_2^t \right] + \frac{\delta t}{\delta x} \left[\underbrace{\left(\frac{\Gamma_1}{\delta x/2} \right)}_{a_{10}} \phi_0^t \right], \quad (5.58)$$

$$\phi_2^{t+1} \simeq \phi_2^t + \frac{\delta t}{\delta x} \left[\underbrace{\left(\frac{\Gamma_2}{\delta x} \right)}_{a_{21}} \phi_1^t + \underbrace{\left(-\frac{\Gamma_3}{\delta x} - \frac{\Gamma_2}{\delta x} \right)}_{a_{22}} \phi_2^t + \underbrace{\left(\frac{\Gamma_3}{\delta x} \right)}_{a_{23}} \phi_3^t \right] + \frac{\delta t}{\delta x} [0], \quad (5.59)$$

$$\phi_3^{t+1} \simeq \phi_3^t + \frac{\delta t}{\delta x} \left[\underbrace{\left(\frac{\Gamma_3}{\delta x} \right)}_{a_{31}} \phi_2^t + \underbrace{\left(-\frac{\Gamma_4}{\delta x} - \frac{\Gamma_3}{\delta x} \right)}_{a_{32}} \phi_3^t + \underbrace{\left(\frac{\Gamma_4}{\delta x} \right)}_{a_{33}} \phi_4^t \right] + \frac{\delta t}{\delta x} [0], \quad (5.60)$$

$$\phi_n^{t+1} \simeq \phi_n^t + \frac{\delta t}{\delta x} \left[\underbrace{\frac{\Gamma_{n-1}}{\delta x}}_{a_{n,n-1}} \phi_{n-1}^t + \underbrace{\left(-\frac{\Gamma_{n+1}}{\delta x/2} - \frac{\Gamma_n}{\delta x} \right)}_{a_{n,n}} \phi_n^t \right] + \frac{\delta t}{\delta x} \left[\underbrace{\frac{\Gamma_{n+1}}{\delta x/2}}_{a_{n,n+1}} \phi_{n+1}^t \right]. \quad (5.61)$$

A equação (5.61) está montada de maneira genérica para um n-ésimo termo, mas para efeito de ilustração, podemos escolher $n = 4$ e obter o seguinte sistema matricial,

$$\begin{pmatrix} \phi_1^{t+1} \\ \phi_2^{t+1} \\ \phi_3^{t+1} \\ \phi_4^{t+1} \end{pmatrix} = \frac{\delta t}{\delta x} \begin{pmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} \phi_1^t \\ \phi_2^t \\ \phi_3^t \\ \phi_4^t \end{pmatrix} + \frac{\delta t}{\delta x} \begin{pmatrix} a_{10} \phi_0^t \\ 0 \\ 0 \\ a_{45} \phi_5^t \end{pmatrix}, \quad (5.62)$$

ou seja, para n pontos nodais o sistema assume a forma seguinte

$$\phi_{n \times 1}^{t+1} = \frac{\delta t}{\delta x} A_{n \times n} \phi_{n \times 1}^t + \frac{\delta t}{\delta x} C_{n \times 1}, \quad (5.63)$$

onde os elementos da matriz A são dados por:

$$\begin{aligned}
 j &= i - 1 \rightarrow a_{i,i-1} = \frac{\Gamma_i}{\delta x}, \\
 j &= i + 1 \rightarrow a_{i,i+1} = \frac{\Gamma_{i+1}}{\delta x}, \\
 j &= i \rightarrow a_{ii} = -a_{i,i-1} - a_{i,i+1}.
 \end{aligned}
 \tag{5.64}$$

5.3.4 Implementação Computacional da Difusão com Termo Transiente

O esquema do tipo **explicit forward Euler** é bastante limitado para resolver a equação da difusão (5.45), pois para haver estabilidade necessitamos que $\delta t < \delta x^2/\Gamma$. Com as condições do problema posto e com o esquema **explicit forward Euler** só é possível realizar simulações para pequenos intervalos de tempo, ficando claro a fragilidade deste tipo de discretização para um intervalo de tempo maior.

Para implementar em R a difusão com termo transiente grande parte do código anterior da difusão no estado estacionário foi aproveitado; primeiramente foram definidas as condições temporais e a sequência de intervalos de tempo. A parte espacial utiliza a mesma estrutura do código anterior. Logo

Trecho de código R: início

```

t0=0
tn=0.02
n=20
delta.t=(tn-t0)/(n-1)
time=matrix(data=seq(t0,tn,delta.t),nrow=n,ncol=1)

```

Trecho de código R: fim.

Outra novidade com relação ao código anterior é a introdução do campo inicial de temperatura que deverá evoluir segundo o esquema proposto por (5.63), neste caso,

Trecho de código R: início

```

phi.initial=matrix(data=0,ncol=1,nrow=n)
for(j in 1:n){ phi.initial[j]=sin((pi)*grid1D[j])}

```

Trecho de código R: fim.

Nenhuma mudança precisou ser efetuada na montagem da matriz A, exceto o fato de que não haverá nesse caso vantagem em armazená-la como esparsa, isto apenas aumentaria o tempo de acesso, pois nenhum algoritmo de inversão será necessário. Fica a possibilidade de comentar o pacote Matrix. A matriz C antes ficava do lado oposto ao sinal de igualdade com relação à matriz A, agora ficam ambas do mesmo lado; conforme podemos notar das equações (5.63) e (5.34), bastando modificar o sinal da matriz C, ou seja

Trecho de código R: início

```
for(i in 1:n){
  if(i==1){C[i]=A10*phi0}
  if(i==n){C[i]=A1nf*phin} }
```

Trecho de código R: fim.

Para não ocupar muito espaço na memória, a gravação em arquivo é recomendada, pois para cada instante de tempo posterior será gerado um novo campo de temperaturas; por isso um loop é gerado e o novo campo é calculado a partir do antigo, em seguida o novo campo é gravado e depois declarado como antigo para que ao rodar novamente o loop, este último campo gere o próximo, então

Trecho de código R: início

```
phi.old=phi.initial
coef=delta.t/delta.x.n
write.table( format(phi.old), file=paste("phi",1,sep=""),
  sep="\t", col.names=F, row.names=F )
for(t in 2:n){
  phi.new=phi.old+(coef)*A%*%phi.old+(coef)*C
  phi.old<-phi.new
  write.table( format(phi.old), file=paste("phi",t,sep=""),
  sep="\t", col.names=F, row.names=F ) }
```

Trecho de código R: fim.

A seguir nas figuras abaixo é possível observar o desempenho do esquema **explicit forward Euler** para o problema de difusão com termo transiente segundo as condições

estabelecidas pelo problema posto. Este resultado será útil para comparações futuras.

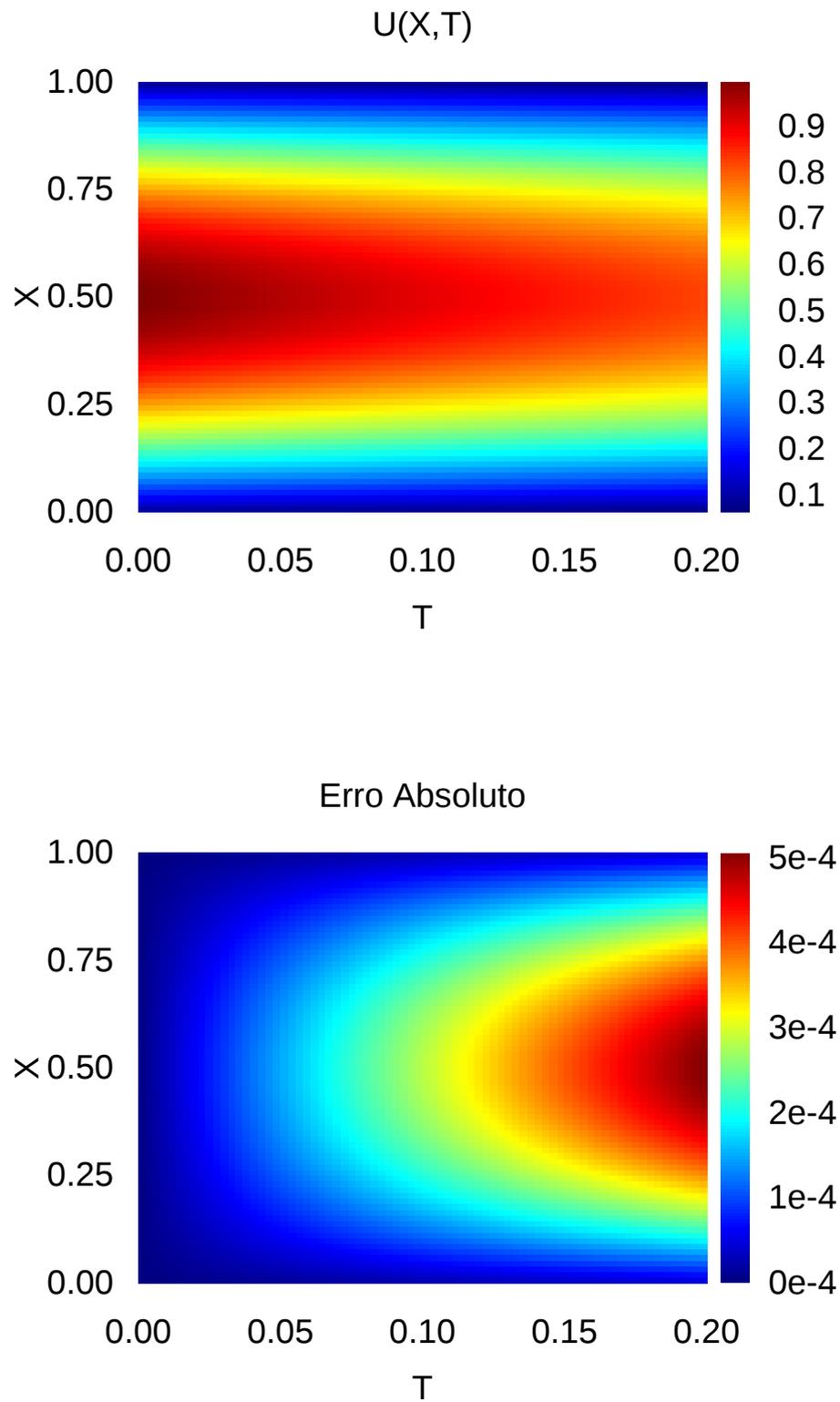


Figura 33: Superior: Solução numérica da equação do calor em 1D usando 20 pontos nodais. Inferior: erro absoluto obtido por comparação entre a solução numérica e a analítica.

Uma outra possibilidade é discretizar o problema através do esquema **implicit backward Euler**. Para este mesmo problema proposto pelas equações (5.45) e (5.46) a discretização também envolve os procedimentos efetuados de (5.48)-(5.52), mas agora é assumido que o lado direito é avaliado no tempo $t + \delta t$ no termo da difusão, é neste limite que o esquema recebe o nome de **implicit backward Euler**, ou seja, ao invés de

$$\phi_P^{t+1} - \phi_P^t \simeq \frac{A\delta t}{A\delta x} \left[\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right], \quad (5.65)$$

a equação (5.52) é modificada para, [23],

$$\phi_P^{t+1} - \phi_P^t \simeq \frac{A\delta t}{A\delta x} \left[\Gamma_e \frac{\phi_E^{t+1} - \phi_P^{t+1}}{\delta x} - \Gamma_w \frac{\phi_P^{t+1} - \phi_W^{t+1}}{\delta x} \right]. \quad (5.66)$$

Agora basta isolar de (5.66) os termos com índice em função dos demais,

$$\frac{\delta x}{\delta t} \phi_P^{t+1} - \Gamma_e \frac{\phi_E^{t+1} - \phi_P^{t+1}}{\delta x} + \Gamma_w \frac{\phi_P^{t+1} - \phi_W^{t+1}}{\delta x} \simeq \frac{\delta x}{\delta t} \phi_P^t. \quad (5.67)$$

O próximo passo consiste em agrupar os termos de acordo com as coordenadas dos pontos nodais e em seguida definir os coeficientes,

$$\underbrace{\left(-\frac{\Gamma_w}{\delta x} \right)}_{a_W} \phi_W^{t+1} + \underbrace{\left(\frac{\Gamma_w}{\delta x} + \frac{\delta x}{\delta t} + \frac{\Gamma_e}{\delta x} \right)}_{a_P} \phi_P^{t+1} + \underbrace{\left(-\frac{\Gamma_e}{\delta x} \right)}_{a_E} \phi_E^{t+1} \simeq \frac{\delta x}{\delta t} \phi_P^t. \quad (5.68)$$

A seguir é necessário construir a forma matricial da equação discretizada (5.68), não esquecendo de assumir os limites nas fronteiras para o primeiro e último pontos nodais. Como exemplo, vamos supor um domínio com quatro pontos nodais,

- ponto nodal 1:

$$\underbrace{\left(-\lim_{\delta x \rightarrow \delta x/2} \frac{\Gamma_w}{\delta x} \right)}_{a_W} \lim_{W \rightarrow w} \phi_W^{t+1} + \underbrace{\left(\lim_{\delta x \rightarrow \delta x/2} \frac{\Gamma_w}{\delta x} + \frac{\delta x}{\delta t} + \frac{\Gamma_e}{\delta x} \right)}_{a_P} \phi_P^{t+1} + \underbrace{\left(-\frac{\Gamma_e}{\delta x} \right)}_{a_E} \phi_E^{t+1} \simeq \frac{\delta x}{\delta t} \phi_P^t, \quad (5.69)$$

$$\underbrace{\left(-\frac{\Gamma_1}{\delta x/2} \right)}_{a_{10}} \phi_0^{t+1} + \underbrace{\left(\frac{\Gamma_1}{\delta x/2} + \frac{\delta x}{\delta t} + \frac{\Gamma_2}{\delta x} \right)}_{a_{11}} \phi_1^{t+1} + \underbrace{\left(-\frac{\Gamma_2}{\delta x} \right)}_{a_{12}} \phi_2^{t+1} \simeq \frac{\delta x}{\delta t} \phi_1^t, \quad (5.70)$$

- ponto nodal 2:

$$\underbrace{\left(-\frac{\Gamma_2}{\delta x}\right)}_{a_{21}}\phi_1^{t+1} + \underbrace{\left(\frac{\Gamma_2}{\delta x} + \frac{\delta x}{\delta t} + \frac{\Gamma_3}{\delta x}\right)}_{a_{22}}\phi_2^{t+1} + \underbrace{\left(-\frac{\Gamma_3}{\delta x}\right)}_{a_{23}}\phi_3^{t+1} \simeq \frac{\delta x}{\delta t}\phi_2^t, \quad (5.71)$$

• ponto nodal 3:

$$\underbrace{\left(-\frac{\Gamma_3}{\delta x}\right)}_{a_{32}}\phi_2^{t+1} + \underbrace{\left(\frac{\Gamma_3}{\delta x} + \frac{\delta x}{\delta t} + \frac{\Gamma_4}{\delta x}\right)}_{a_{33}}\phi_3^{t+1} + \underbrace{\left(-\frac{\Gamma_4}{\delta x}\right)}_{a_{34}}\phi_4^{t+1} \simeq \frac{\delta x}{\delta t}\phi_3^t, \quad (5.72)$$

• ponto nodal 4:

$$\underbrace{\left(-\frac{\Gamma_w}{\delta x}\right)}_{a_W}\phi_W^{t+1} + \underbrace{\left(\frac{\Gamma_w}{\delta x} + \frac{\delta x}{\delta t} + \lim_{\delta x \rightarrow \delta x/2} \frac{\Gamma_e}{\delta x}\right)}_{a_P}\phi_P^{t+1} + \underbrace{\left(-\lim_{\delta x \rightarrow \delta x/2} \frac{\Gamma_e}{\delta x}\right)}_{a_E}\lim_{E \rightarrow e}\phi_E^{t+1} \simeq \frac{\delta x}{\delta t}\phi_P^t, \quad (5.73)$$

$$\underbrace{\left(-\frac{\Gamma_4}{\delta x}\right)}_{a_{43}}\phi_3^{t+1} + \underbrace{\left(\frac{\Gamma_4}{\delta x} + \frac{\delta x}{\delta t} + \frac{\Gamma_5}{\delta x/2}\right)}_{a_{44}}\phi_4^{t+1} + \underbrace{\left(-\frac{\Gamma_5}{\delta x/2}\right)}_{a_{45}}\phi_5^{t+1} \simeq \frac{\delta x}{\delta t}\phi_4^t. \quad (5.74)$$

Ao observar as equações de (5.69)-(5.74) nota-se que estas podem ser reunidas no seguinte sistema linear,

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} \phi_1^{t+1} \\ \phi_2^{t+1} \\ \phi_3^{t+1} \\ \phi_4^{t+1} \end{pmatrix} \simeq \frac{\delta x}{\delta t} \begin{pmatrix} \phi_1^t \\ \phi_2^t \\ \phi_3^t \\ \phi_4^t \end{pmatrix} - \begin{pmatrix} a_{10}\phi_0^{t+1} \\ 0 \\ 0 \\ a_{45}\phi_5^{t+1} \end{pmatrix}, \quad (5.75)$$

lembrando que os valores ϕ_0^{t+1} e ϕ_5^{t+1} são valores nas faces das fronteiras do domínio e, conforme as condições de contorno colocadas pela equação (5.45), são constantes nestas posições para todos os instantes de tempo, ou seja, é possível substituir em (5.75) os valores,

$$\begin{aligned} a_{10}\phi_0^{t+1} &\rightarrow a_{10}\phi_0^t, \\ a_{45}\phi_5^{t+1} &\rightarrow a_{45}\phi_5^t. \end{aligned} \quad (5.76)$$

Trocando (5.76) em (5.75), obtemos em sua forma final o seguinte sistema linear,

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} \phi_1^{t+1} \\ \phi_2^{t+1} \\ \phi_3^{t+1} \\ \phi_4^{t+1} \end{pmatrix} \simeq \frac{\delta x}{\delta t} \begin{pmatrix} \phi_1^t \\ \phi_2^t \\ \phi_3^t \\ \phi_4^t \end{pmatrix} - \begin{pmatrix} a_{10}\phi_0^t \\ 0 \\ 0 \\ a_{45}\phi_5^t \end{pmatrix}. \quad (5.77)$$

Em (5.77) é possível notar que o sistema pode ser resolvido de forma similar ao sistema estudado anteriormente no caso da difusão para o estado estacionário; neste caso também será vantajoso escrever a matriz A como uma matriz esparsa e usar os pacotes de rotinas do pacote Matrix do R. Em linhas gerais, será necessário programar o sistema para a equação matricial

$$A_{n \times n} \phi_{n \times 1}^{t+1} \simeq \frac{\delta x}{\delta t} \phi_{n \times 1}^t - C_{n \times 1}, \quad (5.78)$$

cuja solução será

$$\phi_{n \times 1}^{t+1} \simeq \frac{\delta x}{\delta t} A_{n \times n}^{-1} \phi_{n \times 1}^t - A_{n \times n}^{-1} C_{n \times 1}. \quad (5.79)$$

A matriz A é montada da seguinte forma,

$$\begin{aligned} j &= i - 1 \rightarrow a_{i,i-1} = -\frac{\Gamma_i}{\delta x}, \\ j &= i + 1 \rightarrow a_{i,i+1} = -\frac{\Gamma_{i+1}}{\delta x}, \\ j &= i \rightarrow a_{ii} = -a_{i,i-1} + \frac{\delta x}{\delta t} - a_{i,i+1}. \end{aligned} \quad (5.80)$$

A implementação do código para o esquema **implicit backward Euler** aproveita os códigos anteriores com poucas alterações. Podemos destacar a montagem das matrizes A e C: a escrita da matriz A agora será na forma esparsa e esta implementação permitirá realizar simulações que o código anterior não permitia, pois o método implícito é estável. Na simulação apresentada a seguir, o domínio foi aumentado e também o intervalo de tempo, sem comprometer a solução. O código anterior não rodaria $l = 4$ e $t = 0.5$ para uma quantidade de 500 pontos nodais, caso testado com sucesso no esquema implícito. Seguem as mudanças com relação ao código anterior, primeiramente para os coeficientes externos da matriz A:

Trecho de código R: início

```
A10=-Gamma.x[1]/delta.x.f
```

```
A1nf=-Gamma.x[nf]/delta.x.f
```

Trecho de código R: fim.

A matriz A neste caso é montada com o seguinte código

Trecho de código R: início

```
A=matrix(data=0,nrow=n,ncol=n)
```

```
for(i in 1:n){ for(j in 1:n){ if(i==j){
```

```

if(i==1){
A[i,j+1]=-Gamma.x[j+1]/delta.x.n
A[i,j]=-A10+delta.x.n/delta.t-A[i,j+1]}#if(i==1)
if(i>1&i<n){
A[i,j-1]=-Gamma.x[j]/delta.x.n
A[i,j+1]=-Gamma.x[j+1]/delta.x.n
A[i,j]=-A[i,j-1]+delta.x.n/delta.t-A[i,j+1]}#if(i>2&i<n)
if(i==n){
A[i,j-1]=-Gamma.x[j]/delta.x.n
A[i,j]=-A[i,j-1]+delta.x.n/delta.t-A1nf } } } }
Asparse<-as(A,"sparseMatrix")

```

Trecho de código R: fim.

A matriz C é montada como segue

Trecho de código R: início

```

C=matrix(data=0,nrow=n,ncol=1)
for(i in 1:n){
if(i==1){
C[i]=A10*phi0 }#if(i==n)
if(i==n){ C[i]=A1nf*phinf }#if(i==n) }

```

Trecho de código R: fim.

Por último o sistema linear é resolvido através do seguinte loop

Trecho de código R: início

```

for(t in 2:n){
D=(delta.x.n/delta.t)*phi.old-C
phi.new=try(solve(Asparse, D, sparse=TRUE,tol = 1e-19) )
phi.old<-phi.new

```

```
write.matrix( phi.old, file=paste("phi",t,sep=""), )}
```

Trecho de código R: fim.

A solução segue na figura (34),

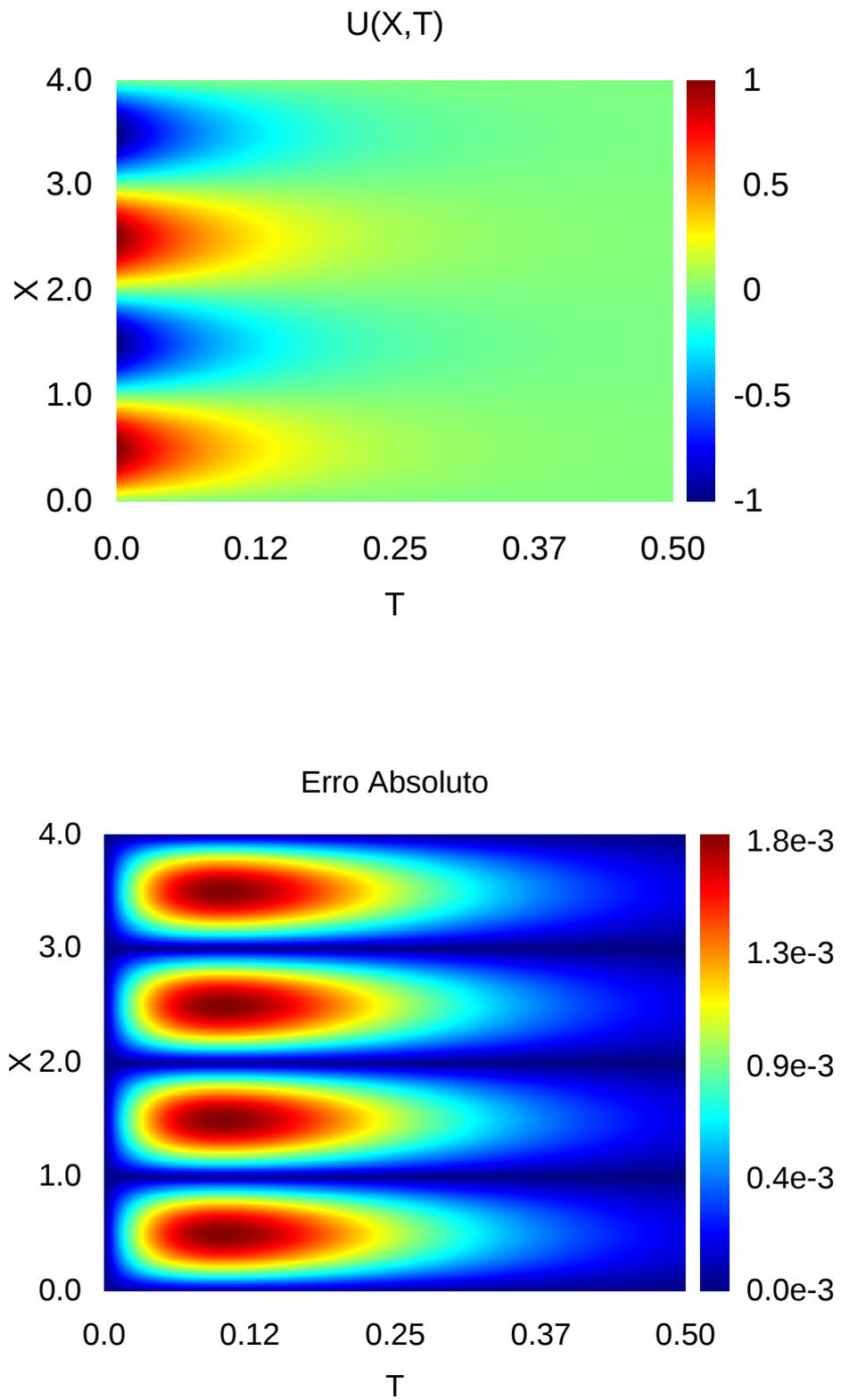


Figura 34: Esquerda: Solução numérica da equação do calor usando 500 pontos nodais; direita erro absoluto obtido por comparação entre a solução numérica e a analítica.

Com as duas soluções através de diferentes esquemas para a difusão com o termo transiente nota-se a possibilidade de substituir um esquema por outro sempre que necessário. Uma outra possibilidade de discretização temporal é o uso do esquema de Crank-Nicolson, que também possui boa precisão.

5.3.5 Difusão do Calor em 2D com “ θ -method”

O estudo da difusão do calor em 2D usando o método de volumes finitos pode ser feito de forma similar ao já efetuado em 1D. A principal diferença consiste no aumento de pontos vizinhos ao ponto investigado. Durante o processo de discretização, ao invés de estudar as propriedades ϕ_W, ϕ_P e ϕ_E , agora teremos $\phi_W, \phi_E, \phi_P, \phi_S$ e ϕ_N .

