Pontifícia Universidade Católica
do Rio de Janeiro

Thiago Sousa Bastos

Development of a multipurpose reservoir simulator based on a plugin architecture

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica.

Advisor : Prof. Ivan Fábio Mota de Menezes
Co-advisor: Dsc. Leonardo Seperuelo Duarte

Rio de Janeiro
April 2021

PUC-Rio - Certificação Digital Nº 1912763/CA

Pontifícia Universidade Católica do Rio de Janeiro

Thiago Sousa Bastos

Development of a multipurpose reservoir simulator based on a plugin architecture

Dissertation presented to the Programa de Pós–graduação em Engenharia Mecânica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Mecânica. Approved by the Examination Committee:

Prof. Ivan Fábio Mota de Menezes
Advisor
Departamento de Engenharia Mecânica – PUC-Rio

Dsc. Leonardo Seperuelo Duarte
Co-advisor
Instituto Tecgraf de Desenvolvimento de Software
Técnico-Cientifico da PUC-Rio – Tecgraf/PUC-Rio

Daniel Nunes de Miranda Filho
Petróleo Brasileiro S.A. – Petrobras

Luiz Otávio Schmall dos Santos
Petróleo Brasileiro S.A. – Petrobras

Rio de Janeiro, April the 29th, 2021

PUC-Rio - Certificação Digital Nº 1912763/CA

All rights reserved.

Thiago Sousa Bastos

The author graduated in 2018 with a major in Mechanical Engineering from Pontifical Catholic University of Rio de Janeiro. While undertaking his graduate studies, he worked as a researcher at Tecgraf/PUC-Rio.

PUC-Rio - Certificação Digital Nº 1912763/CA

Bibliographic data

Bastos, Thiago Sousa

Development of a multipurpose reservoir simulator based on a plugin architecture / Thiago Sousa Bastos; advisor: Ivan Fábio Mota de Menezes; co-advisor: Leonardo Seperuelo Duarte. – 2021.

97 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Mecânica, 2021.

Inclui bibliografia

1. Engenharia Mecânica – Teses. 2. Simulação de reservatórios. 3. Modelo black-oil. 4. Escoamento multifásico. 5. Meios porosos. 6. Sistema baseado em plugin. I. Menezes, Ivan Fábio Mota de. II. Duarte, Leonardo Seperuelo. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Mecânica. IV. Título.

CDD: 621

Acknowledgments

I would like to express my sincere gratitude to my advisors Ivan Menezes and Leonardo Duarte, for their guidance and assistance during this research. Exploring the challenging aspects of programming a reservoir simulator was a very inspiring experience.

Special thanks to Waldemar Celes Filho for giving me the opportunity to work in the project that originated this research.

Thanks to all my colleagues at PUC-Rio. I would like to express my special appreciation to Henrique Santiago, Matheus Hoffmann, and Lucas Vivian for their friendship and company. Having close friends has been a gift through these years.

Thanks to my parents, Horácio and Valléria, for their endless love, support, and unwavering belief in me. Thank you also to my brother Rafael for the constant support and for being an inspiration as a researcher.

Thanks to Ivonett and Elio Accardo for their support and advice during difficult times. Being part of your family is one of the greatest things that happened to me.

Above all, I would like to thank my fiancée, Brunna Accardo, for her unconditional love and constant support throughout this journey. Thank you for being my best friend.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

PUC-Rio - Certificação Digital Nº 1912763/CA

Abstract

Bastos, Thiago Sousa; Menezes, Ivan Fábio Mota de (Advisor); Duarte, Leonardo Seperuelo (Co-Advisor). **Development of a multipurpose reservoir simulator based on a plugin architecture**. Rio de Janeiro, 2021. 97p. Dissertação de mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

During the last decades, large investments were made towards the development of numerical models and methods to forecast and analyze the different aspects of oil and gas recovery. In this context, modern simulators must be able to incorporate a wide variety of options to answer questions related to reservoir management accurately and effectively. In this work, we present a reservoir simulator based on a plugin architecture, where different formulations, solvers, and models can be developed, extended, and enhanced. With this approach, we use the black-oil model to implement traditional and state-of-the-art techniques, including fully- and adaptive-implicit methods, heuristic and PID time-step controllers, Newton-Raphson and Inexact Newton, and C1-continuous and conventional phase-potential single-point upstream weighting. Several plugin configurations were tested and validated with commercial simulators, and their performances were used to determine which are the most suitable to solve multiphase flow problems.