A difusão do calor em 2D pode ser representada pela equação,

$$\rho c_p \frac{\partial T}{\partial t} = \nabla \cdot (\lambda \nabla T), \quad (5.81)$$

que pode ser reescrita de forma conveniente, similar à equação geral do transporte na ausência de termo de fontes e convecção,

$$\frac{\partial T}{\partial t} = \nabla \cdot \left(\frac{\lambda}{\rho c_p} \nabla T \right). \quad (5.82)$$

Ao declarar as quantidades $\rho \equiv 1$, $\Gamma \equiv \lambda/\rho c_p$, $\phi \equiv T$, obtemos

$$\frac{\partial \phi}{\partial t} = \nabla \cdot (\Gamma \nabla \phi). \quad (5.83)$$

Para discretizar a equação (5.83) usando o método de volumes finitos, basta integrar a mesma no volume e no tempo, e em seguida aplicar o teorema de Gauss,

$$\int_t \int_V \frac{\partial \phi}{\partial t} dV dt = \int_t \int_V \nabla \cdot (\Gamma \nabla \phi) dV dt = \int_t \int_A (\Gamma \nabla \phi) \cdot \hat{n} dA dt. \quad (5.84)$$

A discretização da parte espacial no termo da direita será executada usando a interpolação por diferenças centrais, enquanto a discretização do termo temporal da esquerda será executada usando o método θ , que permite escolher entre os três procedimentos de discretização temporal mais usados: (1) Forward Euler, (2) Cranck Nicolson, (3) Backward Euler, bastando escolher entre os valores de θ : $\theta = 0, 0$, $\theta = 0, 5$, $\theta = 1, 0$, respectivamente. Logo a equação (5.84) discretizada assume a forma

$$\begin{aligned}
\frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= \delta y \left(\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right) (1 - \theta) + \\
&+ \delta x \left(\Gamma_n \frac{\phi_N^t - \phi_P^t}{\delta y} - \Gamma_s \frac{\phi_P^t - \phi_S^t}{\delta x} \right) (1 - \theta) + \\
&+ \delta y \left(\Gamma_e \frac{\phi_E^{t+1} - \phi_P^{t+1}}{\delta x} - \Gamma_w \frac{\phi_P^{t+1} - \phi_W^{t+1}}{\delta x} \right) \theta + \\
&+ \delta x \left(\Gamma_n \frac{\phi_N^{t+1} - \phi_P^{t+1}}{\delta y} - \Gamma_s \frac{\phi_P^{t+1} - \phi_S^{t+1}}{\delta x} \right) \theta. \tag{5.85}
\end{aligned}$$

Em seguida basta reorganizar os coeficientes e nomear de forma conveniente cada parcela,

$$\begin{aligned}
\frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= \left(\underbrace{\frac{\delta y}{\delta x} \Gamma_e \phi_E^t}_{a_E} + \underbrace{\frac{\delta y}{\delta x} \Gamma_w \phi_W^t}_{a_W} + \underbrace{\frac{\delta x}{\delta y} \Gamma_s \phi_S^t}_{a_S} + \underbrace{\frac{\delta x}{\delta y} \Gamma_n \phi_N^t}_{a_N} \right) (1 - \theta) + \\
&+ \left(-\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} - \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} - \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} - \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \right) \phi_P^t (1 - \theta) + \\
&+ \left(\underbrace{\frac{\delta y}{\delta x} \Gamma_e \phi_E^{t+1}}_{a_E} + \underbrace{\frac{\delta y}{\delta x} \Gamma_w \phi_W^{t+1}}_{a_W} + \underbrace{\frac{\delta x}{\delta y} \Gamma_s \phi_S^{t+1}}_{a_S} + \underbrace{\frac{\delta x}{\delta y} \Gamma_n \phi_N^{t+1}}_{a_N} \right) \theta + \\
&+ \left(-\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} - \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} - \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} - \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \right) \phi_P^{t+1} \theta, \tag{5.86}
\end{aligned}$$

ou seja,

$$\begin{aligned}
\frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= [a_E \phi_E^t + a_W \phi_W^t + a_S \phi_S^t + a_N \phi_N^t + a_P \phi_P^t] (1 - \theta) + \\
&+ [a_E \phi_E^{t+1} + a_W \phi_W^{t+1} + a_S \phi_S^{t+1} + a_N \phi_N^{t+1} + a_P \phi_P^{t+1}] \theta, \tag{5.87}
\end{aligned}$$

onde o termo a_P é definido como $a_P = -a_E - a_W - a_S - a_N$. O próximo passo é selecionar uma quantidade de pontos e realizar a discretização. Para exemplificar vamos escolher um domínio 3×3 ,

ϕ_{31}	ϕ_{32}	ϕ_{33}
ϕ_{21}	ϕ_{22}	ϕ_{23}
ϕ_{11}	ϕ_{12}	ϕ_{13}

Tabela 5: Representação de um domínio 2D com propriedade ϕ no centro de cada célula.

Usando a equação (5.87) para obter a discretização da propriedade ϕ_{22} obtemos

$$\begin{aligned} \frac{\delta x \delta y}{\delta t} (\phi_{22}^{t+1} - \phi_{22}^t) &= [a_E^{23} \phi_{23}^t + a_W^{21} \phi_{21}^t + a_S^{12} \phi_{12}^t + a_N^{32} \phi_{32}^t + a_P^{22} \phi_{22}^t] (1 - \theta) + \\ &+ [a_E^{23} \phi_{23}^{t+1} + a_W^{21} \phi_{21}^{t+1} + a_S^{12} \phi_{12}^{t+1} + a_N^{32} \phi_{32}^{t+1} + a_P^{22} \phi_{22}^{t+1}] \theta. \end{aligned} \quad (5.88)$$

Aplicando a equação (5.87) para todos os demais pontos, o sistema linear é obtido,

$$\frac{\delta x \delta y}{\delta t} (\tilde{\phi}^{t+1} - \tilde{\phi}^t) = (\tilde{M} \tilde{\phi}^t + \tilde{N}) (1 - \theta) + (\tilde{M} \tilde{\phi}^{t+1} + \tilde{N}) \theta, \quad (5.89)$$

onde

$$\tilde{\phi}^{t+1} \equiv \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix}^{t+1}, \quad \tilde{\phi}^t \equiv \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix}^t, \quad \tilde{N} \equiv \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \\ b_{21} \\ 0 \\ b_{23} \\ b_{31} \\ b_{32} \\ b_{33} \end{bmatrix}. \quad (5.90)$$

Em (5.90) os seguintes coeficientes foram renomeados por questão de simplicidade:

$$\begin{aligned} b_{11} &\equiv a_S^{01} \phi_{01} + a_W^{10} \phi_{10}, \quad b_{12} \equiv a_S^{02} \phi_{02}, \quad b_{13} \equiv a_S^{03} \phi_{03} + a_E^{14} \phi_{14}, \quad b_{21} \equiv a_W^{20} \phi_{20}, \quad b_{23} \equiv a_E^{24} \phi_{24}, \\ b_{31} &\equiv a_N^{41} \phi_{41} + a_W^{30} \phi_{30}, \quad b_{32} \equiv a_N^{42} \phi_{42}, \quad b_{33} \equiv a_N^{43} \phi_{43} + a_E^{34} \phi_{34}. \end{aligned}$$

Finalmente, podemos notar que

a matriz \widetilde{M} é uma matriz penta-diagonal com falhas e pode ser representada como

$$\widetilde{M} \equiv \begin{bmatrix} a_P^{11} & a_E^{12} & 0 & a_N^{21} & 0 & 0 & 0 & 0 & 0 \\ a_W^{11} & a_P^{12} & a_E^{13} & 0 & a_N^{22} & 0 & 0 & 0 & 0 \\ 0 & a_W^{12} & a_P^{13} & 0 & 0 & a_N^{23} & 0 & 0 & 0 \\ a_S^{11} & 0 & 0 & a_P^{21} & a_E^{22} & 0 & a_N^{31} & 0 & 0 \\ 0 & a_S^{12} & 0 & a_W^{21} & a_P^{22} & a_E^{23} & 0 & a_N^{32} & 0 \\ 0 & 0 & a_S^{13} & 0 & a_W^{22} & a_P^{23} & 0 & 0 & a_N^{33} \\ 0 & 0 & 0 & a_S^{21} & 0 & 0 & a_P^{31} & a_E^{32} & 0 \\ 0 & 0 & 0 & 0 & a_S^{22} & 0 & a_W^{31} & a_P^{32} & a_E^{33} \\ 0 & 0 & 0 & 0 & 0 & a_S^{23} & 0 & a_W^{32} & a_P^{33} \end{bmatrix}. \quad (5.91)$$

Neste estágio, basta resolver o sistema linear da equação (5.92),

$$\underbrace{\left[\frac{\delta x \delta y}{\delta t} \mathbb{1} - \theta \widetilde{M} \right]}_A \widetilde{\phi}^{t+1} = \underbrace{\left[\widetilde{M} (1 - \theta) + \frac{\delta x \delta y}{\delta t} \right]}_B \widetilde{\phi}^t + \widetilde{N}, \quad (5.92)$$

$$\widetilde{A} \widetilde{\phi} = \widetilde{B}.$$

O programa heatTransfer2D.R implementa a equação (5.92), e está nos apêndices.

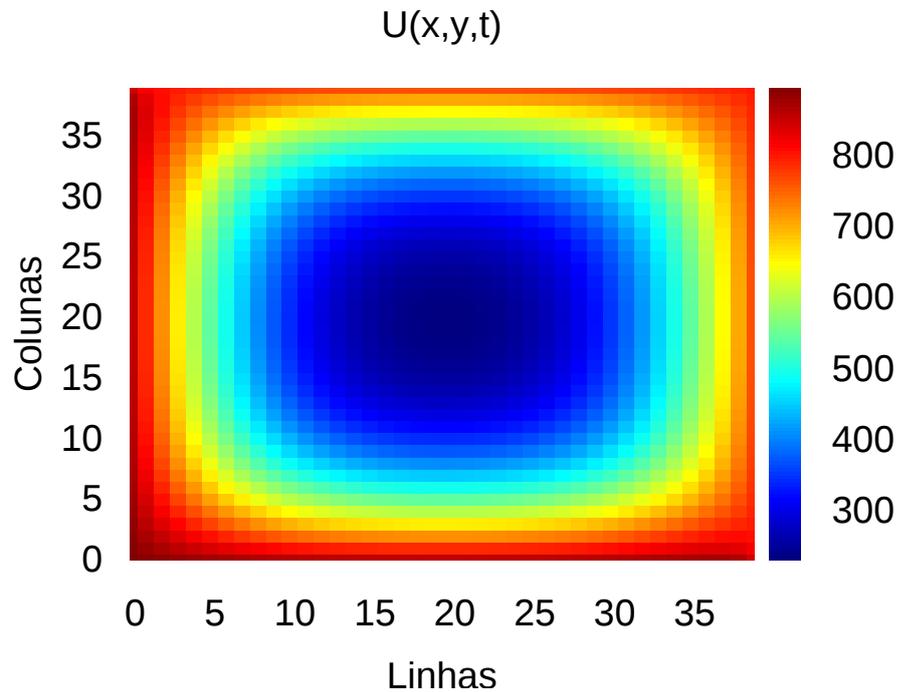


Figura 35: Exemplo de simulação com 40×40 pontos usando o programa `heatTransfer2D.R` após o instante $2k$.

Este estudo da difusão do calor em 2D deu origem ao trabalho “Heat Transfer in Heterogeneous Domain” que será desenvolvido nos apêndices.

5.3.6 Validação do Solver

O solver desenvolvido acima pode ser comparado com o solver desenvolvido em MATLAB por Dominik Gibala em 2012, disponível online no link:

<http://www.mathworks.com/matlabcentral/fileexchange/35068-gui-2d-heat-transfer>.

Na sequência apresentamos as soluções obtidas usando o solver `35068-gui-2d-heat-transfer` e as soluções do nosso solver desenvolvido em R para efeito de comparação. Para efetuar tal comparação iremos estudar a difusão do calor regida pela seguinte equação

$$\frac{\partial u}{\partial t} = \frac{\lambda}{\rho c_p} \nabla^2 u. \quad (5.93)$$

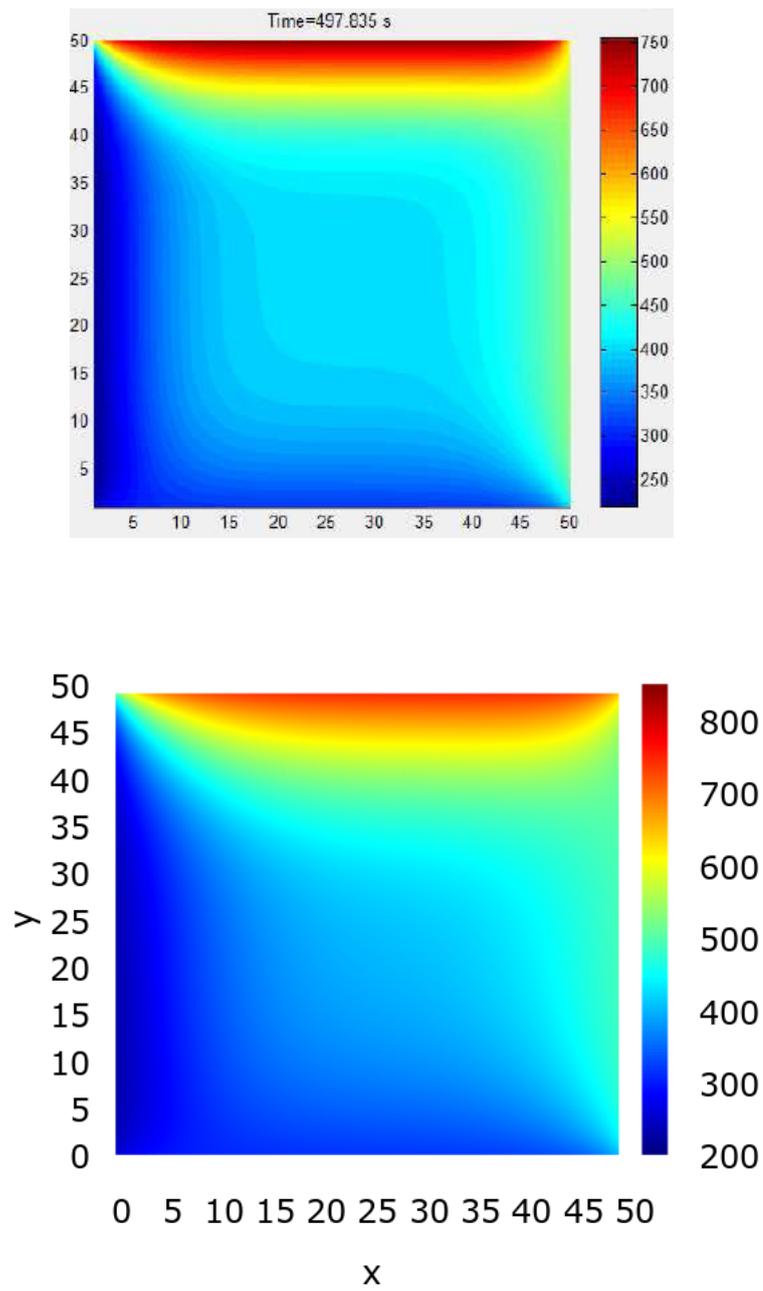


Figura 36: Superior: Solução obtida pelo solver 35068-gui-2d-heat-transfer. Inferior: Solução obtida pelo nosso solver desenvolvido em R. Onde $\rho = 7600$, $c_p = 480$, $\lambda = 47$, $u_0 = 400$, $u_{left} = 200$, $u_{right} = 500$, $u_{bottom} = 300$, $u_{top} = 800$ em unidades do SI.

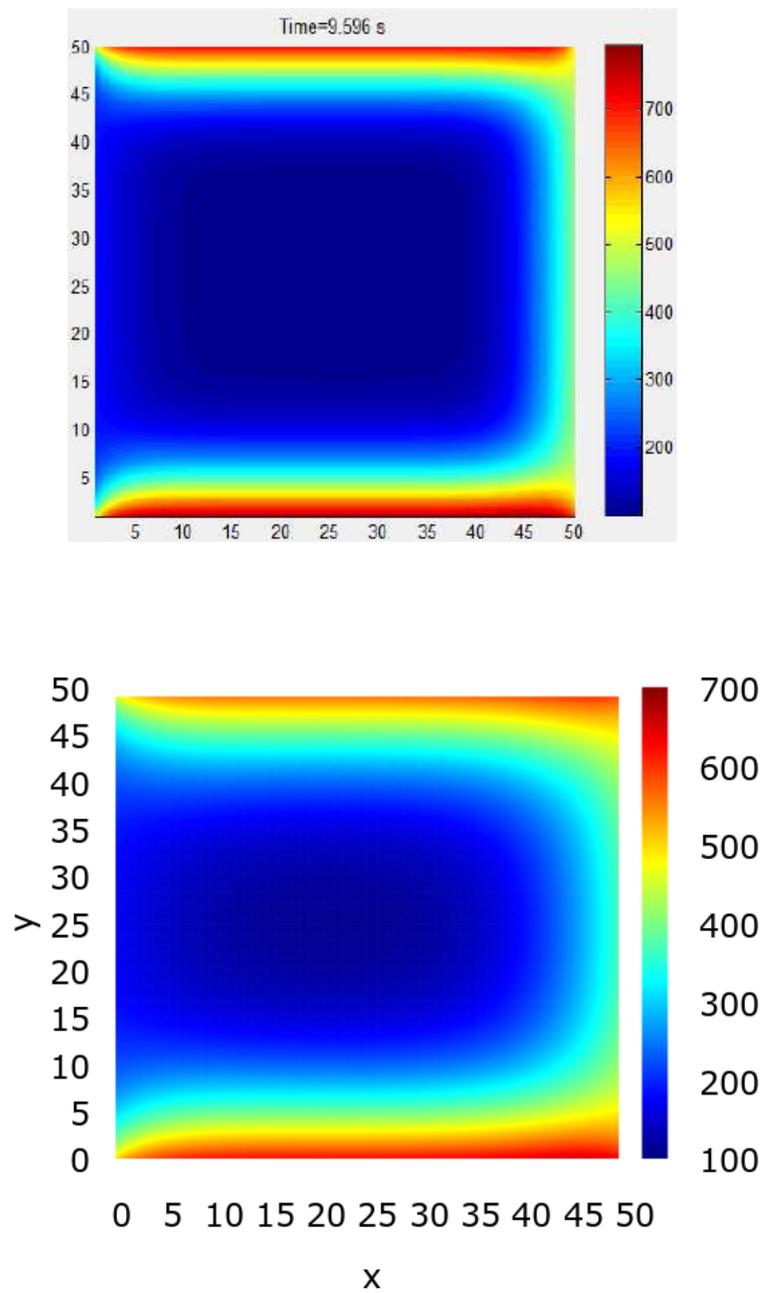


Figura 37: Superior: Solução obtida pelo solver 35068-gui-2d-heat-transfer. Inferior: Solução obtida pelo nosso solver desenvolvido em R. Onde $\rho = 1000$, $c_p = 480$, $\lambda = 200$, $u_0 = 100$, $u_{left} = 200$, $u_{right} = 500$, $u_{bottom} = 900$, $u_{top} = 800$ em unidades do SI.

5.3.7 Escoamento Bifásico com OpenFOAM

Para estudar o escoamento bifásico no OpenFOAM, desenvolvemos um novo solver para as equações da saturação e da pressão. Este solver foi desenvolvido a partir do solver do escoamento incompressível do OpenFOAM.

Inicialmente o arquivo param.H é definido, contendo

```
//-----
lambn = (pow((1-sw),2)*(1-pow(sw,(2.0+theta)/theta)))/mun;
lambw = (pow(sw,((2.0+3.0*theta)/theta)))/muw;
pc = pd*pow(sw,-(1/eta)); dpc = -pd/eta*pow(sw,-((1+eta)/eta));
//-----
```

Em seguida definimos o arquivo pEqn.H,

```
//-----
{
fvScalarMatrix pEqn
(
fvm::laplacian((lambn+lambw)*K,pw) + fvc::laplacian((lambn*dpc)*K,sw)
);
solve(pEqn == -(qn+qw)
);
}
//-----
```

A equação da saturação será definida no arquivo sEqn.H

```
//-----
{
surfaceScalarField phiGG = linearInterpolate((rho*w*lambw*K) & g) & mesh.Sf();
fvScalarMatrix sEqn
```

```
(
fvM::ddt(porosity,sw) - fvc::laplacian(lambw*K,pw) + fvc::div(phiGG) - qw
);
sEqn.relax();
sEqn.solve();//solve(sEqn == qw);
}
//-----
```

Finalmente definimos o arquivo `escoamentoFoam.C` que incluirá os arquivos `param.H`, `pEqn.H` e `sEqn.H`.

Neste arquivo há também as equações para o cálculo das velocidades da água e do óleo.

```
//-----
Vw = -(lambw*K) & fvc::grad(pw);
Vn = -((lambn*K) & (fvc::grad(pw) + dpc*fvc::grad(sw)));
//-----
```

5.3.8 Geração da Malha

A malha é definida em um arquivo chamado `blockMeshDict`. Para construir a malha basta usar o comando `blockMesh`, em seguida esta pode ser visualizada usando o `paraFoam`,

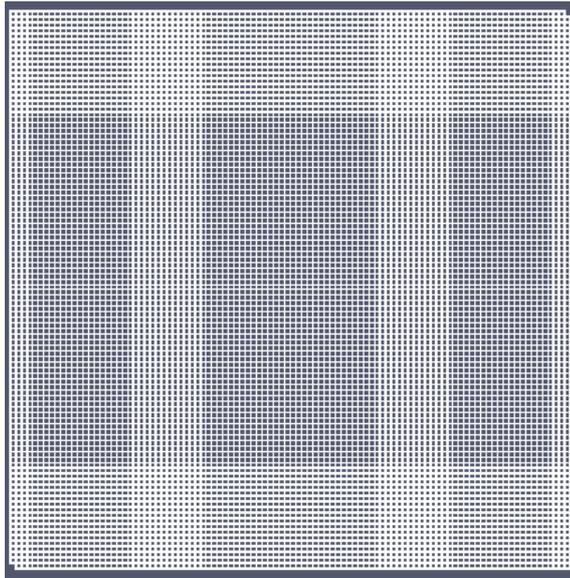


Figura 38: Malha contendo 9998 células gerada pelo comando blockMesh para ser utilizada no estudo do escoamento bifásico. No canto inferior esquerdo será definido o injetor e no canto superior direito será definido o produtor.



Figura 39: Simulação do escoamento bifásico em uma configuração do tipo 1/4 de five spot, após 1500s, contendo um injetor no canto inferior esquerdo e um produtor no canto superior direito, com uma região central de baixa permeabilidade. Dados do problema: $dt = 0.1$, $pw_{inj} = 3e6Pa$, $pw_{prod} = 1e6Pa$, $k = 1e - 8m^2$, $k_{low} = 1e - 11m^2$, $\rho_w = 1000kg \cdot m^{-3}$, $\mu_w = 1e - 3Pa \cdot s$, $\mu_n = 1e - 2Pa \cdot s$, $\theta = 2$, $\phi = 0.2$.

Este estudo do escoamento bifásico com o OpenFOAM deu origem ao trabalho “Influence of the Uncertainty of Geological Parameters on Numerical Reservoir Simulators” que será desenvolvido nos apêndices.

6 CONSIDERAÇÕES FINAIS

Neste trabalho de tese foi feito um levantamento dos modelos em equações diferenciais parciais para o escoamento em meios porosos, e de técnicas numéricas para sua solução computacional. Apresentamos também modelos simples para melhorar a compreensão dos conceitos estudados. O nosso intuito é analisar os elementos necessários para a utilização de uma infra-estrutura computacional baseada no paradigma de computação em nuvens para tratar problemas da física computacional. A infra-estrutura de software foi baseada nos programas de domínio público R e OpenFOAM, para os quais desenvolvemos rotinas e uma metodologia para a solução de equações diferenciais parciais usando a técnica de volumes finitos.

Vale aqui ressaltar que neste trabalho não foi apresentado um estudo das condições de existência e estabilidade das soluções numéricas dos problemas estudados, sem prejuízo, pois estes podem ser encontrados na literatura, da qual nos referimos especialmente às referências [21], [15].

Como perspectiva para trabalhos futuros, listamos:

- Expandir a metodologia desenvolvida para incluir aplicações com o método de volumes finitos em malhas não estruturadas;
- Implementar em R uma metodologia semelhante usando a técnica do Galerkin descontínuo;
- Analisar a viabilidade dos métodos desenvolvidos para estudar problemas de escoamento em regime turbulento, especificamente simulações diretas DNS;
- Estudar o escoamento multi-componente em meios porosos usando outros softwares livres tais como DUNE e Fenics.

APPENDIX A – Dimensionless Equations for Two-Phase Isothermal Flow in Porous Media

XXXIV ENFMC, Foz do Iguaçu, June 2011

Marceliano Eduardo de Oliveira, Murilo Pereira de Almeida

Laboratório de Simulação Numérica, Departamento de Física, Universidade Federal do
Ceará, Fortaleza, Ceará, Brasil

A.1 Introduction

In this work we present a systematic way to convert dimensional into dimensionless equations starting from the concept of adimensional groups established by Buckingham's π theorem. As an application of this method we adimensionalize the equations that model the isothermal flow of two immiscible fluids in porous media.

A.2 Methodology

The first step in the adimensionalization process is the redefinition of a dimensional quantity A as a product of two factors, \bar{A} and A' , where \bar{A} holds the dimension of A and A' is a purely numerical quantity.

By substituting these products into the equation, one can collect the dimensional factors, usually into a fraction, generating a pure number.

The Buckingham's π theorem provides a way to get all the k pure numbers involved in a problem, where $k = (n - m)$ is the difference between the numbers of parameters n and the minimum number of fundamental dimensions m necessary to describe them.

After getting all the π_i numbers, we determine the set of quantities \bar{A} .

A.3 Illustrative Example: Vertical Motion

We will illustrate this method with the equation of the vertical motion under constant acceleration

Equation	$\frac{\partial^2 z}{\partial t^2} = -g$			
Parameters	z_0	t_0	v_0	g
Dimensions	L	T	LT^{-1}	LT^{-2}

Table 1: Vertical Motion

As we can observe in table 1, the four parameters z_0 , t_0 , v_0 and g need only two fundamental dimensions for their complete description, L and T . Each independent parameter produces one equation. We can see in table 2 one equation for the powers of L and another for the powers of T . Powers of other independent fundamental quantities, that are unnecessary for the problem description such as M , θ (temperature), Q (electric charge) and others ..., are not used.

In order to form a basis, it is necessary to fix a number of parameters equal to the number of independent dimensions, raising each to a different power. This ensure a number of equations equal to the amount of unknown exponents.

As we can see in table 1, the present equation has four parameters ($n = 4$), requiring only two fundamental dimensions ($m = 2$), L and T . The k possible dimensionless numbers for each basis is established by Buckingham's π theorem [?], by the relation $k = n - m$. In this case, the numbers π_1 and π_2 with basis $v_0^{x_i} z_0^{y_i}$ can be seen below

Proposals	$\pi_1 = v_0^{x_1} z_0^{y_1} g$			$\pi_2 = v_0^{x_2} z_0^{y_2} t_0$		
Parameters	$v_0^{x_1}$	$z_0^{y_1}$	g	$v_0^{x_2}$	$z_0^{y_2}$	t_0
Powers of L	x_1	y_1	1	x_2	y_2	0
Powers of T	$-x_1$	0	-2	$-x_2$	0	1
Results	$\pi_1 = v_0^{-2} z_0 g$			$\pi_2 = v_0 z_0^{-1} t_0$		

Table 2: Dimensionless numbers.

The elements in table 2 form the so-called dimensional matrix, in this case, $M(\pi_1)$ and $M(\pi_2)$ which are used to obtain the values x_1, y_1 and x_2, y_2 which cancel the dimensions of the numbers π_1 and π_2 . In some cases, it is useful to put the dimensional matrix in echelon form.

As an example, the equation $\frac{\partial^2}{\partial t^2}z = -g$ will be modified by the relations $t = \bar{t}'$, $z = \bar{z}z'$ and $g = \bar{g}g'$, resulting in $\bar{t}^{-2}\bar{z}\frac{\partial^2}{\partial \bar{t}'^2}z' = -\bar{g}g'$. At this stage, it is possible to link the dimensionless numbers π with the adimensionalization process through the proper choice of the factors $\bar{z}, \bar{t}, \bar{v}, \bar{g}$; by observing π_1 we note that it depends on z_0, v_0 , and g parameters but does not depend on t_0 . Thus, we chose $\bar{z} = z_0, \bar{v} = v_0, \bar{g} = g$, and replace them in the equation, getting: $\frac{\partial^2}{\partial \bar{t}'^2}z' = -\frac{\bar{t}^2 g}{z_0}g'$. Now it is necessary to choose correctly \bar{t} in order to obtain π_1 , i.e., $\pi_1 = \frac{z_0 g}{v_0^2} = \frac{\bar{t}^2 g}{z_0} \Rightarrow \bar{t} = z_0 v_0^{-1}$. The result of this procedure is seen in the table below,

Variables	\bar{z}	\bar{t}	\bar{v}	\bar{g}
Use of π_1	z_0	$z_0 v_0^{-1}$	v_0	g
Use of π_2	z_0	t_0	v_0	$v_0 t_0^{-1}$

Table 3: Relations.

The relationships obtained in table 3 perform the adimensionalization of the problem.

Transformations using π_1 or π_2 :

$$\left\{ z, \frac{\partial^2}{\partial t^2}, g \right\} \rightarrow \left\{ z_0 z', (z_0 v_0^{-1})^{-2} \frac{\partial^2}{\partial \bar{t}'^2}, g g' \right\},$$

$$\left\{ z, \frac{\partial^2}{\partial t^2}, g \right\} \rightarrow \left\{ z_0 z', (t_0)^{-2} \frac{\partial^2}{\partial \bar{t}'^2}, v_0 t_0^{-1} g' \right\}.$$

Using the above transformations in equation $\frac{\partial^2}{\partial t^2}z = -g$, the following dimensionless forms are obtained,

Dimensionless forms:

$$\frac{\partial^2}{\partial \bar{t}'^2}z' = -\pi_1 g',$$

$$\frac{\partial^2}{\partial \bar{t}'^2}z' = -\pi_2 g'.$$

The adimensionalization resulting from the use of π_1 is extremely sensitive to different initial velocities, as can be seen in figure (40). In other words, π_1 works as a weight establishing a regime in which the value g' does not contribute significantly to the dimensionless equation.

In addition the number π_2 becomes meaningless if $t_0 = 0$. Also, using π_1 , g' becomes unitary π_1 while using π_2 , $g' = gt_0v_0^{-1}$, in other words, another number π'_2 appears making the results for the transformation π_2 dependent on two parameters: π_2 and π'_2 .

Other bases could be used. For example, the base $t_0^{x_i}v_0^{y_i}$; but this produces dimensionless numbers $\pi_1 = z_0v_0^{-1}t_0^{-1}$ and $\pi_2 = gv_0^{-1}t_0$, which are both uninteresting if $t_0 = 0$, because π_1 goes to infinity and π_2 goes to zero.

A.4 Example: Navier-Stokes Equation

Another interesting example is the flow of fluids through a pipe with a diameter l . In this type of flow, governed by the Navier-Stokes equations, the process of adimensionalization is useful, because it produces naturally the Reynolds number that characterizes the different types of flow: laminar, turbulent or unstable [1]. This problem can be formulated as shown below:

Equation	$\frac{\partial}{\partial t}\vec{v} + \vec{v} \cdot \nabla \vec{v} = -\frac{1}{\rho}\nabla p + \frac{\eta}{\rho}\nabla^2\vec{v}$				
Parameters	l	v	ρ	p	η
Dimensions	L	LT^{-1}	ML^{-3}	$ML^{-1}T^{-2}$	$ML^{-1}T^{-1}$

Table 4: Navier-Stokes: flow in a pipe.

Applying the Buckingham's π theorem in table 4, with the base $(l^{x_i}v^{y_i}\rho^{z_i})$, we establish tentative dimensionless numbers π_1 and π_2

Proposals	$\pi_1 = l^{x_1} v^{y_1} \rho^{z_1} p$				$\pi_2 = l^{x_2} v^{y_2} \rho^{z_2} \eta$			
Parameters	l^{x_1}	v^{y_1}	ρ^{z_1}	p	l^{x_2}	v^{y_2}	ρ^{z_2}	η
Powers of M	0	0	z_1	1	0	0	z_2	1
Powers of L	x_1	y_1	$-3z_1$	-1	x_2	y_2	$-3z_2$	-1
Powers of T	0	$-y_1$	0	-2	0	$-y_2$	0	-1
Results	$\pi_1 = v^{-2} \rho^{-1} p$				$\pi_2 = l^{-1} v^{-1} \rho^{-1} \eta$			

Table 5: Navier Stokes: π_1 and π_2 .

Next, we write the relations between the dimensional variables \bar{A} and the parameters that appear in the π numbers, as done in the vertical motion:

Transformations using π_1 or π_2 :

$$\left\{ \vec{v}, \frac{\partial}{\partial t}, \nabla, \rho, \eta, p \right\} \rightarrow \left\{ v\vec{v}', (lv^{-1})^{-1} \frac{\partial}{\partial t'}, l^{-1}\nabla', \rho\rho', \rho lv\eta', pp' \right\},$$

$$\left\{ \vec{v}, \frac{\partial}{\partial t}, \nabla, \rho, \eta, p \right\} \rightarrow \left\{ v\vec{v}', (lv^{-1})^{-1} \frac{\partial}{\partial t'}, l^{-1}\nabla', \rho\rho', \eta\eta', \rho v^2 p' \right\}.$$

Using the transformations above, the following dimensionless forms are obtained.

Dimensionless forms:

$$\frac{\partial}{\partial t'} \vec{v}' + \vec{v}' \cdot \vec{\nabla}' \vec{v}' = - \frac{\pi_1}{\rho'} \vec{\nabla}' p' + \frac{\eta'}{\rho'} \nabla'^2 \vec{v}',$$

$$\frac{\partial}{\partial t'} \vec{v}' + \vec{v}' \cdot \vec{\nabla}' \vec{v}' = - \frac{1}{\rho'} \vec{\nabla}' p' + \pi_2 \frac{\eta'}{\rho'} \nabla'^2 \vec{v}'.$$

The number π_2 is the inverse of the Reynolds number, $\pi_2 = Re^{-1}$. Moreover, for π_2 the transformation ($\eta \rightarrow \eta\eta'$) makes the constant η' to take a unitary value, and the equation depends on only one parameter; but for π_1 , η' takes another value, and the equation depends on two parameters.

We can observe that π_2 depends on the viscosity, an important parameter for the different types of flows.

In figure (41), we compare the π_2 sensitivity to v , for water and olive oil. It is noticeable that π_2 is less sensitivity to v for water. In this case, the use of a constant $\pi_2 = 0$ has little effect and we recover the Euler equation as a good approximation.

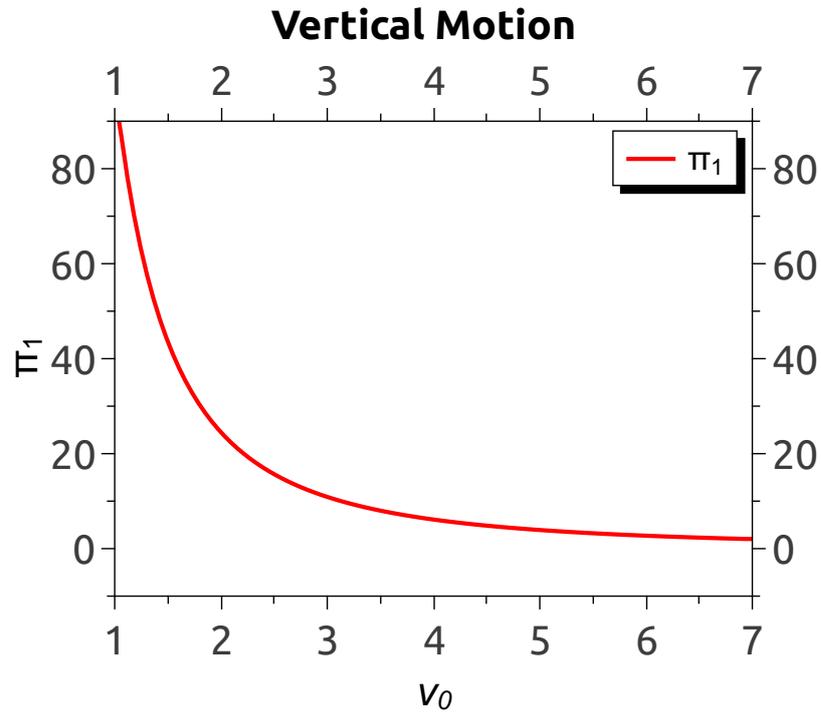


Figure 40: Vertical Motion: $z_0 = 10 \text{ m}$, $g = 9,8 \text{ ms}^{-2}$.

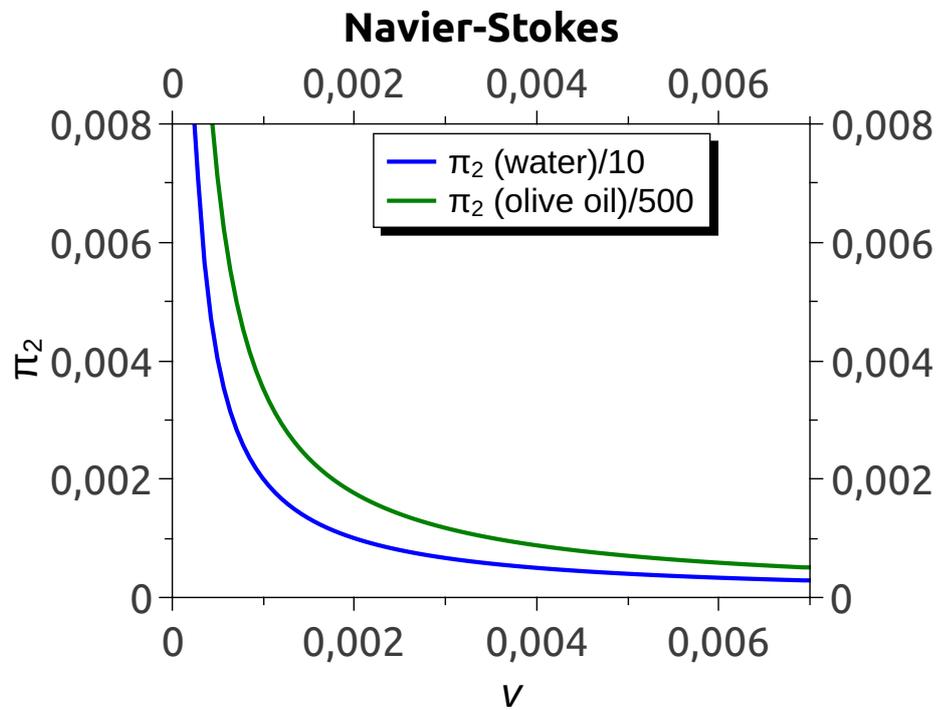


Figure 41: Navier-Stokes: ($l = 5 \text{ cm}$), $Re_{H_2O} < 3000$.

A.5 Two-phase Isothermal Flow in Porous Media

Using the Darcy's law, $u_\alpha = -\lambda_\alpha k \nabla p_\alpha$, $\alpha = w, n$ where w and n are the wetting and non-wetting phases, respectively, the relations, $s_w + s_n = 1$, $p_c = p_n - p_w$, $\lambda_t = \lambda_w + \lambda_n$ and the continuity equation $\phi \frac{\partial}{\partial t} s_\alpha + \nabla \cdot u_\alpha = q_\alpha$, we get the equations for the isothermal flow of two immiscible fluids in porous media [2]:

Model equations:

$$\begin{aligned} -\nabla \cdot (\lambda_t k \nabla p_w + \lambda_n k \nabla p_c) &= q_w + q_n, \\ -\phi \frac{\partial}{\partial t} s_n - \nabla \cdot (\lambda_w k \nabla p_w) &= q_w. \end{aligned}$$

For simplicity, it is interesting to write the above equations in a different form:

Changing the notation:

$$\begin{aligned} -\sum_i \nabla \cdot (\lambda_i k \nabla p_i) &= \sum_i q_i, \quad i = 1, 2, \\ -\phi \frac{\partial}{\partial t} s_n - \nabla \cdot (\lambda_w k \nabla p_w) &= q_w. \end{aligned}$$

This is sufficient to highlight the presence of the term $-\nabla \cdot (k \lambda_i \nabla p_i) = q_i$, whose adimensionalization facilitates the rest of the procedure.

Equation	$-\nabla \cdot (k \lambda_i \nabla p_i) = q_i$					
Parameters	t	A	k	Φ	μ	p
Dimensions	T	L^2	L^2	$L^3 T^{-1}$	$ML^{-1} T^{-1}$	$ML^{-1} T^{-2}$

Table 6: Two-phase isothermal flow in porous media

The set of parameters involved in this model gives the following dimensionless groups:

$\pi_1 = \mu^{x_1} A^{y_1} t^{z_1} p k$					$\pi_2 = \mu^{x_2} A^{y_2} t^{z_2} p \Phi$					$\pi_3 = \mu^{x_3} A^{y_3} t^{z_3} k \Phi$				
μ^{x_1}	A^{y_1}	t^{z_1}	p	k	μ^{x_2}	A^{y_2}	t^{z_2}	p	Φ	μ^{x_3}	A^{y_3}	t^{z_3}	k	Φ
x_1	0	0	1	0	x_2	0	0	1	0	x_3	0	0	0	0
$-x_1$	$2y_1$	0	-1	2	$-x_2$	$2y_2$	0	-1	3	$-x_3$	$2y_3$	0	2	3
$-x_1$	0	z_1	-2	0	$-x_2$	0	z_2	-2	-1	$-x_3$	0	z_3	0	-1
$\pi_1 = \mu^{-1} A^{-1} t p k$					$\pi_2 = \mu^{-1} A^{-3/2} t^2 p \Phi$					$\pi_3 = A^{5/2} t k \Phi$				

Table 7: Two-phase isothermal flow in porous media: π_1 , π_2 and π_3 .

Similar to the vertical motion, we obtain the transformations

Transformations using π_1 , π_2 or π_3 :

$$\begin{aligned} \{\nabla, k, \lambda_i, p_i, q_i\} &\rightarrow \{A^{-1/2}\nabla', kk', \mu^{-1}\lambda'_i, p_i p'_i, t^{-1}q'_i\}, \\ \{\nabla, k, \lambda_i, p_i, q_i\} &\rightarrow \{(t\Phi)^{-1/3}\nabla', (t\Phi)^{5/3}A^{-3/2}k', \mu^{-1}\lambda'_i, p_i p'_i, t^{-1}q'_i\}, \\ \{\nabla, k, \lambda_i, p_i, q_i\} &\rightarrow \{A^{-1/2}\nabla', kk', \mu^{-1}\lambda'_i, \Phi A^{7/2}\mu p'_i, t^{-1}q'_i\}. \end{aligned}$$

Applying the above transformations, we get the following dimensionless forms,

Dimensionless forms:

$$\begin{aligned} -\pi_1 [\nabla' \cdot (k' \lambda'_i \nabla' p'_i)] &= q'_i, \\ -\pi_2 [\nabla' \cdot (k' \lambda'_i \nabla' p'_i)] &= q'_i, \\ -\pi_3 [\nabla' \cdot (k' \lambda'_i \nabla' p'_i)] &= q'_i. \end{aligned}$$

For fluids with considerable high viscosity, the number π_3 would be less interesting than the numbers π_1 and π_2 . Another factor to consider is simplicity of the transformation by π_1 .

Choosing the dimensionless form obtained by π_1 , we conclude the adimensionalisation

Simultaneous adimensionalisation of the equations:

$$\begin{aligned} -\sum_i \pi_1 [\nabla' \cdot (k' \lambda'_i \nabla' p'_i)] &= \sum_i q'_i, \quad i = 1, 2, \\ -\phi \bar{t}^{-1} \frac{\partial}{\partial t} s_n - \bar{t}^{-1} \pi_1 [\nabla' \cdot (k' \lambda'_w \nabla' p'_w)] &= \bar{t}^{-1} q'_w. \end{aligned}$$

Finally, we obtain the model dimensionless equations:

Dimensionless equations for two-phase isothermal flow in porous media:

$$\begin{aligned} -\pi_1 [\nabla' \cdot (k' \lambda'_t \nabla' p'_w + k' \lambda'_n \nabla' p'_c)] &= q'_w + q'_n, \\ -\phi \frac{\partial}{\partial t'} s_n - \pi_1 [\nabla' \cdot (k' \lambda'_w \nabla' p'_w)] &= q'_w. \end{aligned}$$

A.6 Concluding Remarks

The Buckingham's π theorem was used in different contexts to obtain adimensional equations of mathematical models.

When more than one dimensionless numbers were obtained, implying in more than one transformation, an analysis of context was useful for an appropriate choice of π , reducing the arbitrariness of the process.

The use of adimensional equations of the isothermal two-phase flow in porous media simplifies their computational implementation into numerical schemes.

The appearance of the Reynolds number in the case of Navier-Stokes equations is an important indicator of the efficiency of the procedure described here.

A.7 Acknowledgments

This work was financially supported by:

- **CNPQ** - Conselho Nacional de Desenvolvimento Científico e Tecnológico;
- **FCPC** - Fundação Cearense de Pesquisa e Cultura;
- **Petrobras** - Petróleo Brasileiro S. A.

REFERENCES

- [1] Landau L. D. and Lifshitz E. M.. Fluid mechanics, Oxford, Elsevier, 1987.
- [2] Y Epstheyn and Béatrice Rivière. Fully implicit discontinuous finite element methods for two-phase flow, Applied Numerical Mathematics, Elsevier 2006.
- [3] Barenblatt G. I.. Dimensional analysis. New York, Gordon and Breach Science Publishers, 1987.

APPENDIX B – Solving Diffusion-Convection Problem Using Finite Volume Method

XXXVII Brazilian Meeting on Condensed Matter Physics

Costa do Saúípe, May 2014

Marceliano E. de Oliveira, Murilo P. de Almeida, Francisco L. de Oliveira

Laboratório de Simulação Numérica, Departamento de Física, Universidade Federal do Ceará, Fortaleza, Ceará, Brasil

B.1 Abstract

The present work describes how to obtain computational solutions to general scalar transport equations. These equations are discretized using the so called finite volume method, based on the integral form of the general scalar transport equation, each of its term being approximated by a certain numerical scheme. There is a variety of schemes available for this purpose; the methodology of this work consist in implementing some of the available schemes - viz., central difference, upwind, QUICK and POWER-LAW - for particular cases of the transport equation appearing in convection-diffusion problems. The solution to the problem of convection depends only on the Courant number, while when solving the problem of pure diffusion the solution depends on the Peclet number. When convection and diffusion are solved in a single equation, the solution is dependent on both dimensionless numbers, Courant and Peclet, and in this case it is interesting to combine properly several numerical schemes and establish appropriate parameter values to ensure convergence and stability. We obtain the discretized linear system for each scheme studied, which are numerically solved and their results compared to each other in order to evaluate their efficiency. We aim at finding optimal combination of numerical schemes to solve the convection-diffusion equation, which appears in several areas of physics.

B.2 Fundamentals

To study the convection-diffusion problem we will assume the general scalar transport equation

$$\underbrace{\frac{\partial}{\partial t}(\rho\phi)}_{\text{transient}} + \underbrace{\vec{\nabla} \cdot (\rho\phi\vec{u})}_{\text{convection}} = \underbrace{\vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi)}_{\text{diffusion}} + \underbrace{S}_{\text{sources}}. \quad (\text{B.1})$$

We will consider a benchmark problem in stationary case in the absence of sources

$$0 + \underbrace{\vec{\nabla} \cdot (\rho\phi\vec{u})}_{\text{convection}} = \underbrace{\vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi)}_{\text{diffusion}} + \underbrace{0}_{\text{sources}}. \quad (\text{B.2})$$

Integrating this equation in volume, we obtain the approximation

$$\int_V \vec{\nabla} \cdot (\rho\phi\vec{u}) dV = \int_V \vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi) dV, \quad (\text{B.3})$$

which, by the application of Gauss theorem, results into

$$\int_A (\rho\phi\vec{u}) \cdot \hat{n} dA = \int_A (\Gamma\vec{\nabla}\phi) \cdot \hat{n} dA. \quad (\text{B.4})$$

These integrals are performed on the surface of the finite volume elements. For one dimensional problem the equation (B.4) becomes

$$(\rho\phi u)|_w^e = \left(\Gamma \frac{\partial}{\partial x} \phi \right) \Big|_w^e, \quad (\text{B.5})$$

Where e (east) represents the right side and the w (west) the left side of the element.

B.3 Diffusion Scheme

The diffusion term in equation (B.3) will be approximated by

$$\underbrace{\int_V \vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi) dV}_{1D} = \left(\Gamma \frac{\partial}{\partial x} \phi \right) \Big|_w^e \simeq \underbrace{\frac{\Gamma_w}{\delta x}}_{a_W} \phi_W + \underbrace{\left(-\frac{\Gamma_w}{\delta x} - \frac{\Gamma_e}{\delta x} \right)}_{a_P} \phi_P + \underbrace{\frac{\Gamma_e}{\delta x}}_{a_E} \phi_E. \quad (\text{B.6})$$

By using a minimal quantity of points (3 points) to obtain the diffusion term approximation as a linear algebra system we get a system in the form

$$\underbrace{\int_V \vec{\nabla} \cdot (\Gamma \vec{\nabla} \phi) dV}_{1D} = \left(\Gamma \frac{\partial}{\partial x} \phi \right) \Big|_w^e \simeq \begin{pmatrix} a_{P1} & a_{E1} & 0 \\ a_{W2} & a_{P2} & a_{E2} \\ 0 & a_{W3} & a_{P3} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} + \begin{pmatrix} a_{W1} \\ 0 \\ a_{E3} \end{pmatrix} \begin{pmatrix} \phi_0 \\ 0 \\ \phi_4 \end{pmatrix}. \quad (\text{B.7})$$

B.4 Convection: Central Difference Scheme

For the central difference schemes we have

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi) \Big|_w^e \simeq \underbrace{\left(-\frac{1}{2} \rho_w u_w \right)}_{a_W} \phi_W + \underbrace{\left(\frac{1}{2} \rho_e u_e - \frac{1}{2} \rho_w u_w \right)}_{a_P} \phi_P + \underbrace{\left(\frac{1}{2} \rho_e u_e \right)}_{a_E} \phi_E. \quad (\text{B.8})$$

Using a minimal quantity of points (3 points) to obtain the convection term approximation as a linear algebra system we get a system of the form

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi) \Big|_w^e \simeq \begin{pmatrix} a_{P1} & a_{E1} & 0 \\ a_{W2} & a_{P2} & a_{E2} \\ 0 & a_{W3} & a_{P3} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} + \begin{pmatrix} a_{W1} \\ 0 \\ a_{E3} \end{pmatrix} \begin{pmatrix} \phi_0 \\ 0 \\ \phi_4 \end{pmatrix}, \quad (\text{B.9})$$

B.5 Convection: Upwind Scheme

The Upwind Scheme sets the ϕ values using rules based on flux, like

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi) \Big|_w^e \simeq \begin{cases} \rho_e u_e \phi_P - \rho_w u_w \phi_W & \text{if } flux > 0, \\ \rho_e u_e \phi_E - \rho_w u_w \phi_P & \text{if } flux < 0. \end{cases} \quad (\text{B.10})$$

Using a minimal quantity of points (3 points) to obtain the convection term approxi-

mation as a linear algebra system we get a system of the form

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi)|_w^e \simeq \begin{pmatrix} a_{P1} & 0 & 0 \\ a_{W2} & a_{P2} & 0 \\ 0 & a_{W3} & a_{P3} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{pmatrix} + \begin{pmatrix} a_{W1} \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \phi_0 \\ 0 \\ \phi_4 \end{pmatrix}, \quad (\text{B.11})$$

B.6 Convection: QUICK Scheme

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi)|_w^e \simeq \rho_e u_e \left(-\frac{1}{8} \phi_W + \frac{6}{8} \phi_P + \frac{3}{8} \phi_E \right) + \rho_w u_w \left(\frac{6}{8} \phi_W + \frac{3}{8} \phi_P - \frac{1}{8} \phi_{WW} \right). \quad (\text{B.12})$$

Using a minimal quantity of points (4 points) to obtain the convection term approximation as a linear algebra system we get a system of the form

$$\underbrace{\int_V \vec{\nabla} \cdot (\rho \phi \vec{u}) dV}_{1D} = (\rho u \phi)|_w^e \simeq \begin{pmatrix} a_{P1} & a_{E1} & 0 & 0 \\ a_{W2} & a_{P2} & a_{E2} & 0 \\ a_{WW3} & a_{W3} & a_{P3} & a_{E3} \\ 0 & a_{WW4} & a_{W4} & a_{P4} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{pmatrix} + \begin{pmatrix} a_{W1} \\ a_{W2} \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \phi_0 \\ \phi_0 \\ 0 \\ \phi_4 \end{pmatrix}. \quad (\text{B.13})$$

B.7 Convection: Power Law Scheme

The complete discretization using power law scheme is given by the coefficients

$$a_E = \frac{\rho_e u_e}{1 - e^{Pe_e}}, \quad a_W = \left(\frac{\rho_w u_w}{1 - e^{Pe_w}} \right) e^{Pe_w}, \\ a_P = -(a_E + a_W) + (\rho_e u_e - \rho_w u_w). \quad (\text{B.14})$$

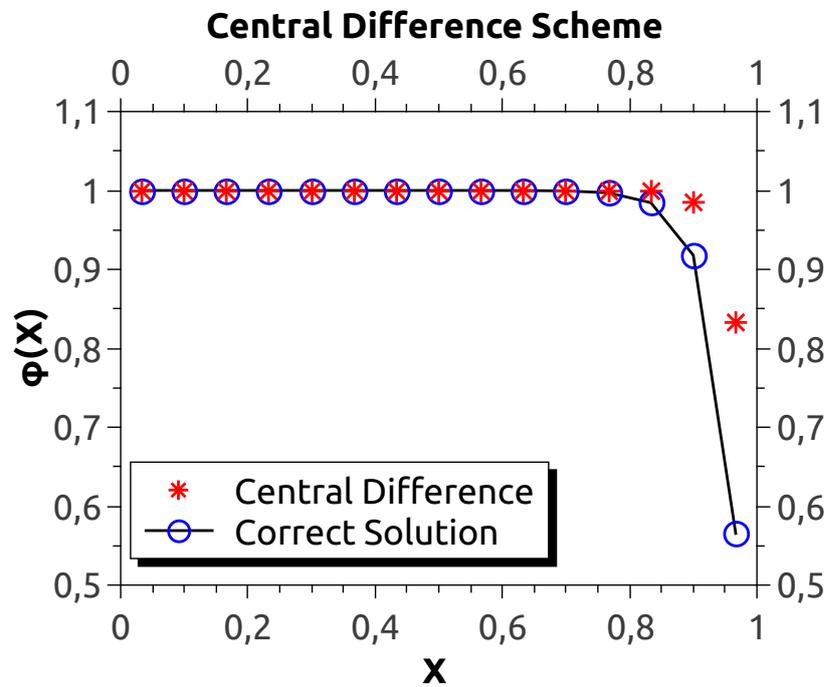
The matrix is obtained by coefficients above in the same fashion as the other schemes.

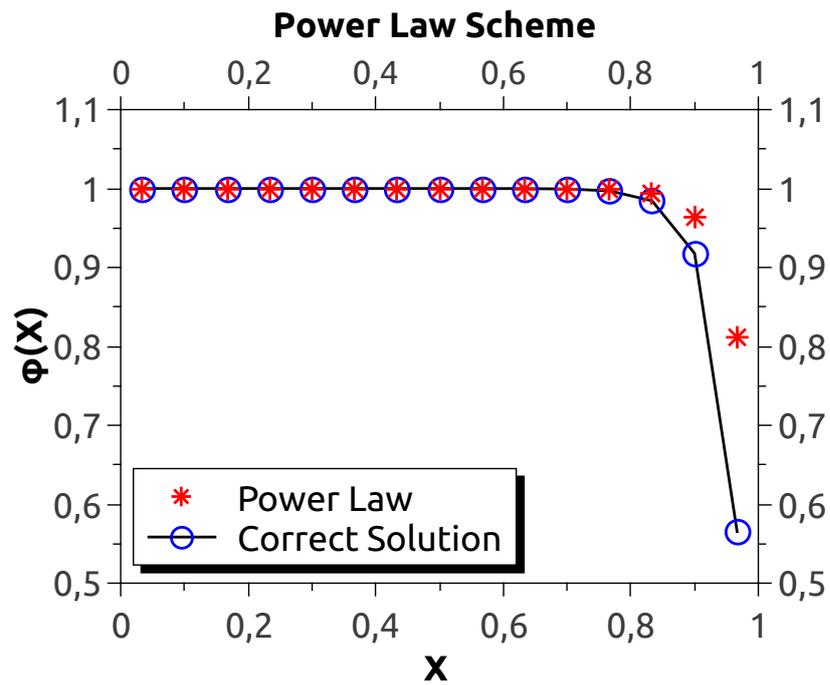
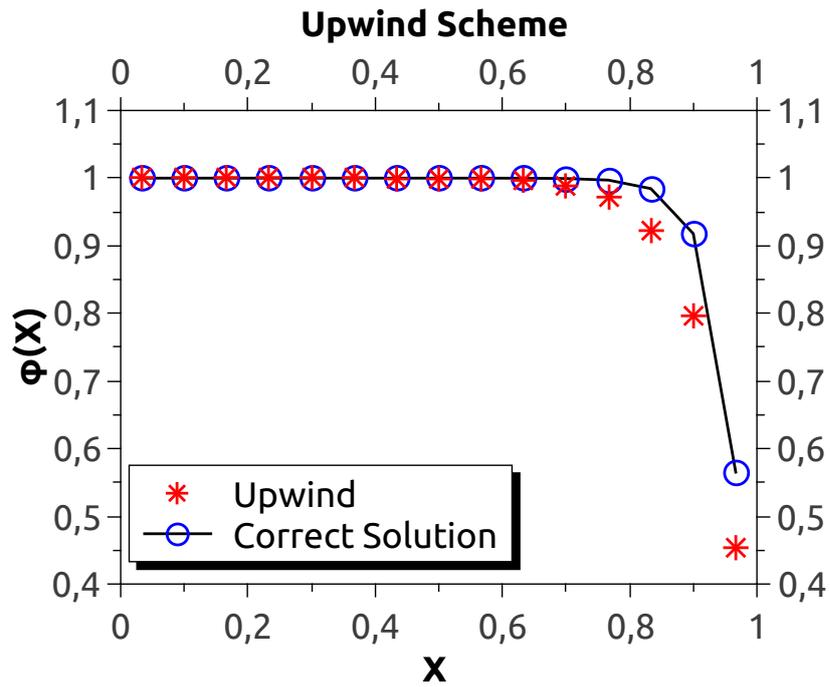
B.8 Numerical Experiment

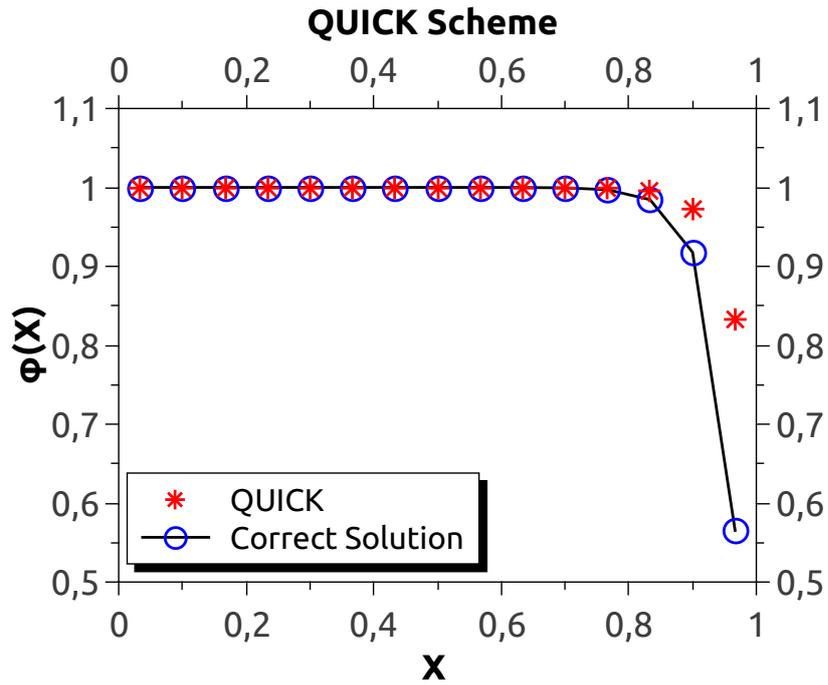
The schemes will be tested using a following problem:

$$\begin{aligned} \{\Gamma, n, u, \phi_0, \phi_L, L, \rho\} &= \{0.1, 15, 2.5, 1, 0, 1, 1\}, \\ \phi(x) &= \phi_0 + \frac{(\phi_L - \phi_0) \left(e^{\frac{\rho u x}{\Gamma}} - 1\right)}{\left(e^{\frac{\rho u L}{\Gamma}} - 1\right)}. \end{aligned} \quad (\text{B.15})$$

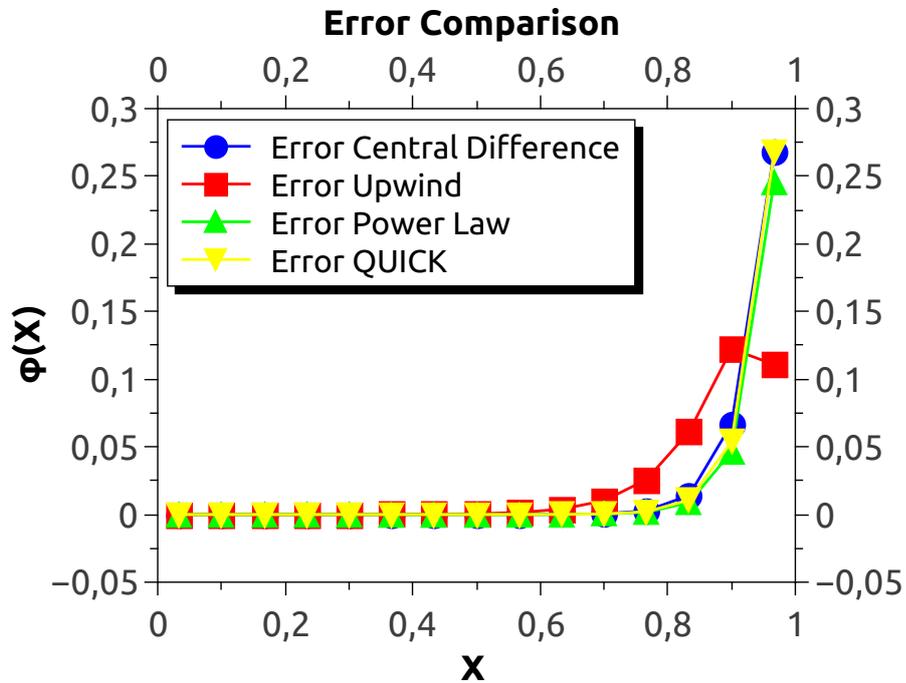
B.9 Results







B.10 Error Comparison Between Schemes



B.11 Concluding Remarks

The Power Law scheme fits best all points except the last point that is best fitted by Upwind scheme.

B.12 Acknowledgments

This work was financially supported by:

- **CNPQ** - Conselho Nacional de Desenvolvimento Científico e Tecnológico;
- **FCPC** - Fundação Cearense de Pesquisa e Cultura;
- **Petrobras** - Petróleo Brasileiro S. A.

REFERENCES

- [1] Michael Schafer. Computational engineering: introduction to numerical methods. Springer, 2006.
- [2] HK Versteeg and W Malalasekara. An introduction to computer fluid dynamics. Pearson Education Limited, England, 2007.

APPENDIX C - Heat Transfer In Heterogeneous Domain Using R

August 2014

Marceliano E. de Oliveira, Murilo P. de Almeida, Francisco L. de Oliveira

Laboratório de Simulação Numérica, Departamento de Física, Universidade Federal do
Ceará, Fortaleza, Ceará, Brasil

C.1 Abstract

The present work shows how to simulate heat transfer in heterogeneous domain using R (open-source software). The case studied is the heat transfer by diffusion in two dimensional domain; we use the Finite Volume Method (FVM) to implement the diffusion (heat) equation, with Central Difference Interpolation Scheme for spatial discretization and θ -Method (Explicit Euler, Crank Nicolson, Implicit Euler) for the time discretization. As a good programming practice, the routines are separated by specificities, and, with few modifications in these routines, the user can change the configuration of the problem and run different simulations. To block some points in the 2D domain, we create an operator that “freezes” the property value at these points along the time. The operator in question is constructed from a mask in matrix form (filled with values one or zero); when the value one is selected the point interacts with its neighbors, and, in other hand, the zero values cancels the interaction. To show how this can be done, we set some sample configurations and run the respective simulations.

C.2 Finite Volume Method

The Finite Volume Method is constructed from the integral form of the General Scalar Transport Equation using vector calculus theorems and identities. We see below the General Scalar Transport Equation,

$$\underbrace{\frac{\partial}{\partial t}(\rho\phi)}_{\text{Transient}} + \underbrace{\vec{\nabla} \cdot (\rho\phi\vec{u})}_{\text{Convection}} = \underbrace{\vec{\nabla} \cdot (\Gamma\vec{\nabla}\phi)}_{\text{Diffusion}} + \underbrace{S}_{\text{Sources}}. \quad (\text{C.1})$$

This equation is widely used when modeling the flow of mass, momentum, energy and other examples of the transport phenomena class.

C.3 Model Equations

The heat equation that governs heat changes is a well known equation, and appears generally in the form

$$\rho c_p \frac{\partial T}{\partial t} = \nabla \cdot (\lambda \nabla T). \quad (\text{C.2})$$

It is easy to see that an equivalent form of equation (C.2), which enables us to compare it with a general scalar transport equation is,

$$\frac{\partial T}{\partial t} = \nabla \cdot \left(\frac{\lambda}{\rho c_p} \nabla T \right), \quad (\text{C.3})$$

and after choosing $\rho \equiv 1$, $\Gamma \equiv \lambda/\rho c_p$, $\phi \equiv T$, we can see that equation (C.3) assumes the form

$$\frac{\partial \phi}{\partial t} = \nabla \cdot (\Gamma \nabla \phi). \quad (\text{C.4})$$

We can see that equation (C.4) is in the format of equation (C.1) with the absence of sources and convection terms.

C.4 Discretization Procedures

We will assume here the discretization using (FVM): in the first step lets integrate over the spatial volume the quantities that appear in equation (C.4), and then integrate them over time

$$\int_t \int_V \frac{\partial \phi}{\partial t} dV dt = \int_t \int_V \nabla \cdot (\Gamma \nabla \phi) dV dt = \int_t \int_A (\Gamma \nabla \phi) \cdot \hat{n} dA dt. \quad (\text{C.5})$$

We can see in equation (C.5) that Gauss divergence theorem was used in the right hand side, and in next steps the surface integral will be used to simplify the discretization.

To integrate in space, we will use the finite volume technique called **Central Difference Interpolation**, which gives a second order accurate approximation; the integration in time will be done using the θ -method. By choosing among three distinct values of θ , we are able to simulate with one of the three most frequently used techniques for time integration: $\theta = 0.0$ corresponds to **Forward Euler Method**; other option is $\theta = 0.5$ that results in the **Crank Nicolson Method**; and finally, $\theta = 1.0$ is the **Backward Euler Method**. After this previous considerations, our approximation becomes

$$\begin{aligned}
\frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= \delta y \left(\Gamma_e \frac{\phi_E^t - \phi_P^t}{\delta x} - \Gamma_w \frac{\phi_P^t - \phi_W^t}{\delta x} \right) (1 - \theta) + \\
&+ \delta x \left(\Gamma_n \frac{\phi_N^t - \phi_P^t}{\delta y} - \Gamma_s \frac{\phi_P^t - \phi_S^t}{\delta x} \right) (1 - \theta) + \\
&+ \delta y \left(\Gamma_e \frac{\phi_E^{t+1} - \phi_P^{t+1}}{\delta x} - \Gamma_w \frac{\phi_P^{t+1} - \phi_W^{t+1}}{\delta x} \right) \theta + \\
&+ \delta x \left(\Gamma_n \frac{\phi_N^{t+1} - \phi_P^{t+1}}{\delta y} - \Gamma_s \frac{\phi_P^{t+1} - \phi_S^{t+1}}{\delta x} \right) \theta. \tag{C.6}
\end{aligned}$$

Let's define the convenient notation for terms that multiply the ϕ_i values; this practice is useful in order to obtain the spatial discretization matrix. Then

$$\begin{aligned}
\frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= \left(\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} \phi_E^t + \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} \phi_W^t + \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} \phi_S^t + \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \phi_N^t \right) (1 - \theta) + \\
&+ \left(-\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} - \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} - \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} - \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \right) \phi_P^t (1 - \theta) + \\
&+ \left(\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} \phi_E^{t+1} + \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} \phi_W^{t+1} + \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} \phi_S^{t+1} + \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \phi_N^{t+1} \right) \theta + \\
&+ \left(-\underbrace{\frac{\delta y}{\delta x} \Gamma_e}_{a_E} - \underbrace{\frac{\delta y}{\delta x} \Gamma_w}_{a_W} - \underbrace{\frac{\delta x}{\delta y} \Gamma_s}_{a_S} - \underbrace{\frac{\delta x}{\delta y} \Gamma_n}_{a_N} \right) \phi_P^{t+1} \theta. \tag{C.7}
\end{aligned}$$

This enables us to change the format of equation (C.7) and write this in convenient

notation as

$$\begin{aligned} \frac{\delta x \delta y}{\delta t} (\phi_P^{t+1} - \phi_P^t) &= [a_E \phi_E^t + a_W \phi_W^t + a_S \phi_S^t + a_N \phi_N^t + a_P \phi_P^t] (1 - \theta) + \\ &+ [a_E \phi_E^{t+1} + a_W \phi_W^{t+1} + a_S \phi_S^{t+1} + a_N \phi_N^{t+1} + a_P \phi_P^{t+1}] \theta, \end{aligned} \quad (\text{C.8})$$

where the value a_P is defined by $a_P = -a_E - a_W - a_S - a_N$. The next step consists in selecting a number of points and their distribution over two dimensional domain. In this case, the property ϕ will be evaluated at nodal point located at the center of each cell. As an example follow a simple representation of this domain with 3 points in x direction and 3 points in y direction

ϕ_{31}	ϕ_{32}	ϕ_{33}
ϕ_{21}	ϕ_{22}	ϕ_{23}
ϕ_{11}	ϕ_{12}	ϕ_{13}

Table 6: Representation of 2D domain with property ϕ in center of each cell.

When we expand the discretization formula (C.8) at a generic point, the point is denoted the P coordinate and its neighbors are denoted W , E , S , and N , as highlighted by table (7)

ϕ_{31}	$\underbrace{\phi_{32}}_N$	ϕ_{33}
$\underbrace{\phi_{21}}_W$	$\underbrace{\phi_{22}}_P$	$\underbrace{\phi_{23}}_E$
ϕ_{11}	$\underbrace{\phi_{12}}_S$	ϕ_{13}

Table 7: Representation of point and your neighbors, useful to expand equation (C.8).

Observing the table (7) and equation (C.8), the discretization for ϕ_{22} becomes

$$\begin{aligned} \frac{\delta x \delta y}{\delta t} (\phi_{22}^{t+1} - \phi_{22}^t) &= [a_E^{23} \phi_{23}^t + a_W^{21} \phi_{21}^t + a_S^{12} \phi_{12}^t + a_N^{32} \phi_{32}^t + a_P^{22} \phi_{22}^t] (1 - \theta) + \\ &+ [a_E^{23} \phi_{23}^{t+1} + a_W^{21} \phi_{21}^{t+1} + a_S^{12} \phi_{12}^{t+1} + a_N^{32} \phi_{32}^{t+1} + a_P^{22} \phi_{22}^{t+1}] \theta. \end{aligned} \quad (\text{C.9})$$

If we repeat this process from ϕ_{11} to ϕ_{33} we will get a linear system with nine equations. We need to perform a different analysis on the borders of the domain because, for instance at ϕ_{11} , have no W neighbor, and so ϕ_{10} consists of a virtual value which is external to the domain matrix and acts like a source term. We can write the linear system in the form

$$\frac{\delta x \delta y}{\delta t} \left(\tilde{\phi}^{t+1} - \tilde{\phi}^t \right) = \left(\widetilde{M} \tilde{\phi}^t + \widetilde{N} \right) (1 - \theta) + \left(\widetilde{M} \tilde{\phi}^{t+1} + \widetilde{N} \right) \theta, \quad (\text{C.10})$$

$$\tilde{\phi}^{t+1} \equiv \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix}^{t+1}, \quad \tilde{\phi}^t \equiv \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix}^t, \quad \widetilde{N} \equiv \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \\ b_{21} \\ 0 \\ b_{23} \\ b_{31} \\ b_{32} \\ b_{33} \end{bmatrix}, \quad (\text{C.11})$$

where: $b_{11} \equiv a_S^{01} \phi_{01} + a_W^{10} \phi_{10}$, $b_{12} \equiv a_S^{02} \phi_{02}$, $b_{13} \equiv a_S^{03} \phi_{03} + a_E^{14} \phi_{14}$, $b_{21} \equiv a_W^{20} \phi_{20}$, $b_{23} \equiv a_E^{24} \phi_{24}$, $b_{31} \equiv a_N^{41} \phi_{41} + a_W^{30} \phi_{30}$, $b_{32} \equiv a_N^{42} \phi_{42}$, $b_{33} \equiv a_N^{43} \phi_{43} + a_E^{34} \phi_{34}$, and

$$\widetilde{M} \equiv \begin{bmatrix} a_P^{11} & a_E^{12} & 0 & a_N^{21} & 0 & 0 & 0 & 0 & 0 \\ a_W^{11} & a_P^{12} & a_E^{13} & 0 & a_N^{22} & 0 & 0 & 0 & 0 \\ 0 & a_W^{12} & a_P^{13} & 0 & 0 & a_N^{23} & 0 & 0 & 0 \\ a_S^{11} & 0 & 0 & a_P^{21} & a_E^{22} & 0 & a_N^{31} & 0 & 0 \\ 0 & a_S^{12} & 0 & a_W^{21} & a_P^{22} & a_E^{23} & 0 & a_N^{32} & 0 \\ 0 & 0 & a_S^{13} & 0 & a_W^{22} & a_P^{23} & 0 & 0 & a_N^{33} \\ 0 & 0 & 0 & a_S^{21} & 0 & 0 & a_P^{31} & a_E^{32} & 0 \\ 0 & 0 & 0 & 0 & a_S^{22} & 0 & a_W^{31} & a_P^{32} & a_E^{33} \\ 0 & 0 & 0 & 0 & 0 & a_S^{23} & 0 & a_W^{32} & a_P^{33} \end{bmatrix}, \quad (\text{C.12})$$

C.5 Solving The Linear System

As we can see, the matrix \widetilde{M} has a tetra-diagonal sparse form with null terms in some diagonals. To solve this linear system, an appropriate choice is to select the R Matrix package that has methods for solving linear sparse system; the sparse method of R Matrix package enhances the performance of the code. The system to be solved is an appropriate

arrange of equation (C.10),

$$\underbrace{\left[\frac{\delta x \delta y}{\delta t} \mathbb{1} - \theta \widetilde{M} \right]}_A \widetilde{\phi}^{t+1} = \underbrace{\left[\widetilde{M} (1 - \theta) + \frac{\delta x \delta y}{\delta t} \right]}_B \widetilde{\phi}^t + \widetilde{N},$$

$$\widetilde{A} \widetilde{\phi} = \widetilde{B}. \quad (\text{C.13})$$

The equation (C.13) is easily solved using sparse methods of the R Matrix package, and after the iterations we will be able to observe the time evolution of quantity ϕ .

C.6 Construction of Freeze Operator

The time derivative plays a central role on the construction of the Freeze operator (\widetilde{F}). By observing the left hand side (LHS) of equation (C.8), we see that the LHS is the time derivative in discrete form, and this implies that if we set the LHS equal to zero the property ϕ_P remains unchanged as time evolves; but to set the LHS equal zero we just need to multiply the RHS by zero. When dealing to a generic system equation we just need to create a matrix operator (\widetilde{F}) that acts this way: it makes null the RHS member of equation associated to the point P .

As an example, lets block ϕ_{22} of the 2D domain presented on table (6). To do this, lets first create a mask that sets the values 1 for unblocked points and 0 for the blocked point ϕ_{22} :

ϕ_{31}	ϕ_{32}	ϕ_{33}	\implies	1	1	1
ϕ_{21}	ϕ_{22}	ϕ_{23}		1	0	1
ϕ_{11}	ϕ_{12}	ϕ_{13}		1	1	1

Table 8: Representation of 2D domain and associated mask that block point ϕ_{22} .

The form that the vector $\widetilde{\phi}_P^t$ assumes in the system of equations can be used as a

guide to the first step to construct $\tilde{F}_{N \times 1}$ as

$$\tilde{\phi}^t \equiv \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix}^t \implies \tilde{F}_{N \times 1} \equiv \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \quad (\text{C.14})$$

For the square matrix \tilde{M} of the system of equations, we will create a square freeze operator. The simplest way to reach this goal consists in replicating the column vector $\tilde{F}_{N \text{points} \times 1}$ in all columns of a new empty square matrix. Then the $\tilde{F}_{N \text{points} \times N \text{points}}$ associated with \tilde{M} is

$$\tilde{F}_{N \times N} \equiv \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad (\text{C.15})$$

Finally, we need to apply the \tilde{F} operator to the system of equation. This operator will act only on the RHS making Hadamard's product (\circ). The transformation generated by \tilde{F} turns

$$\frac{\delta x \delta y}{\delta t} (\tilde{\phi}^{t+1} - \tilde{\phi}^t) = (\tilde{M} \tilde{\phi}^t + \tilde{N}) (1 - \theta) + (\tilde{M} \tilde{\phi}^{t+1} + \tilde{N}) \theta, \quad (\text{C.16})$$

into

$$\begin{aligned} \frac{\delta x \delta y}{\delta t} (\tilde{\phi}^{t+1} - \tilde{\phi}^t) &= (\tilde{F}_{N \times N} \circ \tilde{M} \tilde{\phi}^t + \tilde{F}_{N \times 1} \circ \tilde{N}) (1 - \theta) + \\ &+ (\tilde{F}_{N \times N} \circ \tilde{M} \tilde{\phi}^{t+1} + \tilde{F}_{N \times 1} \circ \tilde{N}) \theta. \end{aligned} \quad (\text{C.17})$$

And the final form of system to be solved, analogous to (C.13), is

$$\underbrace{\left[\frac{\delta x \delta y}{\delta t} \mathbb{1} - \theta \tilde{F}_{N \times N} \circ \tilde{M} \right]}_A \tilde{\phi}^{t+1} = \underbrace{\left[\tilde{F}_{N \times N} \circ \tilde{M} (1 - \theta) + \frac{\delta x \delta y}{\delta t} \mathbb{1} \right]}_B \tilde{\phi}^t + \tilde{F}_{N \times 1} \circ \tilde{N}. \quad (\text{C.18})$$

As we can see by comparing equation (C.18) to equation (C.13), the action of \tilde{F} operator does not imply a dramatic change in equation and, if a program that implements equation (C.13) already exists, the use of the \tilde{F} operator is a good option to blocking points in this program with minimal changes: 1 - Load a mask matrix; 2 - Construct $\tilde{F}_{N \times 1}$ and $\tilde{F}_{N \times N}$ from mask matrix; 3 - Multiply the \tilde{F} in same fashion that we do when transforming equation (C.16) in equation (C.17).

C.7 About Our Program

Our program was developed using the R language, and to run it extra packages must be added to the basic R installation: Matrix package and MASS package.

The variable names have extensive form to make it easy the description of variables and improving readability of all sub-routines codes. This coding practice could dispense with the use of comments; another code characteristic is that the naming of variables is direct compatible with the variable nomenclature used in the Finite Volume Method literature.

We use a folder structure mixed with shell scripts to make it easy to edit an run a simulation; when opening the program folder, we can observe one folder (with routines) and two scripts (EDIT.sh, RUN.sh), as follows

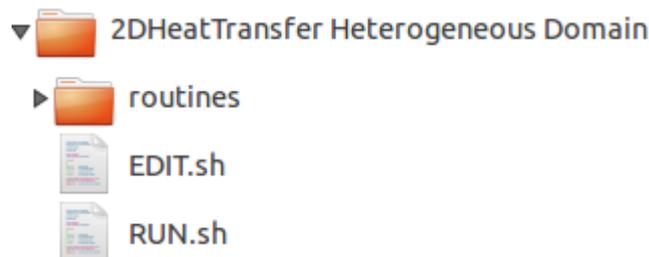


Figure 42: Folder structure of our program before execute EDIT.sh script.

The folder routines contains all routines to run a simulation. When executing EDIT.sh a new folder will be created and named results and all contents in folder routines will be copied to results; after this, an editor will open all routines located in the results folder. The routines in folder routines remain unchanged and protected and, when saving the an edited files only the codes in the results folder are changed. If we need to change a file in the folder routines, this can be made manually by editing it directly. After executing script EDIT.sh the program folder changes to

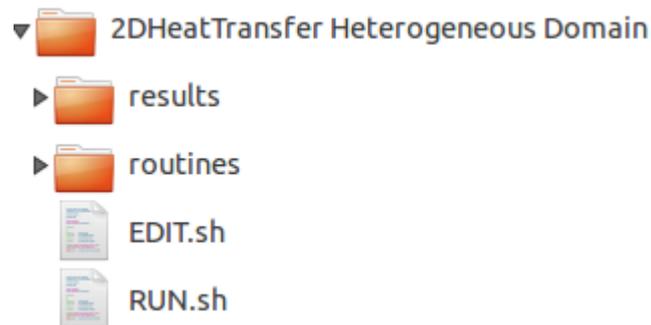


Figura 43: Folder structure of our program after execute EDIT.sh script.

The next step consists in executing the script RUN.sh that will run sequentially the routines in the folder results; all results obtained will be saved in the results folder together with other elements that will remain in results folder.

The contents in folder routines are

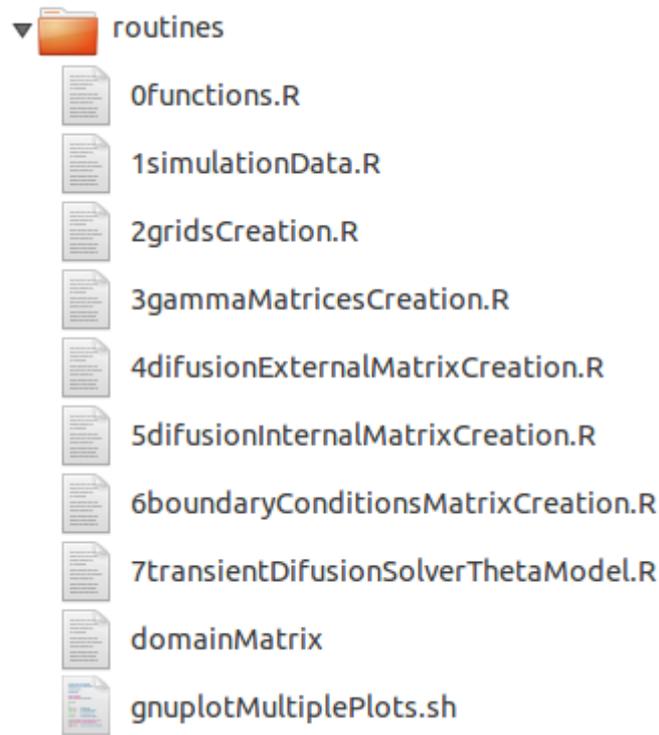


Figura 44: Contents of folder routines that will be used to solve problem.

All sub-routines are specialized and each creates an object that will be loaded from the solver, in the present case the routine `7transientDifusionSolverThetaModel.R`; Some defined functions that will be loaded and reused by other routines are in `0functions.R`. The parameters and constants of model can be adjusted in `1simulationData.R`. The geometrical grids are created by routine `2gridsCreation.R`. Routines (`3gammaMatricesCreation.R`)-(`6boundaryConditionsMatrixCreation.R`) create the matrices used in equation (C.18) and finally `7transientDifusionSolverThetaModel.R` loads matrices and solves (C.18). The domain mask will be loaded to solve system with blocked points; this mask is a text file named `domainMatrix` and can be created by any text editor or any other computer algebra system R, OCTAVE, MATLAB, etc. The supplementary script `gnuplotMultiplePlots.sh` can be used to easily plot the results using gnuplot.

C.8 Some Configurations

To run some examples here we will set some configurations in `1simulationData.R`

```

# Initial data: square domain
xLength=1
yLength=1
xInitial=0
yInitial=0
interval=1
tInitial=0
tFinal=20000
xFinal=xLength
yFinal=yLength
deltaZ=1

# Initial data: points and instants
nPointsX=50
nPointsY=50
nInstants=40000

# Initial data: material
density=1
heatConductivity=1.0e-5
heatCapacity=1
phiInitialAllPoints=200
phiInitialW=900
phiInitialE=900
phiInitialS=900
phiInitialN=900
Gamma=heatConductivity/(density*heatCapacity)

# ThetaModel parameter
# theta=(0,0.5,1)=(forward Euler, Cranck Nicolson, backward Euler)
theta=1.0

```

Figura 45: File 1simulationData.R with values used in case studies.

After editing the simulation data lets create 3 case studies with different blocked points, in other words 3 different domainMatrix files, one to each case. We will replicate the folder 2DHeatTransferHeterogeneousDomain 3 times and renaming each folder; a file containing 50×50 points is too large to be shown here, but we can show an image of these matrices using color white in position that contains 1 value and black for 0. In all the following cases the correspondence between column or row number and position can be consulted in the file named xGrid created by script RUN.sh; in this case $\{1, 10, 20, 30, 40, 50\} = \{0.01, 0.19, 0.39, 0.59, 0.79, 0.99\}$.

Domain Configurations of Case Studies

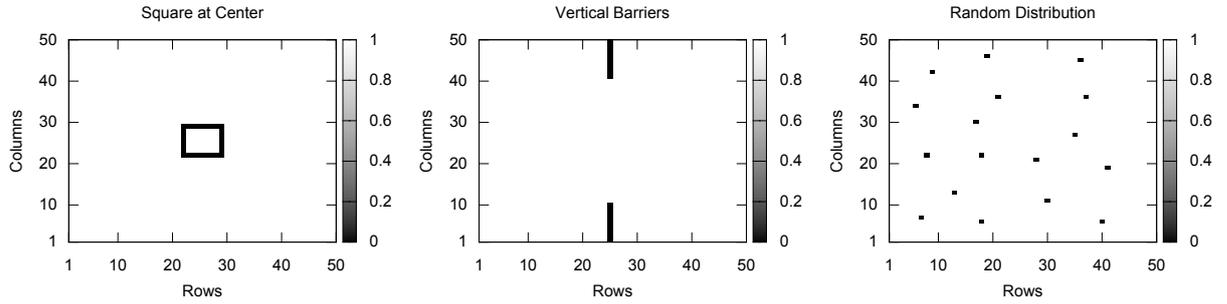


Figure 46: Representation of domains with blocked points generated plotting domainMatrix of each case.

Results of Square at Center Configuration

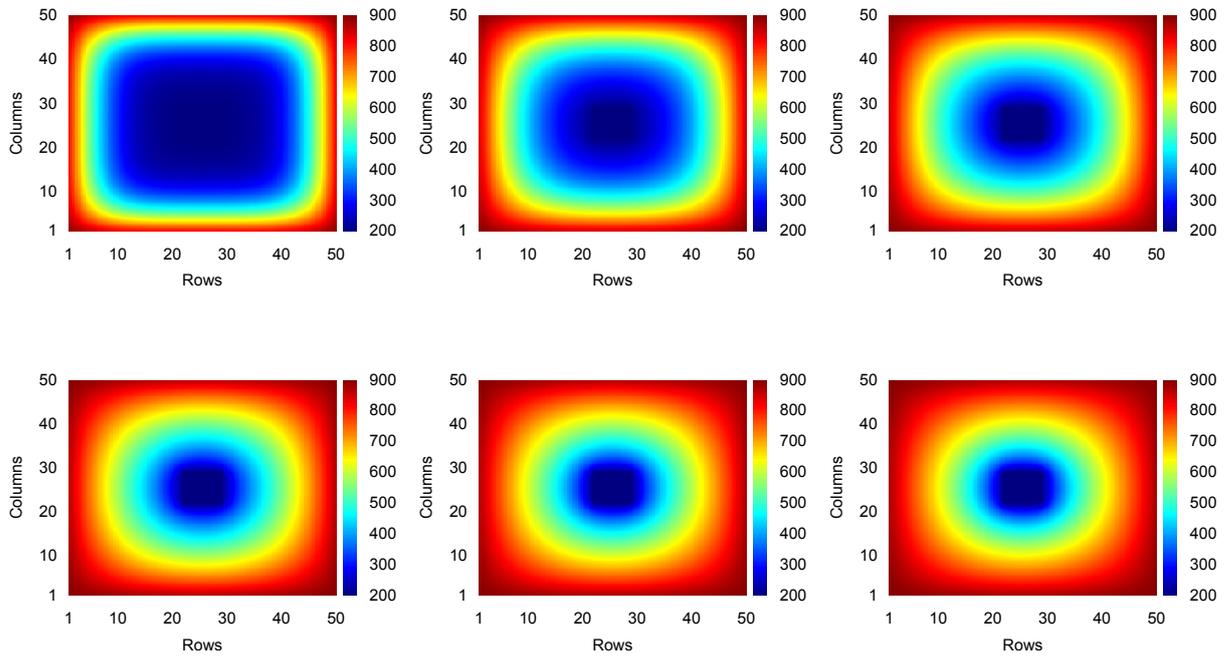


Figure 47: Evolution in time: sequence at instants $\{1k, 2k, 3k, 5k, 6k\}$.

Results of Vertical Barriers Configuration

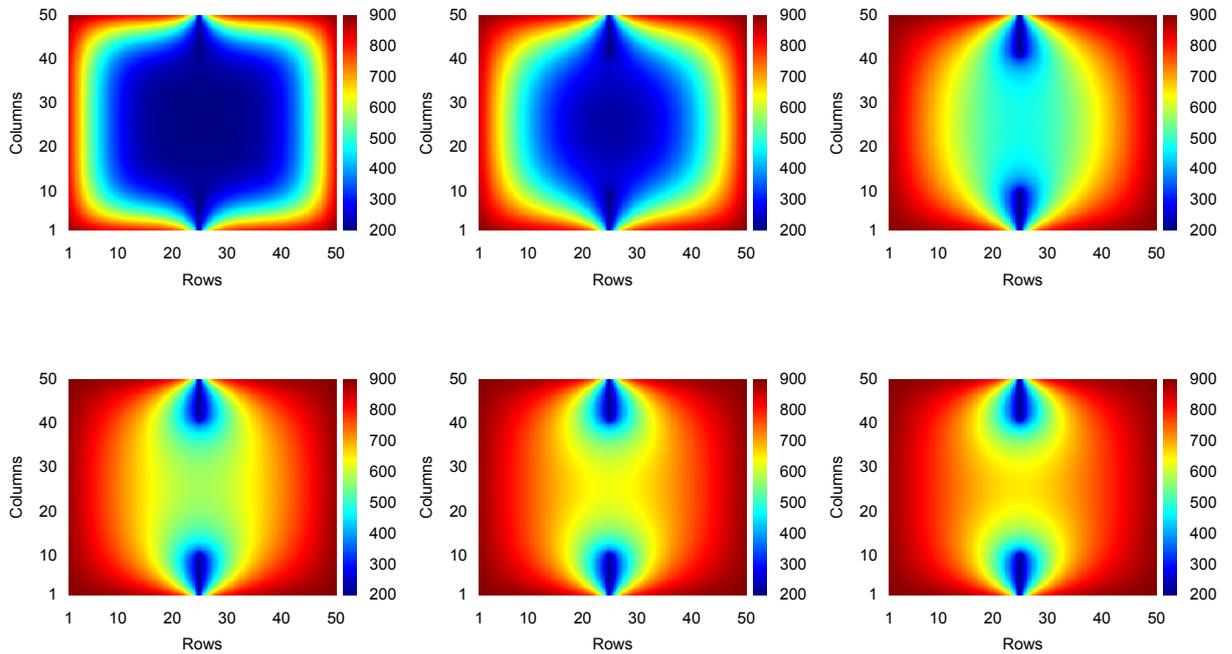


Figure 48: Evolution in time: sequence at instants $\{1k, 2k, 6.5k, 10k, 15k, 35k\}$.

Results of Vertical Barriers Configuration

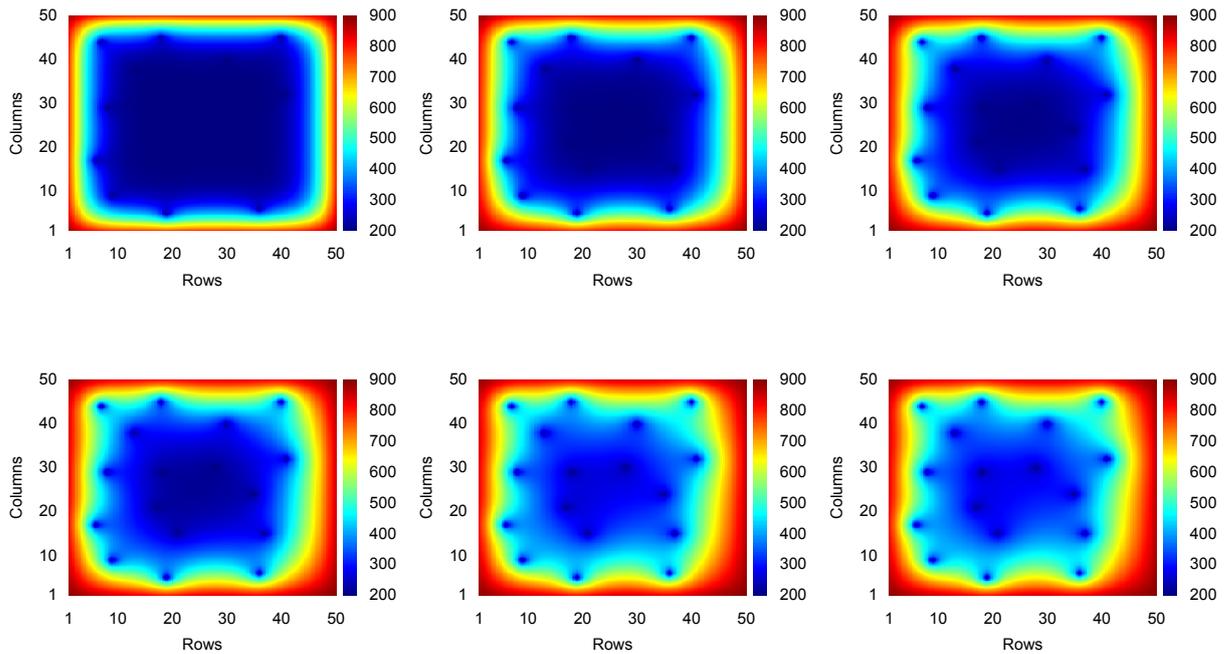


Figure 49: Evolution in time: sequence at instants $\{0.5k, 1k, 1.5k, 2k, 5k, 40k\}$.

C.9 Concluding Remarks

We build a computer program for simulating 2D diffusion problems using (FVM) and R program; to generate an irregular domain we show that an operator (\tilde{F}) constructed from a mask with 0 and 1 is a good option; this operator acts as a Hadamard's product with geometrical discretization. Our program is highly customizable and can be modified to solve another problem by changing only a few subparts.

C.10 Acknowledgments

This work was financially supported by:

- **CNPQ** - Conselho Nacional de Desenvolvimento Científico e Tecnológico;
- **FCPC** - Fundação Cearense de Pesquisa e Cultura;
- **Petrobras** - Petróleo Brasileiro S. A.

REFERENCES

- [1] Michael Schafer. Computational engineering: introduction to numerical methods. Springer, 2006.
- [2] HK Versteeg and W Malalasekara. An introduction to computer fluid dynamics. Pearson Education Limited, England, 2007.
- [3] Joel F. Ferziger and Milovan Péric. Computational methods for fluid dynamics. Springer, 2002.

APPENDIX D - Influence of the Uncertainty of Geological Parameters on Numerical Reservoir Simulators

August 2014

Murilo P. de Almeida, Marceliano E. de Oliveira

Laboratório de Simulação Numérica, Departamento de Física, Universidade Federal do
Ceará, Fortaleza, Ceará, Brasil

D.1 Abstract

We present the results of a series of numerical simulations of the flow of two immiscible incompressible fluids in a two-dimensional reservoir model of a 1/4-five spot well arrangement, with random spatial distribution of two permeability values. We test three types of configurations according to the position where the low permeability cells are placed: in the the first (injector) type these cells are around the injector well in the second (producer) they are around the producer well and in the third (middle) they are in the region between the two previous ones. From a statistical test of hypothesis analysis we conclude that the fluid flow is direction dependent and that the the time of breakthrough and the accumulated oil production are bigger in the producer configurations than in the injector. The set of middle type of configurations exhibits the highest fluctuations of the results.

D.2 Introduction

Fluid flow through porous media with low Reynolds number is generally modeled using Darcy's equation, which depends on parameters representing geological and fluid physical properties that are determined by laboratory experiments over sample material collected during drilling operation. Since the sample data is collected on a small number of sites, viz., drilled wells, it poorly represents the overall reservoir, which extends to regions of hundreds of meters and sometimes kilometers.

Numerical reservoir simulators are widely used in the petroleum industry because they enable the researchers and engineers to test hypothesis about the field properties and exploitation schemes in order to forecast production and devise optimized operation strategies. Even during the production period, numerical simulators are continuously deployed to guide operational decision making; at this stage the simulator parameters are frequently adjusted to better match history production data.

Uncertainty about the reservoir properties is unavoidable and one must deal with it in interpreting and using the results of numerical simulations. We perform a sensitivity analysis of a simple reservoir model by randomly distributing low permeability blocks over different regions of an otherwise homogeneous spatial domain in order to quantify the influence of data uncertainty over different parts of the reservoir. There are several physical (geological) parameters and response variables that could be used in the analysis, but in order to simplify the study we chose to random vary only the location of the lower permeability blocks (as will be described in the Model section) and to observe the breakthrough time and the water injection rate and the accumulated injected water, both measured at breakthrough moment. The model is initially filled with a fluid mixture composed by 80% of oil and 20% of water. We simulate the flow of the two immiscible fluids in this porous media model generated by the injection of a mixture of 15% of oil and 85% of water.

We are interested in quantifying the fluctuation of the response variables on a series of controlled simulations as a function of the regions where the random permeability occurs, i.e., where we lack complete geological information. From this we can determine the regions where to get the most detailed geological description in order to have a more precise numerical reservoir model.

D.3 Model Description

We use a two-dimensional (square) reservoir model with 100 m sides. Two wells are located at diagonally opposite corners, one used for the fluid injection, the *injector*, and the other, the *producer*, for fluid production. This square is divided into 400 square cells of $5 \times 5 m^2$. The porous media parameters (porosity and permeability) are time independent and constant within each cell; the permeability depends on the cell (spatial) position. The porosity is homogeneous and has value $\phi = 0.2$. The permeability of each cell is an isotropic two-dimensional tensor $\mathbf{K} = k\mathbf{I}$ whose eigenvalue k can take either of the values, $k = k_{\text{reg}}$ (regular permeability) or $k = k_{\text{low}}$ (low permeability). We partition the reservoir domain into three regions: one around the lower left corner, the other around the upper right corner and the third in the *middle* region between the two others. We construct three configuration classes according to the region where the low permeabilities are placed and where the injector and producer wells are located (see Figs. (54), (55) and (56)):

1. *Injector*: This class contains the configurations whose injection well is placed at the lower left corner and the low permeability cells are randomly distributed on the partition region around this corner. The producer well is located at the upper right corner;
2. *Producer*: This class contains the same random low permeability configurations as the injector class but the injector and the producer wells are switched: The injection occurs in the upper right corner and the production at the lower left corner;
3. *Middle*: The low permeability cells are randomly placed on the middle region, the injection occurs at the lower left corner and the production at the upper right corner.

Each class is further subdivided into 4 subclasses according to the probability of appearance of the low permeability cells. The probability (*fraction*) values used are 0.2, 0.3, 0.4 and 0.5.

A generic random configuration over region A with probability f , is generated by independently choosing the permeability of each of its cells, $i \in A$, according to the rule,

$$\begin{aligned} Pr\{k_i = k_{\text{low}}\} &= f \quad \text{and} \\ Pr\{k_i = k_{\text{reg}}\} &= 1 - f, \end{aligned}$$

and $k_i = k_{\text{reg}}$ if $i \notin A$. The entire reservoir domain is initially filled with a mixture of 80% of non-wetting fluid (oil) and 20% of wetting fluid (water).

D.4 Equations

The mathematical model of the two-fluid flow in porous media we use is based on mass conservation together with Darcy's law and a capillary relation.

Let the spatial domain be $\Omega = [0, 100] \times [0, 100]$. Let its boundary $\partial\Omega$ be composed by three disjoint subsets $\partial\Omega = \Gamma_N \cup \Gamma_i \cup \Gamma_o$, where Γ_N represents the no-flow boundary, Γ_i is the inflow boundary and Γ_o the outflow boundary. See Fig.(54)

Let the porosity be denoted by ϕ and the permeability tensor by \mathbf{K} . Let S_n and S_w denote the oil (non-wetting fluid) and water (wetting fluid) concentrations, respectively. The resulting equations are

$$-\nabla \cdot [(\lambda_n + \lambda_w) \mathbf{K} \nabla p_w] - \nabla \cdot \left[\lambda_n \frac{\partial p_c}{\partial S_w} \mathbf{K} \nabla S_w \right] = q_n + q_w \quad (\text{D.1})$$

$$\phi \frac{\partial S_w}{\partial t} - \nabla \cdot [\lambda_w \mathbf{K} \nabla p_w] = q_w, \quad (\text{D.2})$$

where p_n and p_w denote the respective fluid partial pressure, and q_n and q_w denote their source terms. The transmissibilities are denoted by

$$\lambda_\alpha = \frac{k_{r\alpha}}{\mu_\alpha}, \quad \alpha = n, w, \quad (\text{D.3})$$

where μ_α is the viscosity of fluid α and $k_{r\alpha} = k_{r\alpha}(S_\alpha)$ is the relative permeability of fluid α which is a function of the saturation S_α . We use the Brooks-Corey model for capillary pressure and relative permeability, (see Ref. [5]),

$$p_c(S_w) = p_n - p_w = p_b S_w^{-1/\lambda}, \quad (\text{D.4})$$

where p_b , the *bubbling pressure*, and λ , the *pore-size distribution index*, are characteristics constants to be determined experimentally. The relative permeabilities are given by

$$k_{rw}(S_w) = S_w^{(2+3\lambda)/\lambda} \quad (\text{D.5})$$

$$k_{rn}(S_w) = (1 - S_w)^2 (1 - S_w^{(2+\lambda)/\lambda}). \quad (\text{D.6})$$

In our simulations we take $\lambda = 2$.

D.5 Boundary Conditions

The boundary of the spatial domain is partitioned into three non-overlapping subsets, where we imposed the boundary conditions (see Fig. (54)):

1. On the no-flow boundary, Γ_N : $\lambda_w \mathbf{K} \nabla p_w \cdot \vec{n} = 0$ and $\lambda_n \mathbf{K} \nabla p_n \cdot \vec{n} = 0$;
2. On the inlet, Γ_i : $p_w = P_{in}$ and $S_w = 0.85$;
3. On the outlet, Γ_o : $p_w = P_{out}$ and $\lambda_n \mathbf{K} \nabla p_c \cdot \vec{n} = 0$.

D.6 Solver

We numerically solve Eqs. (D.1) and (D.2) using the OpenFoam free-software [3]. The details of the equations implementation into OpenFoam will be presented elsewhere [18]. We use an IMPES (IMplicit Pressure and EXplicit Saturation) scheme: Knowing the initial saturation field S_w we solve Eq. (D.1) for p_w , which is then used in Eq. (D.2) to advance S_w in time using an explicit Euler scheme. These steps are repeated until the final simulation time is reached.

D.7 Simulations

As described in the Model section, we perform simulations with configurations resulting from the combination of two factors: the region and the probability of occupation of low permeability cells, with three regions (injector, middle and producer) and four probabilities (0.2, 0.3, 0.4 and 0.5). For each combination region-probability we generate 1000 random configurations and simulate the flow with the corresponding boundary conditions using OpenFoam. We fix the time step $\Delta t = 0.1$ and the final simulation time at $t = 3000$. At each time step we evaluate the water saturation S_w , the partial water pressure p_w at each cell and compute the effective velocities \vec{U}_w and \vec{U}_n on the cell boundaries. We record in an output (history) file, at each time step, the current simulation time, the water inflow rate, the oil inflow rate, the water outflow rate and the oil outflow rate, as well as the total injection and production of water and oil. By analyzing these *history files* we observed that the breakthrough time, which is the time when the water front reaches the producer well can be characterized (in the our simulations) by the water production rate, w_prod_rate , crossing the (critical) value 0.02. A typical curve of water

production rate versus time is seen in Fig. (50). At a post-simulation processing we spot the breakthrough moment at each history file and mount a (data frame) file containing the information of the configuration (its sequence number, *region* and *probability*), the breakthrough time, the oil and water injection and production rates and the accumulated injected and produced water and oil. To analyze the data we use the free-software R [20].

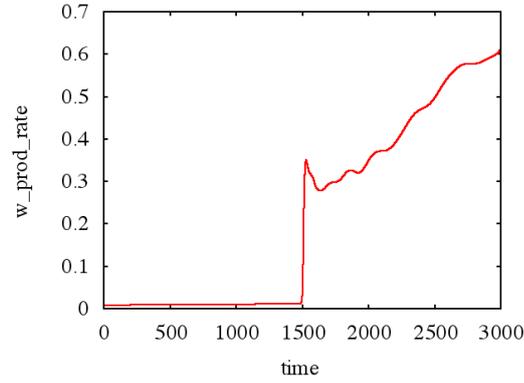


Figure 50: Typical curve of the water production rate versus time. At the breakthrough time the curve undergoes a rapid increase, crossing in this case the value 0.02.

D.8 Parameters

The parameter values used in the simulations are:

$$\mu_n = 10^{-2} \text{ kg m}^{-1} \text{ s}^{-1}$$

$$\mu_w = 10^{-3} \text{ kg m}^{-1} \text{ s}^{-1}$$

$$\lambda = 2.0$$

$$p_b = 10^3 \text{ Pa}$$

$$k_{\text{reg}} = 10^{-8} \text{ m}^2$$

$$k_{\text{low}} = 10^{-11} \text{ m}^2$$

The boundary conditions are:

$$p_w = \begin{cases} 3.0 \times 10^6 \text{ Pa} & \text{on the inflow} \\ 1.0 \times 10^6 \text{ Pa} & \text{on the outflow} \end{cases}$$

$$s_w = 0.85 \quad \text{on the inflow}$$

$$\nabla s_w = 0 \quad \text{on the outflow}$$

D.9 Results and Discussion

Number of blocks. The permeability configurations are randomly generated according to an occupation probability over a determined region of the reservoir domain. For each combination of region and probability we random generate 999 configurations. The numerical simulation we some of them fail to converge with time step $\Delta t = 0.1$, and so, they are discarded. The number of valid configurations in each category is reported in Table (9). From this table we can already detect the system dependence on the direction of the flow. The injector and producer set have common permeability configurations, but with some of them the numerical simulation converge in one direction but not in the other, as can be seen comparing the contents of the first and third rows of Table (9). All of the 999 generated configurations are valid in the injector set with probabilities 0.2, 0.3 and 0.4, and only one configuration is discarded with probability 0.5. On the other hand, in the producer set, we have to discard 2 configurations with probability 0.2, 17 with probability 0.3, 40 with probability 0.4 and 86 with probability 0.5. This also shows that low permeability regions close to the producer well may result in stiffer models which are highly computational demanding.

The number of blocks in each valid configuration is therefore a random number whose mean and standard deviation depend on the region and on probability as shown in Tables (10) and (11). The maximum number of blocks in the injector (producer) region is 59 and in the middle region is 242. Dividing the number of blocks n_blocks actually occupied by the low permeability cells by the total number of blocks of the region we get the ratio of occupation of the configuration whose averages and standard deviations are presented in Tables (12) and (13).

fraction	injector	middle	producer
0.2	999	999	997
0.3	999	999	982
0.4	999	998	959
0.5	998	994	913

Table 9: Number of configurations leading to converging simulation with $\Delta t = 0.1$, as a function of the region and of the occupancy probability

fraction	injector	middle	producer
0.2	11.94	48.59	11.93
0.3	17.81	72.89	17.78
0.4	23.73	96.33	23.66
0.5	29.49	120.68	29.38

Table 10: Average of the number of blocks with low permeability as a function of the region and of the occupancy probability

fraction	injector	middle	producer
0.2	3.07	6.19	3.06
0.3	3.65	7.18	3.65
0.4	3.70	7.58	3.68
0.5	3.91	7.62	3.90

Table 11: Standard deviation of the number of low permeability blocks as a function of the region and of the occupancy probability

fraction	injector	middle	producer
0.2	0.2023040	0.2007834	0.2021828
0.3	0.3019121	0.3011896	0.3013739
0.4	0.4022158	0.3980647	0.4010710
0.5	0.4997792	0.4986739	0.4979115

Table 12: Average of the rate of occupation by low permeability cells as a function of the region and of the occupancy probability

D.10 Selection of variables.

In order to detect the correlated variables and reduce the number of analysis to be performed we use two scattered plots:

1. the first is presented in Fig. (57) and contains the plots of all possible combinations of two variables taken from the set composed by region, probability, time of breakthrough (t_{bt}), water injection rate, oil injection rate, water production rate and oil production rate;
2. the second is presented in Fig. (58) and contains the plots of two variable combinations taken from the set composed by region, probability, and the values at breakthrough time of the accumulated injected and produced water and oil.

fraction	injector	middle	producer
0.2	0.05201688	0.02559131	0.05193054
0.3	0.06194378	0.02967523	0.06192722
0.4	0.06271174	0.03131553	0.06244157
0.5	0.06631703	0.03147741	0.06614187

Table 13: Standard deviation of the rate of occupation by low permeability cells as a function of the region and of the occupancy probability

In these scattered plots the variable *region* takes on numerical values

$$region = \begin{cases} 1 & \equiv \textit{injector} \\ 2 & \equiv \textit{middle} \\ 3 & \equiv \textit{producer} \end{cases} \quad (\text{D.7})$$

Let us examine the first set of plots, Fig. (57). The plots on the third row of this figure shows the behavior of the breakthrough time (t_bt) versus the other variables. We can see that t_bt takes value over intervals that depend both on the region and on the probability (first and second columns, respectively). The same happens with the water injection rate (w_inj_rate), as can be seen on the first and second columns of the fourth row of plots. On the fifth row, we see that the oil injection rate (o_inj_rate) takes value over a very narrow interval, $-6.0 \times 10^{-4} \leq o_inj_rate \leq -3.0 \times 10^{-4}$, which can be practically considered as a constant value with very small random fluctuations, which can result, for instance, from roundoff effects. On the sixth row, we observe that the water production rate, which is used to specify the breakthrough time, has most of its values about 0.02 and exhibits three higher values, which are produced at three isolated simulations whose characteristics are shown in Table (14) below. Excluding these three points, the remaining values of the water production rate are within 0.02 and 0.0204.

region	<i>fraction</i>	<i>index</i>	t_bt	w_prod_rate
producer	0.50	100	1721.10	0.0242
producer	0.50	642	1632.10	0.0316
producer	0.50	798	2904.30	0.0460

Table 14: Spurious configurations with w_prod_rate greater than 0.0204

We identify a strong correlation between the water injection rate and oil production rate, which is evidenced by the rectilinear graph on the plots at the seventh row and fourth column (and fourth row and seventh column). From these observations, we conclude that we must study the fluctuations of the time of breakthrough and water injection rate and neglect the other variables (oil injection rate, water production rate and oil production

rate) since they are either constant or correlated to the water injection rate.

On Fig. (58) we observe that the accumulated quantities of injected and produced, water and oil, are all strongly correlated among themselves. Therefore, we need to study only one of these variables, for instance the accumulated injected water.

We conclude that the time of breakthrough, the water injection rate and the accumulated injected water may be used to study the fluctuations in the reservoir response to the variations of the region and probability variables. The scattered plots of these variables are presented in Fig. (59)

D.11 Time of Breakthrough.

We compute the mean and standard deviation values of the breakthrough time for the various configurations. The effect of the flow direction over the results is evidenced by computing the difference of the breakthrough time of the producer and injector configuration with the same permeability configuration. The mean and standard deviation values of this difference, for each fraction value, is shown in Tables (22) and (23). From the related plots and histograms (Fig. (53), and Fig. (60)) we conclude that:

1. The breakthrough time, t_{bt} , increases with the amount of low permeability cells (probability);
2. Within the same probability configurations, the mean value of t_{bt} gets bigger as the the region of obstacles approaches the producer well;
3. The standard deviation values are bigger when the low permeability cells occupy the middle region; this might result from the bigger number cells than in the other two configuration types (injector and producer);
4. The histograms become more spread as the probability increases;
5. The histograms of the middle configurations are flatter than those of the injector and producer type;
6. The histograms are asymmetric, with longer tails towards high values of t_{bt} .

D.12 Water Injection Rate

Injected water and its rate are expressed by negative numbers since they are related to the flow of the effective water velocity through the boundary, and the flow is negative when the angle between the velocity and the outward boundary normal is greater than $\pi/2$, i.e., when water enters the domain of the reservoir. For the sake of clarity, we use the absolute value of these quantities in the following tables and figures.

From the average and standard deviation values of the water injection rate for the various configurations, and from the related plots and histograms (Tables (15) and (16), and Fig. (51)) we conclude that:

1. The (absolute value of the) water injection rate at breakthrough time, w_inj_rate , decreases with the increase of of low permeability cells (probability);
2. Within the same probability configurations, the (absolute value of the) mean value of w_inj_rate is higher bigger as the the region of obstacles approaches the injector well;
3. The (absolute value of the) mean of w_inj_rate decreases drastically as we increase the probability of low permeability cells in the regions closer to the producer well; the disparity of values between the injector and the producer is higher for higher probability values;
4. The standard deviation values increase as the region moves from injector to producer, and as the probability value increases;
5. The histograms become more spread as the probability varies from 0.2 to 0.5 and as the region of low permeability moves from injector to producer;
6. The histograms are asymmetric, with longer tails towards smaller absolute value of the water injection rate, w_inj_rate .

fraction	injector	middle	producer
0.2	0.45	0.44	0.45
0.3	0.45	0.42	0.43
0.4	0.44	0.40	0.41
0.5	0.43	0.37	0.38

Table 15: Averages of the (absolute value of) water injection rate (at breakthrough time) as a function of the region and of the probability variables.

fraction	injector	middle	producer
0.2	0.01	0.01	0.01
0.3	0.01	0.01	0.02
0.4	0.01	0.02	0.02
0.5	0.02	0.02	0.04

Table 16: Standard deviation values of the water injection rate at breakthrough time as a function of the region and of the probability variables.

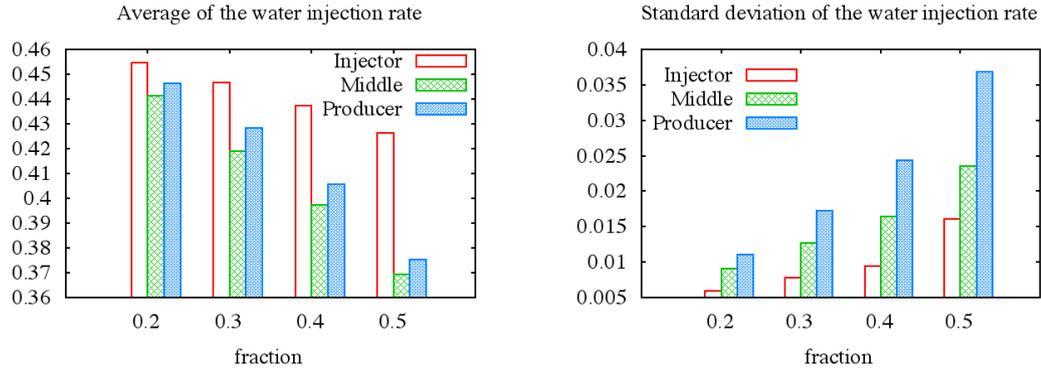


Figure 51: Plots of averages of absolute values (left) and standard deviations (right) of the water injection rate for the various probabilities and positioning of the low permeability cells.

D.13 Accumulated Injected water

The negative values of accumulated injected water mean that the injected water results from a negative flow into the reservoir domain. From the mean and standard deviation values of accumulated injected water for the various configurations, and from the related plots and histograms (Tables (17) and (18), and Fig. (52)) we conclude that:

1. The (absolute value of the) accumulated injected water injection rate at breakthrough time, w_{inj} , increases with the increase of low permeability cells (probability) for the producer configurations;
2. For the injector configurations the absolute value of the injected water remain almost for all the probabilities, with a small reduction as the probability varies from 0.2 to 0.5.
3. The standard deviation values are bigger for the middle type configurations with all probability values; the values related to the injector are slightly bigger than those of the producer configurations;
4. The histograms become more spread as the probability varies from 0.2 to 0.5 and

as the region of low permeability moves from injector to producer;

5. The histograms are more symmetric than for the other variables (t_bt and w_inj_rate); the smaller asymmetry shows with longer tails towards bigger absolute value of the accumulated injected water, w_inj .

fraction	injector	middle	producer
0.2	605.73	599.88	611.76
0.3	606.68	601.03	615.45
0.4	602.71	606.85	618.84
0.5	601.19	605.23	620.44

Table 17: Averages of the absolute values of the total (accumulated) injected water (at breakthrough time) as a function of the region and of the probability variables.

fraction	injector	middle	producer
0.2	27.16	28.34	21.48
0.3	37.93	52.00	30.87
0.4	48.05	71.48	42.47
0.5	58.01	78.50	52.17

Table 18: Standard deviation values of the total (accumulated) injected water (at breakthrough time) as a function of the region and of the probability variables.

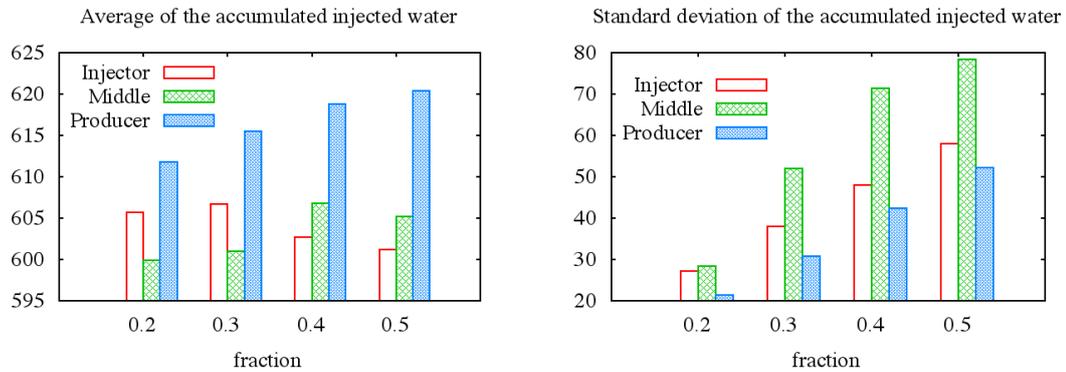


Figure 52: Plots of averages of absolute values (left) and standard deviations (right) of the accumulated injected water for the various probabilities and positioning of the low permeability cells.

D.14 Hypothesis tests

We perform the two-sided Student's t-test to quantify the difference between the mean of the observed functions in the *injector* and *producer* type configurations. We use the significance level $\alpha = 0.01\%$. The output of the tests are presented in the Tables (19),

(20), (21), which contain for each *fraction* value the number of degrees of freedom (d.f), the critical value t_c , the computed t value of the samples, the probability p and the 99.99% interval of confidence for the difference $\mu_{inj}(X) - \mu_{prod}(X)$, where $\mu_{inj}(X)$ ($\mu_{prod}(X)$) is the mean of the variable X of the *injector* (*producer*) set of configurations.

fraction	d.f.	t_c	t	p	interval
0.2	1983	-3.898521	-15.76	2.2e-16	(-54.68, -32.99)
0.3	1977	-3.898545	-18.11	2.2e-16	(-94.26, -60.87)
0.4	1862	-3.899037	-24.64	2.2e-16	(-170.32, -123.79)
0.5	1540	-3.900807	-28.16	2.2e-16	(-290.79, -220.03)

Table 19: Output values of the two-sided Student's t-test applied to the compare the mean of the breakthrough time of the injector and of the producer configurations.

fraction	d.f.	t_c	t	p	interval
0.2	1531	-3.900868	20.34	2.2e-16	(0.0065, 0.0099)
0.3	1364	-3.902129	29.95	2.2e-16	(0.0157, 0.0204)
0.4	1226	-3.903432	37.76	2.2e-16	(0.0285, 0.0351)
0.5	1221	-3.903484	38.69	2.2e-16	(0.0460, 0.0563)

Table 20: Output values of the two-sided Student's t-test applied to the compare the mean of the water injection rate (at breakthrough time) of the injector and of the producer configurations.

fraction	d.f.	t_c	t	p	interval
0.2	1895	-3.89889	-5.507	4.149e-08	(-10.307894, -1.762291)
0.3	1912	-3.898816	-5.6511	1.834e-08	(-14.828397, -2.720762)
0.4	1942	-3.898689	-7.8763	5.562e-15	(-24.108657, -8.143985)
0.5	1908	-3.898833	-7.6375	3.479e-14	(-29.077354, -9.423398)

Table 21: Output values of the two-sided Student's t-test applied to the compare the mean of the total injected water (at breakthrough time) of the injector and of the producer configurations.

D.15 Total produced oil

Although not necessary in the previous sections, we analyze here the total produced oil at breakthrough time, one of the most important factors considered in the management of a reservoir since it express the total amount of oil that can be recovered with water injection. This variable can be used as a performance measure of the various configurations considered.

We observe that, on average, the total oil production up to breakthrough time is higher when the producer well is in the region of low permeabilities and that the higher

the fraction of low permeabilities cells the higher is the amount of total oil produced, i.e., the reservoir is better swept when the producer well is surrounded by low permeabilities regions, which are obstacles for the flow. On the other hand, if the injector is in region with low permeabilities, the total oil production decreases as the *fraction* of low permeability cells increases.

The configurations with higher fluctuation on the values are those where the low permeabilities are in the middle region as can be seen in Fig. (53). The biggest and the smallest values of the total oil produced among all simulations were attained by type *middle* configurations with *fraction* = 0.5. The biggest value was 853.505 and the smallest was 399.439.

fraction	injector	middle	producer
0.2	590.79	585.08	596.68
0.3	591.71	586.18	600.27
0.4	587.84	591.84	603.57
0.5	586.34	590.25	605.13

Table 22: Average of the total produced oil at breakthrough time as a function of fraction and region

fraction	injector	middle	producer
0.2	26.50	27.65	20.95
0.3	37.01	50.73	30.12
0.4	46.89	69.75	41.44
0.5	56.60	76.60	50.91

Table 23: Standard deviation of the total produced oil at breakthrough time as a function of fraction and region

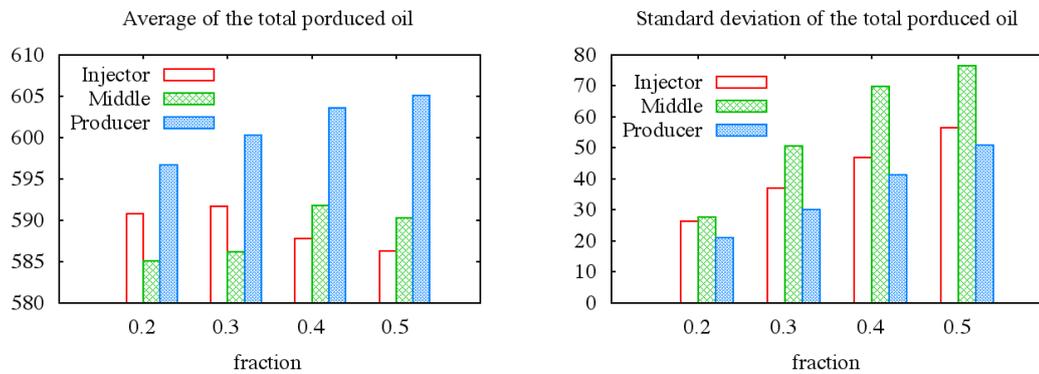


Figure 53: Plots of averages (left) and standard deviations (right) of the total produced oil for the various probabilities and positioning of the low permeability cells.

D.16 Conclusions

From this series of numerical simulations of two immiscible fluid flow in a two-dimensional porous media, we can conclude that the existence of low permeability regions (obstacles to the flow) interferes with the numerical results, which can be observed in a set of response functions, from which we chose the time of breakthrough (t_{bt}) and the values at breakthrough time of the water injection rate (w_{inj_rate}) and of the total injected water (w_{inj}).

1. The results depend on the flow direction: *injector* and *producer* type of configurations produced significantly different responses.
2. Average responses show that the *producer* configurations have larger values of the breakthrough time and of the total produced oil than the *injector* configurations. This suggests that the water might be injected in regions of better permeability in order to get a better sweep of the reservoir.
3. The highest variability of the breakthrough time, and the total injected water and produced oil, occurred within the *middle* configuration sets.

fraction	injector	middle	producer
0.2	2.69	3.26	2.25
0.3	5.68	11.79	5.54
0.4	8.74	23.29	11.62
0.5	13.44	33.50	27.66

Table 24: Spread of the t_{bt} as a function of region and probability.

fraction	injector	middle	producer
0.2	23.92	21.87	26.57
0.3	16.65	11.75	17.28
0.4	13.58	8.65	12.30
0.5	11.20	7.62	8.36

Table 25: Signal-to-noise ratio of the breakthrough time t_{bt} as a function of the region and probability

D.16.1 Figures

fraction	injector	middle	producer
0.2	7.82014×10^{-5}	1.846986×10^{-4}	2.727221×10^{-4}
0.3	1.376279×10^{-4}	3.830502×10^{-4}	6.944751×10^{-4}
0.4	2.012878×10^{-4}	6.765368×10^{-4}	1.4633205×10^{-3}
0.5	6.029076×10^{-4}	1.505675×10^{-3}	3.6107259×10^{-3}

Table 26: Spread of the w_inj_rate as a function of region and probability.

fraction	injector	middle	producer
0.2	76.24	48.89	40.46
0.3	56.96	33.06	24.83
0.4	46.61	24.22	16.63
0.5	26.58	15.64	10.17

Table 27: Signal-to-noise ratio of the water injection rate at breakthrough time w_inj_rate as a function of the region and probability

fraction	injector	middle	producer
0.2	1.22	1.34	0.75
0.3	2.37	4.48	1.55
0.4	3.82	8.36	2.91
0.5	5.57	10.10	4.37

Table 28: Spread of the w_inj as a function of region and probability.

fraction	injector	middle	producer
0.2	22.30	21.17	28.48
0.3	15.99	11.56	19.94
0.4	12.54	8.49	14.57
0.5	10.36	7.71	11.89

Table 29: Signal-to-noise ratio of the accumulated injected water at breakthrough time w_inj as a function of the region and probability

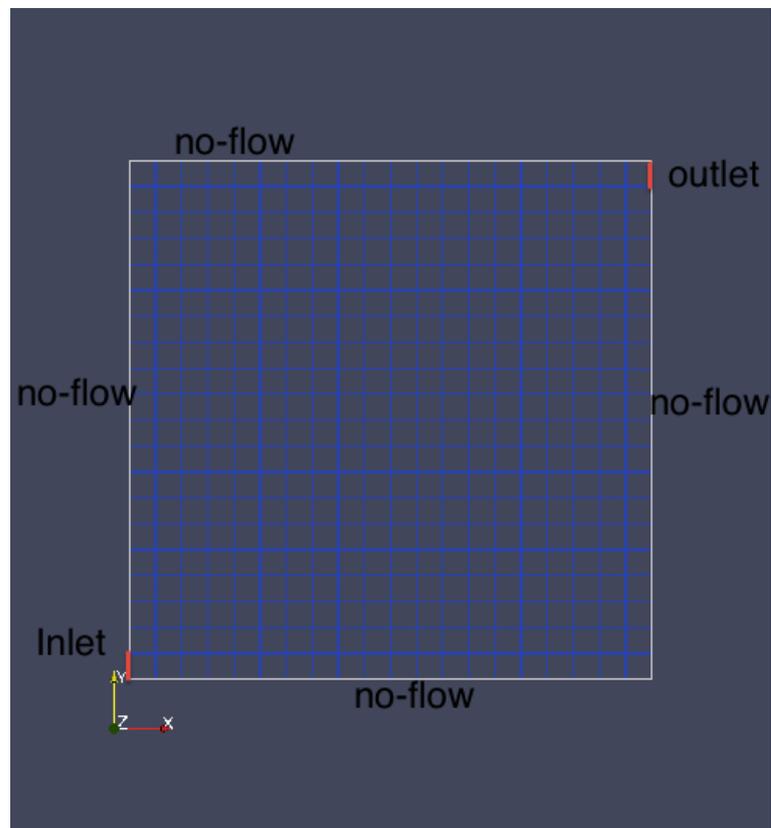


Figure 54: Mesh of the spatial domain of the numerical model.

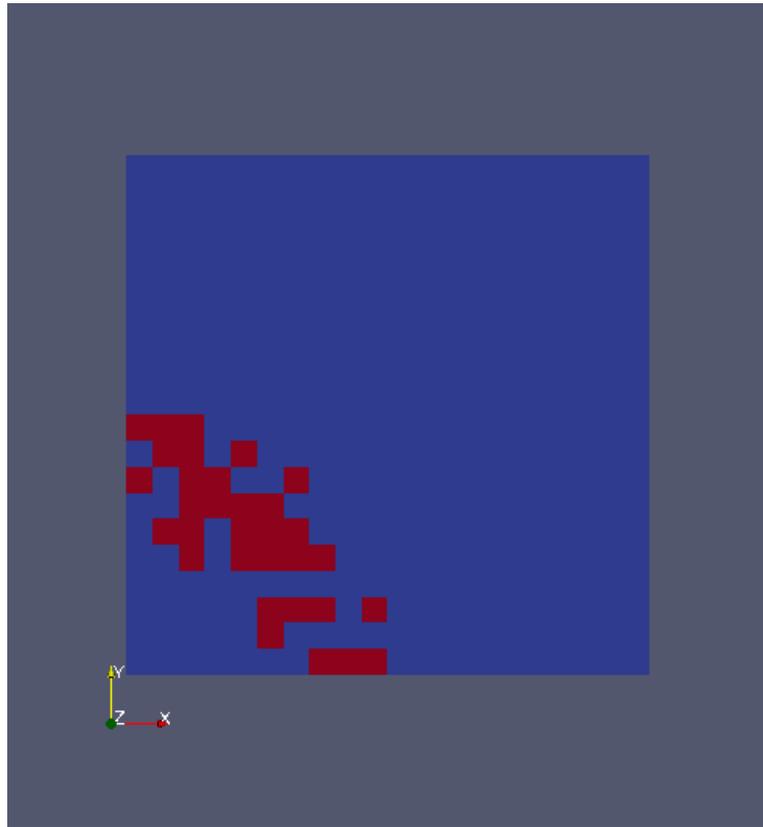


Figure 55: A sample configuration showing in dark the cells with lower permeability in a quarter-circle region of radius 50 m around the lower left corner. This type of permeability distribution is used in the *injector* and *producer* classes of configurations.

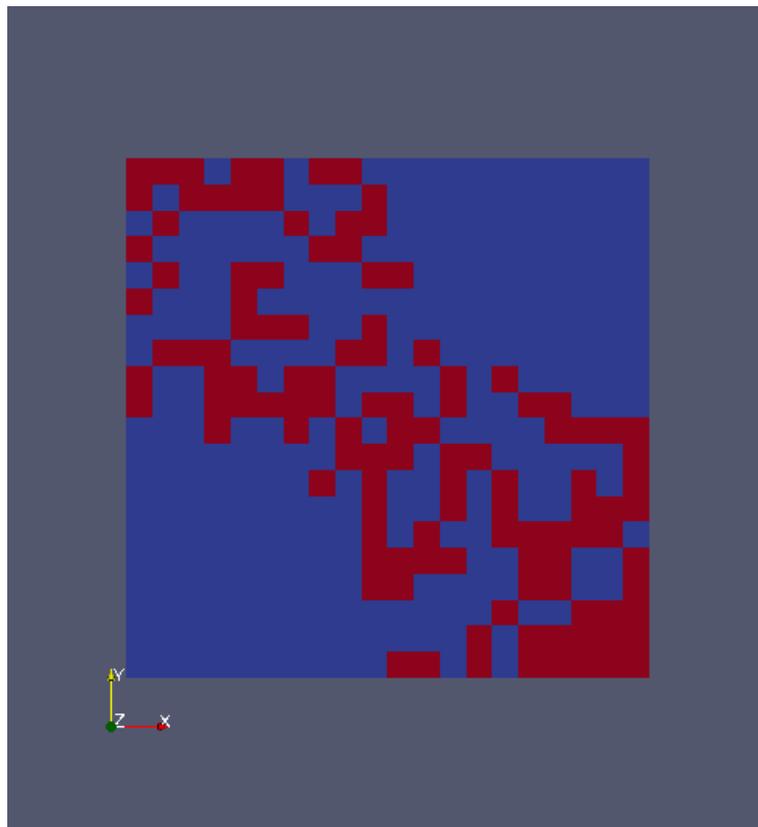


Figure 56: A sample configuration showing the distribution of low permeability cells in the intermediate (middle) region. This type of distribution is used in the *middle* configuration class.

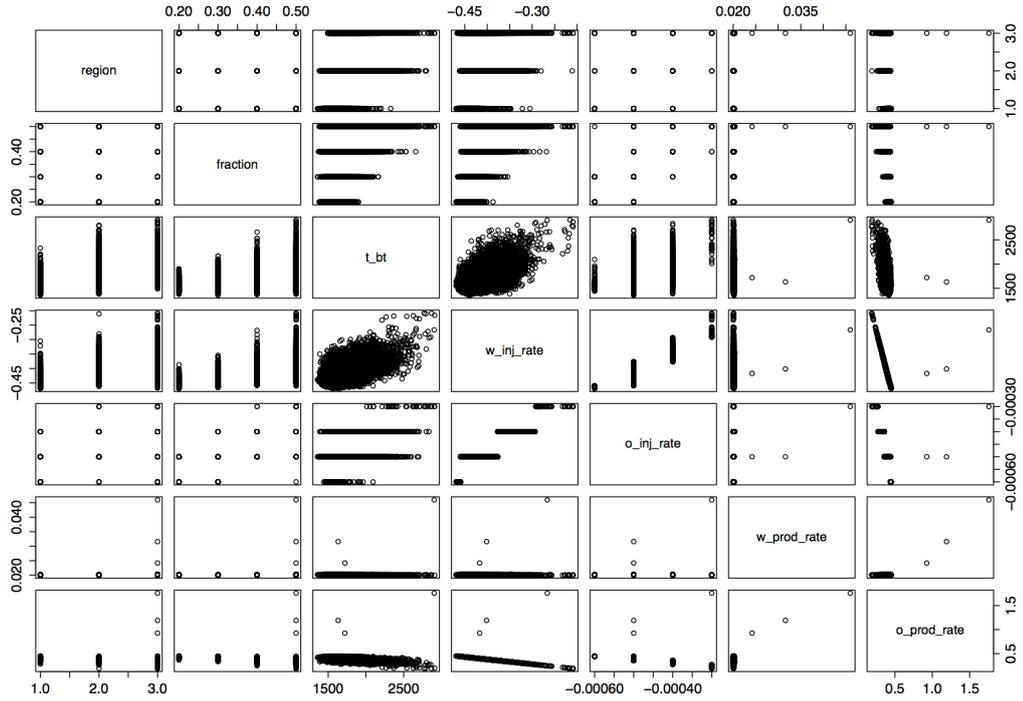


Figure 57: Scattered plots of region (1 = injector, 2 = middle and 3 = producer), probability, breakthrough time and the rates of injection and production, of water and oil, at breakthrough time .

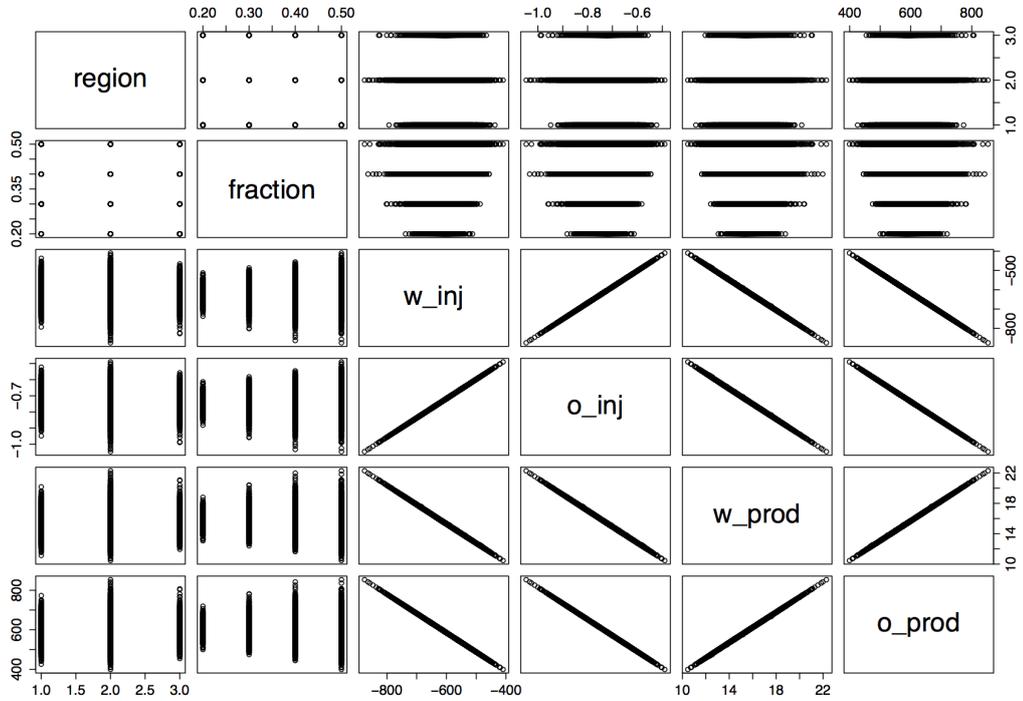


Figure 58: Scattered plots of region (1 = injector, 2 = middle and 3 = producer), probability and the accumulated amount of water and oil, injected and produced, at breakthrough time.

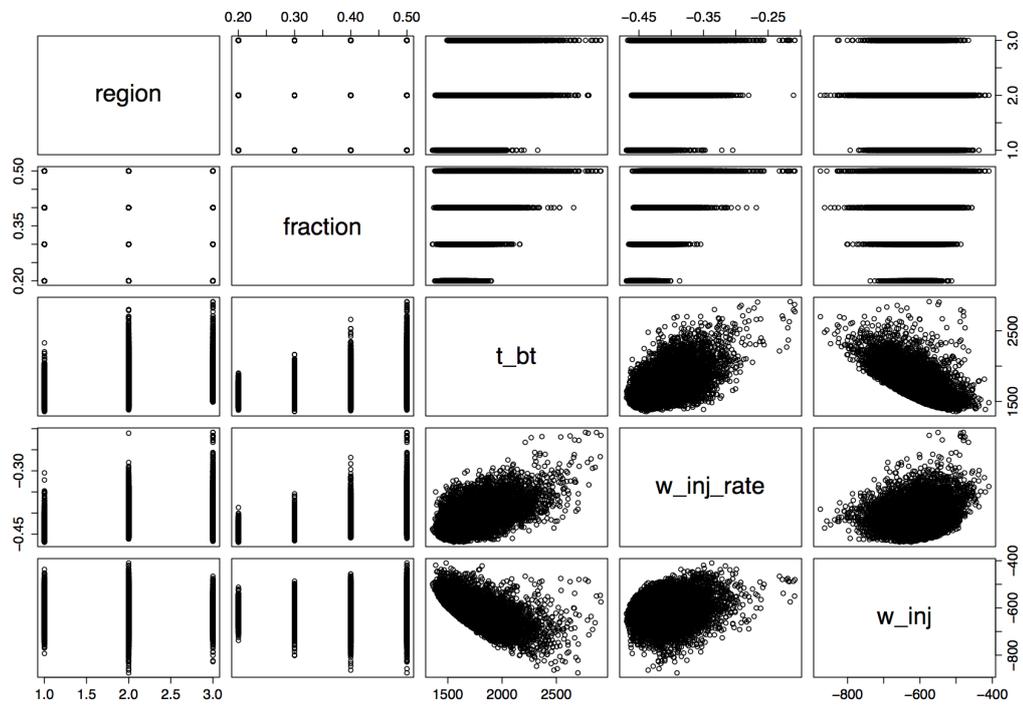


Figure 59: Scattered plots of region (1 = *injector*, 2 = *middle* and 3 = *producer*), probability, time of breakthrough, and water injection rate and the accumulated injected water at breakthrough time.

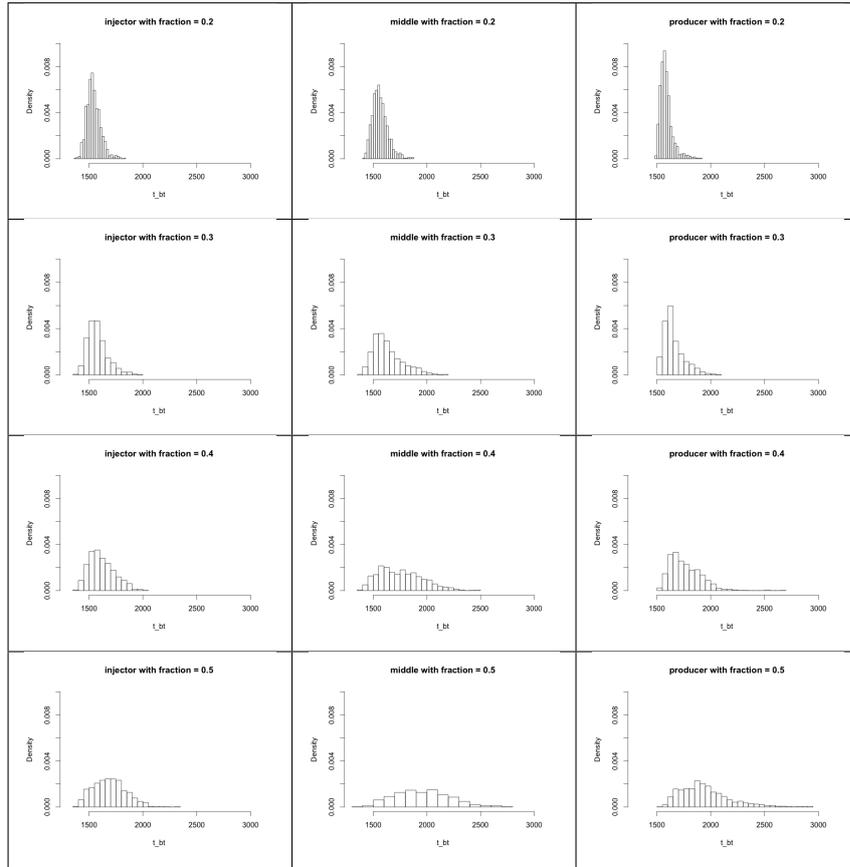


Figure 60: Histogram plots of the breakthrough time for the various probabilities and positioning of the low permeability regions.

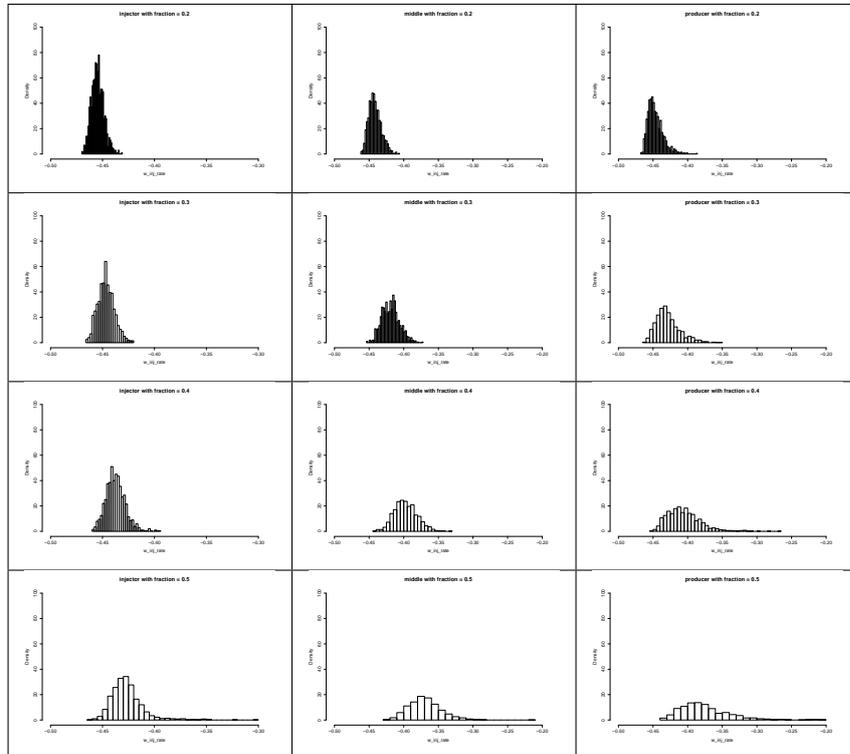


Figure 61: Histogram plots of the water injection rate at breakthrough time for the various probabilities and positioning of the low permeability regions.

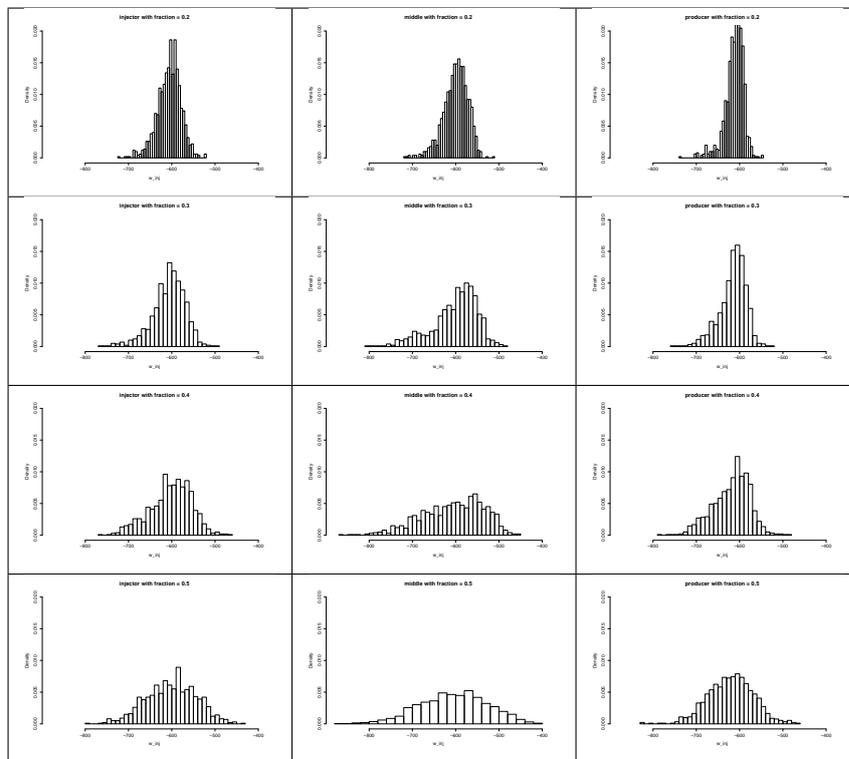


Figure 62: Histogram plots of the accumulated injected water at breakthrough time for the various probabilities and positioning of the low permeability regions..

APÊNDICE E – Código de Alguns Programas

E.1 Programa `heat.equation.1.R`: Método de linhas

```

# Programa heat.equation.1.R

# Sequência de tempo:
tl<-0; tr<-0.2; nt<-20; ht<-((tr-tl)/(nt-1)); t<-seq(tl,tr,ht)

# Sequencia espacial:
xl<-0;xr<-1 ;nx<-20;hx<-((xr-xl)/(nx-1));x<-seq(xl,xr,hx)

# Criando as equações:
Ui<-1:20;names(Ui)<-paste("U",Ui,sep="")

# Impondo uma condição inicial:
for(i in Ui){xi<-xl+(i-1)*hx;Ui[i]<-sin(pi*xi)}

# Criando o vetor y que sempre leva os dados iniciais para a região da função model:
y<-(Ui) parms<-c(alpha=1,hx=hx)

# Passando o valor de t para a variável times:
times<-t

# Estabelecendo o sistema de EDOs que rege o modelo na função model:
model<-function(times,y,parms){ with(as.list(c(y,parms)),{
U(xl,t)=U1(t)=0, U(xr,t)=U20(t)=0 -> U1x=0, U20x=0 # BC
U1.t<-alpha*((U2-2*U1+U0)/(hx^2))# BC
U2.t<-alpha*((U3-2*U2+U1)/(hx^2))

```

```

U3.t<-alpha*((U4-2*U3+U2)/(hx^2))
U4.t<-alpha*((U5-2*U4+U3)/(hx^2))
U5.t<-alpha*((U6-2*U5+U4)/(hx^2))
U6.t<-alpha*((U7-2*U6+U5)/(hx^2))
U7.t<-alpha*((U8-2*U7+U6)/(hx^2))
U8.t<-alpha*((U9-2*U8+U7)/(hx^2))
U9.t<-alpha*((U10-2*U9+U8)/(hx^2))
U10.t<-alpha*((U11-2*U10+U9)/(hx^2))
U11.t<-alpha*((U12-2*U11+U10)/(hx^2))
U12.t<-alpha*((U13-2*U12+U11)/(hx^2))
U13.t<-alpha*((U14-2*U13+U12)/(hx^2))
U14.t<-alpha*((U15-2*U14+U13)/(hx^2))
U15.t<-alpha*((U16-2*U15+U14)/(hx^2))
U16.t<-alpha*((U17-2*U16+U15)/(hx^2))
U17.t<-alpha*((U18-2*U17+U16)/(hx^2))
U18.t<-alpha*((U19-2*U18+U17)/(hx^2))
U19.t<-alpha*((U20-2*U19+U18)/(hx^2)) U20.
t<-alpha*((U21-2*U20+U19)/(hx^2)) # BC
return(list(c(U1.t,U2.t,U3.t,U4.t,U5.t,U6.t,U7.t,U8.t,U9.t,U10.t,U11.t,
U12.t,U13.t,U14.t,U15.t,U16.t,U17.t,U18.t,U19.t,U20.t))) }) }
# Carregando a biblioteca deSolve
library(deSolve)
# Resolvendo o sistema de equações
out<-ode(y,times,model,parms)
# Plotando um gráfico 3D
library(shape);library(rgl)

```

```

U.analitica<-matrix(nrow=nt,ncol=nx)

for(i in 1:nt){
  for(j in 1:nx){
    U.analitica[i,j]<-sin(pi*x[j])*exp(-pi^2*t[i])
  }#for(j in 1:nt)
}#for(i in 1:nt)

jet.colors <- colorRampPalette( c("lightgray", "blue", "green", "orange", "red" ) )
pal <- jet.colors(100)
col.ind <- cut(out[,-1],100)
persp3d(t,x,out[,-1],col=pal[col.ind],zlab="u(x,t)")
persp3d(t,x,out[,-1]-U.analitica,col=pal[col.ind],zlab="error")

# Plotando um mapa de contorno
image(t,x,out[,-1],ticktype="detailed",border="NA",col=pal,main="u(x,t)")

```

E.2 Programa `heat.equation.2.R`: Método de linhas (otimizado)

```

# Programa heat.equation.2.R

# Sequencia de tempo
t0<-0;tf<-0.2;nt<-150;ht<-((tf-t0)/(nt-1));times<-seq(t0,tf,ht)

# Sequencia espacial
xl<-0;xr<-1;nx<-150;hx<-((xr-xl)/(nx-1));x<-seq(xl,xr,hx)

# Condições iniciais e parâmetros
y<-1:nx;names(y)<-paste("U",y,sep=""); for(i in y){xi<-xl+(i-1)*hx;y[i]<-sin(pi*xi)}

```

```

y[1]=0;y[nx]=0

parms<-c(alpha=1,hx=hx,nx=nx)

# Equações do modelo
model<-function(times,y,parms){ with( as.list(c(y,parms)), {
odes<-1:nx;names(odes)<-paste("U",odes,sep="",".t")
for(i in 2:(nx-1)){odes[i]<-alpha*((y[i+1]-2*y[i]+y[i-1])/(hx^2))}
odes[1]<-y[1];odes[nx]<-y[nx]
list(odes)} ) }

# Resolvendo o modelo
library(deSolve); out<-ode(y,times,model,parms)

# Plotando em 3D
library(shape) library(rgl)

U.analitica<-matrix(nrow=nt,ncol=nx)

for(i in 1:nt){
for(j in 1:nx){
U.analitica[i,j]<-sin(pi*x[j])*exp(-pi^2*t[i])
}#for(j in 1:nt)
}#for(i in 1:nx)

jet.colors <- colorRampPalette( c("lightgray","blue","green","orange","red") )
pal <- jet.colors(100)

col.ind <- cut(out[,-1],100) # colour indices of each point
persp3d(times,x,out[,-1],col=pal[col.ind],zlab="u(x,t)")
persp3d(times,x,out[,-1]-U.analitica,col=pal[col.ind],zlab="error")

# Plotando o mapa de contorno
image(times,x,out[,-1],ticktype="detailed",border="NA",col=pal,main="u(x,t)",xlab="t")

```

E.3 Programa 2phase.flow.1.R: Método de linhas

```

# Programa 2phase.flow.1.R
derivate<-function(y,x,hx,nx){
Y<-splinefun(x,y)
f<-Y(x,deriv=1)
f<-splinefun(x,f)
yle<-Y(x[1]-hx)
yre<-Y(x[nx]+hx)
dY<-1:nx
for(i in 1:nx){
if(i<nx){
dY[i]<-2*(y[i+1]-y[i])/hx-f(x[i+1]) }#if
else{ dY[i]<-2*(yre-y[i])/hx-f(x[nx]+hx) }#else
}#for(i in 1:nx)
return(dY) }#derivate<-function(y,x,hx,nx)
xi<-0; xf<-100; hx<-0.1; x<-seq(xi,xf,hx); nx<-((xf-xi)/(hx) + 1 )
ti<-0; tf<-1; ht<-0.01; delta.t<-tf-ti; t<-seq(ti,tf,ht); nt<-((tf-ti)/(ht) + 1 )
Sn<-1:nx; Sn1<-0.20; Snn<-0.80
# Campo de saturacao inicial (abrupto)
for(i in 1:nx){ if( i>(1) ){ Sn[i]<-Snn }else{ Sn[i]<-Sn1; } }
Sn0<-Sn
x1<-0;y1<-0.2;x2<-100;y2<-0.8;C3<-0.3;
C2<-(( y1*exp(-C3*x1)-y2*exp(-C3*x2) )/(y2-y1)
C1<-(( y1*y2*exp(-C3*x1)-y1*y2*exp(-C3*x2) )/(y2-y1)
Sn<-function(x){

```

```

x1<-0;y1<-0.2; x2<-100;y2<-0.8; C3<-0.3
C2<- ( y1*exp(-C3*x1)-y2*exp(-C3*x2) )/(y2-y1)
C1<- ( y1*y2*exp(-C3*x1)-y1*y2*exp(-C3*x2) )/(y2-y1)
Sn<-C1/(C2+exp(-C3*x)) return(Sn) } # Campo de saturacao inicial (suave)
Sn0<-Sn(x)
plot(x,Sn0);lines(x,Sn(x))
# Testando a derivada analítica, contra a derivada numérica derivate()
dSn.test<- ( C1*C3*exp(-C3*x) )/( ( C2+exp(-C3*x) )^2 )
plot(x,dSn.test);lines(x,derivate(Sn(x),x,hx,nx)) Sn<-Sn(x)
k<-1e-8; phi<-0.2; Pd<-1e+3; theta<-2; muw<-1e-3; mun<-1e-2
Pwinj<-3e+6; Pwprod<-1e+6
system.time( {
tempo.total<-300
while(tf<tempo.total){
Pc<-Pd*( 1-Sn )^( (-1)/(theta) );krw<-( 1-Sn )^( (2+3*theta)/(theta) )
lambdaw<- ( krw)/(muw) )
krn<- ( Sn)^2 *( 1 - (1-Sn)^((2+theta)/(theta)) ); lambdan<- ( krn)/(mun) )
dPc<-derivate(Pc,x,hx,nx); d2Pc <-derivate(dPc,x,hx,nx)
I<-k*derivate( ( lambdan+lambdaw ),x,hx,nx)
II<-k*( lambdan+lambdaw ); III<-k*derivate(lambdan,x,hx,nx)
IV<-dPc V<-k*lambdan; VI<-d2Pc
Pw2test=3000; percent.error=100; while( percent.error > 1){
Pw1<-Pwinj; Pw2<-Pw2test; Pw<-c(Pw1=Pw1,Pw2=Pw2)
Pwparms<-c(I,II,III,IV,V,VI)
PwModel<-function(x,Pw,Pwparms){
with(as.list(c(Pw,Pwparms)),{

```

```

j<-(x-xi)/hx+1; dPw1<-Pw2
dPw2<-(-1)*(( I[j]*Pw2 + III[j]*IV[j] + V[j]*VI[j] )/( II[j] ))
list( c(dPw1,dPw2) ) )
library(deSolve); PwOut<-ode(Pw,x,PwModel,Pwparms)
Pw<-1:nx; convert<-as.array(PwOut[,2])
for(i in 1:nx){ Pw[i]<-convert[i]}
delta<-PwOut[nx,2]-Pwprod; percent.error<- ( abs(delta)/Pwprod ) *100
Pw2test=Pw2test+10 }#while( percent.error > 0.1)
library(Hmisc)
setwd("/home/marceliano/SIMULACOES/hx01ht02C308cv/PRES")
jpeg(paste("fig", tf, ".jpg", sep = ""))
par(mai=c(1.1,1.5,1,0.5))
plot(x,Pw,xlab="X",ylab="PRES",type="line",col=2,yaxt='n',xaxt='n',
xlim=c(0,100), ylim=c(1e+6,3e+6), lwd=2,lty=1,
cex.axis=2,cex.lab=2,cex.main=2,main="")
axis(1,at=c(0,20,40,60,80,100),
labels=c(0,20,40,60,80,100))
axis(2,at=c(1e+6,1.5e+6,2e+6,2.5e+6,3.0e+6),
labels=c(1e+6,1.5e+6,2e+6,2.5e+6,3.0e+6))
dev.off()
dPw<-derivate(Pw,x,hx,nx)
d2Pw<-derivate(dPw,x,hx,nx)
VII<-k*derivate(lambdaw,x,hx,nx)
VIII<-k*lambdaw
SnI<-c(Sn=Sn)
Snparms<-c(phi=phi,VII,VIII,dPw,d2Pw,xi=xi,nx=nx,hx=hx)

```

```

SnModel<-function(t,SnI,Snparms){ with(as.list(c(Sn,Snparms)),{
odes<-1:nx; names(odes)<-paste("dSn",odes,sep="")
for(p in 1:nx){
odes[p]<-(-1)*( VII[p]*dPw[p] + VIII[p]*d2Pw[p] )/( phi ) }#for(p in 2:nx)
list(c(odes)) }, )}
library(deSolve); SnOut<-ode(SnI,t,SnModel,Snparms)
Sn<-1:nx; convert<-as.array(SnOut[nt,]); for(i in 1:nx){
Sn[i]<-convert[i+1] if(Sn[i]<0.2){Sn[i]<-0.2}
if(Sn[i]>Sn0[i]){Sn[i]<-Sn0[i]} } Sn[1]<-0.2
Sni<-smooth.spline(x,Sn,spar=0.3) Sn<-Sni$y
setwd("/home/marceliano/SIMULACOES/hx01ht02C308cv/SAT")
jpeg(paste("fig", tf, ".jpg", sep = ""))
par(mai=c(1.1,1.5,1,0.5))
plot(x,1-Sn,xlab="X",ylab="SAT",type="line",
col=3,lwd=2,lty=1,main="",yaxt='n',xaxt='n',
xlim=c(0,100), ylim=c(0,1), cex.axis=2,cex.lab=2,cex.main=2)
axis(1,at=c(0,20,40,60,80,100),labels=c(0,20,40,60,80,100))
axis(2,at=c(0,0.2,0.4,0.6,0.8,1.0),labels=c(0,0.2,0.4,0.6,0.8,1.0))
minor.tick(nx=4,ny=4)
dev.off()
ti=ti+delta.t
tf=tf+delta.t
t<-seq(ti,tf,ht)
}})

```

E.4 Programa difusao1D.estacionario.R: Volumes Finitos

```

setwd("/home/marceliano/Dropbox
/Qualificação.direta/FVM/1D steady state difusion")

l=0.5; x0=0; xnf=0.5; n=20; interval=3

phi0=100; phinf=500; Gamma=100

nf=n+1; delta.x.n=l/n; delta.x.f=delta.x.n/2;

first.node=x0+delta.x.f; last.node=first.node+(n-1)*delta.x.n

grid1D<-seq(first.node,last.node,delta.x.n)

Gamma.x=matrix(data=0,nrow=1,ncol=nf)

for(j in 1:nf){Gamma.x[j]<-Gamma}

A10=Gamma.x[1]/delta.x.f; A1nf=Gamma.x[nf]/delta.x.f

library(Matrix)

A=matrix(data=0,nrow=n,ncol=n)

for(i in 1:n){ for(j in 1:n){ if(i==j){

if(i==1){ A[i,j+1]=Gamma.x[j]/delta.x.n; A[i,j]=-A10 -A[i,j+1] }#if(i==1)

if(i>1 & i<n){ A[i,j-1]=Gamma.x[j]/delta.x.n

A[i,j+1]=Gamma.x[j+1]/delta.x.n

A[i,j]=-A[i,j-1]-A[i,j+1] }#if(i>2 & i<n)

if(i==n){ A[i,j-1]=Gamma.x[j]/delta.x.n; A[i,j]=-A[i,j-1]-A1nf } } }

Asparse<-as(A,"sparseMatrix")

object.size(A); object.size(Asparse)

C=matrix(data=0,nrow=n,ncol=1)

for(i in 1:n){

if(i==1){ C[i]=-A10*phi0 }

```

```

if(i==n){ C[i]=-A1nf*phin } }
system.time(( phi=try( solve(Asparse, C, sparse=TRUE,tol = 1e-19) ) ))

```

E.5 heatTransfer2D.R

```

# 0functions.R
# This file contain extra defined functions
source("1simulationData.R")
library(Matrix) library(MASS)
# Definition of saveMatrix("namefile",matrix)
saveMatrix=function(namefile,matrix){
write.matrix(
matrixToBeSaved, file=nameFile
)#write.matrix
}#saveMatrix=function(namefile,matrix)
# Definition of loadMatrix("namefile")
loadMatrix=function(nameFile){
data=as.matrix(read.table(file=nameFile))
out=matrix( data=data, nrow=nrow(data), ncol=ncol(data) )
return(out) }#loadMatrix
# Definition of isInteger(a,b)
isInteger=function(a,b){
if( (a/b)%%1==0 ){
booleanVariable=TRUE
}else{
booleanVariable=FALSE

```

```

}#else

return(booleanVariable)

}#isInteger=function(a,b)

# 1simulationData.R

# This file contain the simulation parameters

# Initial data: square domain
xLength=1; yLength=1; xInitial=0; yInitial=0
tInitial=0; tFinal=15000; xFinal=xLength
yFinal=yLength; deltaZ=1

# Initial data: points and instants
nPointsX=40; nPointsY=40; nInstants=20000

# Initial data: material
density=1; heatConductivity=1.0e-5; heatCapacity=1
phiInitialAllPoints=200; phiInitialW=900; phiInitialE=800
phiInitialS=900; phiInitialN=800; Gamma=heatConductivity/(density*heatCapacity)

# ThetaModel parameter theta=(0,0.5,1) ->
# -> (forward Euler, Cranck Nicolson, backward Euler)
theta=1.0

# Obtaining parameters for FVM method
dxPointToPoint=xLength/nPointsX; dyPointToPoint=yLength/nPointsY
dxPointToFace=dxPointToPoint/2; dyPointToFace=dyPointToPoint/2
firstPointLocationX=xInitial+dxPointToFace; firstPointLocationY=yInitial+dyPointToFace
lastPointLocationX=firstPointLocationX+(nPointsX-1)*dxPointToPoint
lastPointLocationY=firstPointLocationY+(nPointsY-1)*dyPointToPoint
dt=(tFinal-tInitial)/(nInstants-1)

```

```

# 2gridsCreation.R

# This file create xGrid and tGrid that contain nodal points coordinates and time
instants

source("0functions.R")
source("1simulationData.R")

library(Matrix)
library(MASS)

# Create xGrid xGrid=matrix(data=0,nrow=nPointsX,ncol=1)
# Create yGrid yGrid=matrix(data=0,nrow=nPointsY,ncol=1)
# Create tGrid tGrid=matrix(data=0,nrow=nInstants,ncol=1)

# Input xGrid values
pointLocationX=firstPointLocationX
for(l in 1:nPointsX){
xGrid[l]<-pointLocationX

pointLocationX=pointLocationX+dxPointToPoint
}#for(l in 1:nPointsX)

# Input yGrid values
pointLocationY=firstPointLocationY
for(l in 1:nPointsY){
yGrid[l]=pointLocationY

pointLocationY=pointLocationY+dyPointToPoint
}#for(l in 1:nPointsY)

# Input tGrid values
instantLocation=tInitial
for(l in 1:nInstants){

```

```

tGrid[l]=instantLocation

instantLocation=instantLocation+dt

}#for(l in 1:nInstants)

# Save xGrid saveMatrix("xGrid",xGrid)

# Save yGrid saveMatrix("yGrid",yGrid)

# Save tGrid saveMatrix("tGrid",tGrid)

# 3gammaMatricesCreation.R

# This file create the wGamma, eGamma,

# sGamma and nGamma matrices

source("0functions.R")

source("1simulationData.R")

library(Matrix)

library(MASS)

# Create wGamma, eGamma, sGamma, nGamma

wGamma=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)

eGamma=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)

sGamma=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)

nGamma=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)

# Input wGamma values

counter=1 for(i in 1:nPointsY){

for(j in 1:nPointsX){

wGamma[counter,1]=Gamma counter=counter+1

}#for(j in 1:nPointsX)

}#for(i in 1:nPointsY)

# Input eGamma values

```

```

counter=1
for(i in 1:nPointsY){
for(j in 1:nPointsX){
eGamma[counter,1]=Gamma counter=counter+1
}#for(j in 1:nPointsX)
}#for(i in 1:nPointsY)
# Input sGamma values
counter=1
for(i in 1:nPointsY){
for(j in 1:nPointsX){
sGamma[counter,1]=Gamma counter=counter+1
}#for(j in 1:nPointsX)
}#for(i in 1:nPointsY)
# Input nGamma values
counter=1
for(i in 1:nPointsY){
for(j in 1:nPointsX){
nGamma[counter,1]=Gamma counter=counter+1
}#for(j in 1:nPointsX)
}#for(i in 1:nPointsY)
# Save wGamma, eGamma, sGamma, nGamma
saveMatrix("wGamma",wGamma); saveMatrix("eGamma",eGamma)
saveMatrix("sGamma",sGamma); saveMatrix("nGamma",nGamma)

# 4difusionExternalMatrixCreation.R
# This file create difusionExternalMatrices

```

```

source("0functions.R")

source("1simulationData.R")

library(Matrix)

library(MASS)

# load wGamma, eGamma, sGamma, nGamma
wGamma=loadMatrix("wGamma"); eGamma=loadMatrix("eGamma")
sGamma=loadMatrix("sGamma"); nGamma=loadMatrix("nGamma")

# Create difusionExternalMatrixW, difusionExternalMatrixE,
# difusionExternalMatrixS, difusionExternalMatrixN
difusionExternalMatrixW=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)
difusionExternalMatrixE=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)
difusionExternalMatrixS=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)
difusionExternalMatrixN=matrix(data=0,nrow=nPointsX*nPointsY,ncol=1)

# Input values in difusionExternalMatrixW
for(j in 1:(nPointsX*nPointsY)){
  if(isInteger((j-1),nPointsX)){
    difusionExternalMatrixW[j,1]=
      (dt*dyPointToPoint*deltaZ)*wGamma[j,1]/(dxPointToFace)
  }#if(isInteger((j-1),nPointsX))
}#for(j in 1:(nPointsX*nPointsY))

# Input values in difusionExternalMatrixE
for(j in 1:(nPointsX*nPointsY)){
  if(j>1 & isInteger(j,nPointsX)){
    difusionExternalMatrixE[j,1]=
      (dt*dyPointToPoint*deltaZ)*eGamma[j,1]/(dxPointToFace)
  }#if(j>1 & isInteger(j,nPointsX))
}

```

```

}#for(j in 1:(nPointsX*nPointsY))

# Input values in difusionExternalMatrixS

for(j in 1:(nPointsX*nPointsY)){

if(j<=nPointsX){

difusionExternalMatrixS[j,1]=(dt*dxPointToPoint*deltaZ)*sGamma[j,1]/(dyPointToFace)

}#if(j<=nPointsX)

}#for(j in 1:(nPointsX*nPointsY))

# Input values in difusionExternalMatrixN

for(j in 1:(nPointsX*nPointsY)){

if(j>(nPointsX*nPointsY-nPointsX)){

difusionExternalMatrixN[j,1]=(dt*dyPointToPoint*deltaZ)*nGamma[j,1]/(dyPointToFace)

}#if(j>(nPointsX*nPointsY-nPointsX))

}#for(j in 1:(nPointsX*nPointsY))

# Save difusionExternalMatrixW, difusionExternalMatrixE,

# difusionExternalMatrixS, difusionExternalMatrixN

saveMatrix("difusionExternalMatrixW",difusionExternalMatrixW)

saveMatrix("difusionExternalMatrixE",difusionExternalMatrixE)

saveMatrix("difusionExternalMatrixS",difusionExternalMatrixS)

saveMatrix("difusionExternalMatrixN",difusionExternalMatrixN)

# 5difusionInternalMatrixCreation.R

# This file create difusionInternalMatrix

source("0functions.R")

source("1simulationData.R")

library(Matrix)

library(MASS)

```

```

# Read wGamma, eGamma, sGamma, nGamma
wGamma=loadMatrix("wGamma")
eGamma=loadMatrix("eGamma")
sGamma=loadMatrix("sGamma")
nGamma=loadMatrix("nGamma")

difusionExternalMatrixW=loadMatrix("difusionExternalMatrixW")
difusionExternalMatrixE=loadMatrix("difusionExternalMatrixE")
difusionExternalMatrixS=loadMatrix("difusionExternalMatrixS")
difusionExternalMatrixN=loadMatrix("difusionExternalMatrixN")

# Create difusionInternalMatrix
difusionInternalMatrix=matrix(data=0,ncol=nPointsX*
nPointsY,nrow=nPointsX*nPointsY)
wCounter=1; eCounter=1; sCounter=1; nCounter=1
for(i in 1:(nPointsX*nPointsY)){
for(j in 1:(nPointsX*nPointsY)){
# diagonal of west contribution
if( i==(j+1) ){
difusionInternalMatrix[i,j]=(deltaZ*dyPointToPoint)*
wGamma[wCounter,1]/(dxPointToPoint)
if( isInteger(wCounter,nPointsX) ){
difusionInternalMatrix[i,j]=0
}#if( isInteger(wCounter,nPointsX) )
wCounter=wCounter+1
}#if( i==(j+1) )
# diagonal of east contribution
if(i==j-1){

```

```

difusionInternalMatrix[i,j]=(deltaZ*dyPointToPoint)*
eGamma[eCounter+1,1]/(dxPointToPoint)
if( isInteger(eCounter,nPointsX) ){
difusionInternalMatrix[i,j]<-0
}#if( isInteger(eCounter,nPointsX) )
eCounter=eCounter+1
}#if(i==j-1)
# diagonal of south contribution
if( i==(j+nPointsX) ){
difusionInternalMatrix[i,j]=(deltaZ*dxPointToPoint)*
sGamma[sCounter,1]/(dyPointToPoint)
sCounter=sCounter+1
}#if(i==(j+nPointsX))
# diagonal of north contribution
if(i==j-nPointsX){
difusionInternalMatrix[i,j]=(deltaZ*dxPointToPoint)*
nGamma[nCounter+nPointsX,1]/(dyPointToPoint)
nCounter=nCounter+1
}#if(i==j-nPointsY)
}#for(i in 1:nPointsX*nPointsY) }#for(j in 1:nPointsX*nPointsY)
for(i in 1:(nPointsX*nPointsY)){
for(j in 1:(nPointsX*nPointsY)){
# diagonal of point contribution if(i==j){
if(i==1){
difusionInternalMatrix[i,j]=-difusionExternalMatrixW[i,1]+
-difusionExternalMatrixE[i,1]+

```

```

-difusionExternalMatrixS[i,1]+
-difusionExternalMatrixN[i,1]+ #aw
#-difusionInternalMatrix[i,j-1]+ #ae
-difusionInternalMatrix[i,j+1]+ #as
#-difusionInternalMatrix[i,j-nPointsX]+ #an
-difusionInternalMatrix[i,j+nPointsX]
}#if(i==1)
if(i>1 & i<=nPointsX){
difusionInternalMatrix[i,j]=-difusionExternalMatrixW[i,1]+
-difusionExternalMatrixE[i,1]+
-difusionExternalMatrixS[i,1]+
-difusionExternalMatrixN[i,1]+ #aw
-difusionInternalMatrix[i,j-1]+ #ae
-difusionInternalMatrix[i,j+1]+ #as
#-difusionInternalMatrix[i,j-nPointsX]+ #an
-difusionInternalMatrix[i,j+nPointsX]
}#if(i>1 & i<=nPointsX)
if( i>nPointsX & i<=(nPointsX*nPointsY-nPointsX) ){
difusionInternalMatrix[i,j]=-difusionExternalMatrixW[i,1]+
-difusionExternalMatrixE[i,1]+
-difusionExternalMatrixS[i,1]+
-difusionExternalMatrixN[i,1]+ #aw
-difusionInternalMatrix[i,j-1]+ #ae
-difusionInternalMatrix[i,j+1]+ #as
-difusionInternalMatrix[i,j-nPointsX]+ #an
-difusionInternalMatrix[i,j+nPointsX]

```

```

}#if( i>1 & i<(nPointsX*nPointsY) )
if(i<(nPointsX*nPointsY) & i>(nPointsX*nPointsY-nPointsX)){
difusionInternalMatrix[i,j]=-difusionExternalMatrixW[i,1]+
-difusionExternalMatrixE[i,1]+
-difusionExternalMatrixS[i,1]+
-difusionExternalMatrixN[i,1]+ #aw
-difusionInternalMatrix[i,j-1]+ #ae
-difusionInternalMatrix[i,j+1]+ #as
-difusionInternalMatrix[i,j-nPointsX]#+ #an
#-difusionInternalMatrix[i,j+nPointsX]
}#if(i<(nPointsX*nPointsY-nPointsX) & i>(nPointsX*nPointsY-nPointsX))
if(i==(nPointsX*nPointsY)){
difusionInternalMatrix[i,j]=-difusionExternalMatrixW[i,1]+
-difusionExternalMatrixE[i,1]+
-difusionExternalMatrixS[i,1]+
-difusionExternalMatrixN[i,1]+ #aw
-difusionInternalMatrix[i,j-1]+ #ae
#-difusionInternalMatrix[i,j+1]+ #as
-difusionInternalMatrix[i,j-nPointsX]#+ #an
#-difusionInternalMatrix[i,j+nPointsX]
}#if(i<(nPointsX*nPointsY-nPointsX) & i>(nPointsX*nPointsY-nPointsX))
}#if(i==j)
}#for(i in 1:nPointsX*nPointsY)
}#for(j in 1:nPointsX*nPointsY)
# Save difusionInternalMatrix
saveMatrix("difusionInternalMatrix",difusionInternalMatrix)

```

```

# marcelianooliveira@gmail.com
# 6boundaryConditionsMatrixCreation.R
# This file create boundaryConditionsMatrix
source("0functions.R")
source("1simulationData.R")
library(Matrix) library(MASS)
# Create boundaryConditionsMatrixW, boundaryConditionsMatrixE,
# boundaryConditionsMatrixS, boundaryConditionsMatrixN
boundaryConditionsMatrixW=
matrix(data=phiInitialW,nrow=nPointsX*nPointsY,ncol=1)
boundaryConditionsMatrixE=
matrix(data=phiInitialE,nrow=nPointsX*nPointsY,ncol=1)
boundaryConditionsMatrixS=
matrix(data=phiInitialS,nrow=nPointsX*nPointsY,ncol=1)
boundaryConditionsMatrixN=
matrix(data=phiInitialN,nrow=nPointsX*nPointsY,ncol=1)
# Input values in boundaryConditionsMatrixW
counter=1
for(j in 1:(nPointsX*nPointsY)){
  if(isInteger((j-1),nPointsX)){
    if(counter==1){
      boundaryConditionsMatrixW[j,1]=phiInitialW
    }else{
      boundaryConditionsMatrixW[j,1]=0
    }#else

```

```

counter=counter+1
}#if(isInteger((j-1),nPointsX))
}#for(j in 1:(nPointsX*nPointsY))
# Input values in boundaryConditionsMatrixE
counter=1
for(j in 1:(nPointsX*nPointsY)){
if(j>1 & isInteger(j,nPointsX)){
if(counter==nPointsY){
boundaryConditionsMatrixE[j,1]=phiInitialE
}else{
boundaryConditionsMatrixE[j,1]=0
}#else
counter=counter+1
}#if(j>1 & isInteger(j,nPointsX))
}#for(j in 1:(nPointsX*nPointsY))
# Input values in boundaryConditionsMatrixS
counter=1
for(j in 1:(nPointsX*nPointsY)){
if(j<=nPointsX){
if(counter==1){
boundaryConditionsMatrixS[j,1]=phiInitialS
}else{
boundaryConditionsMatrixS[j,1]=0
}#else
counter=counter+1
}#if(j<=nPointsX)

```

```

}#for(j in 1:(nPointsX*nPointsY))

Input values in boundaryConditionsMatrixN

counter=1

for(j in 1:(nPointsX*nPointsY)){
if(j>(nPointsX*nPointsY-nPointsX)){
if(counter==nPointsX){
boundaryConditionsMatrixN[j,1]=phiInitialN
}else{
boundaryConditionsMatrixN[j,1]=0
}#else # counter=counter+1
}#if(j>(nPointsX*nPointsY-nPointsX))
}#for(j in 1:(nPointsX*nPointsY))

# Save boundaryConditionsMatrixW, boundaryConditionsMatrixE,
#boundaryConditionsMatrixS, boundaryConditionsMatrixN

saveMatrix("boundaryConditionsMatrixW",boundaryConditionsMatrixW)
saveMatrix("boundaryConditionsMatrixE",boundaryConditionsMatrixE)
saveMatrix("boundaryConditionsMatrixS",boundaryConditionsMatrixS)
saveMatrix("boundaryConditionsMatrixN",boundaryConditionsMatrixN)

# 7transientDifusionSolverThetaModel.R

# This file contain the main program that solve the problem

source("0functions.R")

source("1simulationData.R")

library(Matrix)

library(MASS)

# Load matrices

```

```

xGrid=loadMatrix("xGrid")

boundaryConditionsMatrixW=loadMatrix("boundaryConditionsMatrixW")

boundaryConditionsMatrixE=loadMatrix("boundaryConditionsMatrixE")

boundaryConditionsMatrixS=loadMatrix("boundaryConditionsMatrixS")

boundaryConditionsMatrixN=loadMatrix("boundaryConditionsMatrixN")

difusionExternalMatrixW=loadMatrix("difusionExternalMatrixW")

difusionExternalMatrixE=loadMatrix("difusionExternalMatrixE")

difusionExternalMatrixS=loadMatrix("difusionExternalMatrixS")

difusionExternalMatrixN=loadMatrix("difusionExternalMatrixN")

difusionInternalMatrix=loadMatrix("difusionInternalMatrix")

# Declare phi(x,y,t=0)

phiInitialT=matrix(data=phiInitialAllPoints,nrow=nPointsX*nPointsY,ncol=1)

# Declare A matrix

A=(dxPointToPoint*dyPointToPoint*deltaZ)*diag(nPointsX*nPointsY)-
difusionInternalMatrix*theta

# Convert A matrix to Asparse matrix

Asparse=as(A,"sparseMatrix")

rm(A)

# Declare C matrix

C=difusionExternalMatrixW*boundaryConditionsMatrixW+
difusionExternalMatrixE*boundaryConditionsMatrixE+
difusionExternalMatrixS*boundaryConditionsMatrixS+
difusionExternalMatrixN*boundaryConditionsMatrixN

# Advancing in time

phiOld=phiInitialT

phiNew=phiInitialT

```

```

for(l in 1:nInstants){
if(l>1){
# Declare B matrix
B=((
dxPointToPoint*dyPointToPoint*deltaZ)*diag(nPointsX*nPointsY)+
difusionInternalMatrix*(1-theta))%*%phiOld+
C
# Solve the system Ax=B
phiNew=try(solve(Asparse,B,sparse=TRUE,tol=1e-19))
# Reload phiNew to phiOld before run loop again phiOld=phiNew
}#if(l>1)
# Reassemble phiNew matrix in 2D grid format
phi=matrix(data=0,nrow=nPointsY,ncol=nPointsX)
counter=1
for(i in 1:nPointsY){
for(j in 1:nPointsX){
phi[i,j]=phiNew[counter] counter=counter+1
}#for(j in 1:nPointsX)
}#for(i in 1:nPointsY)
# Save phi saveMatrix(paste("phi",l,sep=""),phi)
# Print phi when computed print(paste("phi",l,sep=""))
}#for(l=1:nInstants)

```

REFERÊNCIAS

- [1] <http://www.mphpa.org/classic/HISTORY/H-06c18.htm>.
- [2] <http://www.r-project.org/>.
- [3] Openfoam software.
- [4] Jamal H Abou-Kassem, SM Farouq Ali, and M Rafiq Islam. *Petroleum reservoir simulation: a basic approach*. Houston, TX: Gulf Pub. Co., 2006.
- [5] Jacob Bear. *Dynamics of fluids in Porous Media*. Dover Publications, Inc., New York, USA.
- [6] Jose Luiz Boldrini, Sueli IR Costa, VL Figueredo, and Henry G Wetzler. *Álgebra linear*. Harbra, 1980.
- [7] Zhangxin Chen. *Reservoir simulation*, volume 77. SIAM, 2007.
- [8] Michael J. Crawley. *The R Book*. John Wiley & Sons, Ltd., 2010.
- [9] Y Epshteyn and Béatrice Rivière. Fully implicit discontinuous finite element methods for two-phase flow. *Applied Numerical Mathematics*, 57(4):383–401, 2007.
- [10] Yekaterina Epshteyn and Beatrice Rivière. Analysis of discontinuous galerkin methods for incompressible two-phase flow. *Journal of Computational and Applied Mathematics*, 225(2):487 – 509, 2009.
- [11] Richard E Ewing. *The mathematics of reservoir simulation*. Siam, 1983.
- [12] Mark S Gockenbach. *Partial differential equations: analytical and numerical methods*. Siam, 2010.
- [13] R S Johnson. Notes on pdes classification and canonical forms, 2007.
- [14] George Em Karniadakis and Spencer J Sherwin. *Spectral/hp element methods for CFD*. Oxford University Press, 1999.
- [15] Heinz-Otto Kreis and Jenz Lorenz. *Initial-Boundary Problems and the Navier-Stokes Equation*. SIAM, 1989.
- [16] Hyun Jin Lee and William E Schiesser. *Ordinary and partial differential equation routines in C, C++, Fortran, Java, Maple, and Matlab*. CRC Press, 2003.
- [17] Jichun Li and Yi-Tung Chen. *Computational partial differential equations using MATLAB*. Crc Press, 2011.

- [18] Marceliano E. de Oliveira Murilo P. Almeida. Openfoam implementation of the equations two immiscible fluid flow in porous media.
- [19] Donald W Peaceman. *Fundamentals of numerical reservoir simulation*. Elsevier, 1977.
- [20] R Development Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2012. ISBN 3-900051-07-0.
- [21] R.D. Richtmyer. *Difference Methods for Initial Value Problems*. Richtmyer. Interscience tracts in pure and applied mathematics. 1967.
- [22] Béatrice Rivière. *Discontinuous Galerkin methods for solving elliptic and parabolic equations: theory and implementation*, volume 35. Siam, 2008.
- [23] Michael Schäfer. *Computational engineering: introduction to numerical methods*. Springer, 2006.
- [24] William E Schiesser and Graham W Griffiths. *A compendium of partial differential equation models: method of lines analysis with Matlab*. Cambridge University Press, 2009.
- [25] Karline Soetaert, Thomas Petzoldt, and R Woodrow Setzer. Solving differential equations in r: Package desolve. *Journal of Statistical Software*, 33(9):1–25, 2010.
- [26] Harvey Gould & Jan Tobochnik. *An Introduction to Computer Simulation Methods: Applications to Physical Systems*. Addison-Wesley Publishing Company, 2nd edition, 1996.
- [27] HK Vesteeg and W Malalasekera. *An introduction to computational fluid dynamics*. Pearson Education Limited, England, 2007.