PUC-Rio - Certificação Digital Nº 1912763/CA

Keywords

Reservoir simulation   Black-Oil model   Multiphase-flow   Porous media   Plugin-based framework

Resumo

Bastos, Thiago Sousa; Menezes, Ivan Fábio Mota de; Duarte, Leonardo Seperuelo. **Desenvolvimento de um simulador numérico de reservatórios baseado em uma arquitetura de plugins**. Rio de Janeiro, 2021. 97p. Dissertação de Mestrado – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Nas últimas décadas, grandes investimentos foram feitos no desenvolvimento de modelos e métodos numéricos para prever e analisar os diferentes aspectos do processo de recuperação de óleo e gás. Neste contexto, os simuladores modernos devem ser capazes de incorporar uma grande variedade de opções para responder questões relacionadas ao gerenciamento de reservatórios de forma rápida e precisa. Neste trabalho, nós apresentamos um simulador de reservatórios baseado em uma arquitetura de plugins, onde diferentes formulações, *solvers* e modelos podem ser desenvolvidos, estendidos e aprimorados. A partir desta abordagem, utilizamos o modelo *black-oil* para implementar técnicas tradicionais e do estado da arte, como os métodos totalmente e adaptativamente implícito, os métodos de Newton-Raphson e Newton Inexato, controladores heurístico e PID para passo de tempo adaptativo e aproximações de fluxo de um ponto baseados no potencial de fase tradicional e C1-contínuo. Diversas configurações de plugins foram testadas e validadas com simuladores comerciais e seus desempenhos foram utilizados para determinar quais as mais adequadas para resolver problemas de escoamento multifásico.

PUC-Rio - Certificação Digital Nº 1912763/CA

Palavras-chave

Simulação de reservatórios Modelo black-oil Escoamento multifásico Meios porosos Sistema baseado em plugin

Table of contents

PUC-Rio - Certificação Digital Nº 1912763/CA

List of figures

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

List of tables

PUC-Rio - Certificação Digital Nº 1912763/CA

List of Abreviations

3D – Three-dimensional	
AIM – Adaptive Implicit Method	
AMG – Algebraic Multi-Grid	
API – Application Programming Interface	
BHP – Bottom-Hole Pressure	
BICGSTAB – Bi-Conjugate Gradient Stabilized	
C1-PPU – C1-continuous Phase Potential Upwind	
CFL – Courant–Friedrichs–Lewy	
CPR – Constrained Pressure Residual	
CSC – Compressed Sparse Column	
CSR – Compressed Sparse Row	
DAE – Discrete Algebraic Equations	
DRS – Dynamic Row Sum	
EOS – Equation Of State	
FIM – Fully Implicit Method	
FVF – Formation Volume Factor	
GMRES – Generalized Minimal Residuals	
GSim – GERESIM Simulator	
HC – Hydrocarbon Components	
HU – Hybrid Upwind	
ILU – Incomplete LU Factorization	
IMPES – Implicit in Pressure and Explicit in Saturations	
MPFA – Multi-Point Flux Approximation	
MSFV – Multiscale Finite Volume	
NF – Nested Factorization	
OOP – Object-Oriented Programming	
ORTHOMIN – Orthogonal Minimization	
PDE – Partial Differential Equation	
PPU – Phase Potential Upwind	
PVT – Pressure-Volume-Temperature	
SOA – Service-Oriented Architecture	
SFI – Sequential Fully Implicit	
TPFA – Two-Point Flux Approximation	

PUC-Rio - Certificação Digital Nº 1912763/CA

1 Introduction

The oil industry is known as one of the most demanding for large-scale simulations due to the large amounts of money and time usually invested on complex operations for oil and gas production, especially on offshore fields. As oil and gas are frequently found deep underground and covering a wide areal extension, it is not possible to have a complete deterministic characterization of the rock and fluid properties in the entire reservoir. The job of a reservoir engineer is to predict and study this system using indirect tools and multiscale field measurements to envision the best solution to optimize oil and gas production.

The process of modeling and simulating flow in porous media involves many assumptions and approximations. A petroleum reservoir is an anisotropic and heterogeneous medium with extensions in the order of kilometers. These characteristics associated with the behavior of rocks and fluids therein lead to nonlinear models. Numerically modeling the behavior of petroleum reservoirs usually involves discretizing partial differential equations in time and space, leading to large systems of discrete algebraic equations. The effects of the nonlinearities and the size of real fields result in computationally expensive studies, requiring the engineers to carefully choose the model and the numerical methods. The process of choosing the most suitable combination of models and numerical approximations often involves several simulation runs, using adequate upscaling techniques.

In this context, a general-purpose reservoir simulator with a flexible architecture is an essential tool, as it provides various features needed to properly represent the reservoirs. It enables the user to test various recovery strategies, design optimal development plans, and understand whether the exploration is profitable. Furthermore, the modularity of such architecture enables the developers to add new functionalities rapidly and reduce maintenance costs. The primary goal of this research is to study and develop a flexible multipurpose reservoir simulator, including its formulation, design, and implementation details.

PUC-Rio - Certificação Digital Nº 1912763/CA

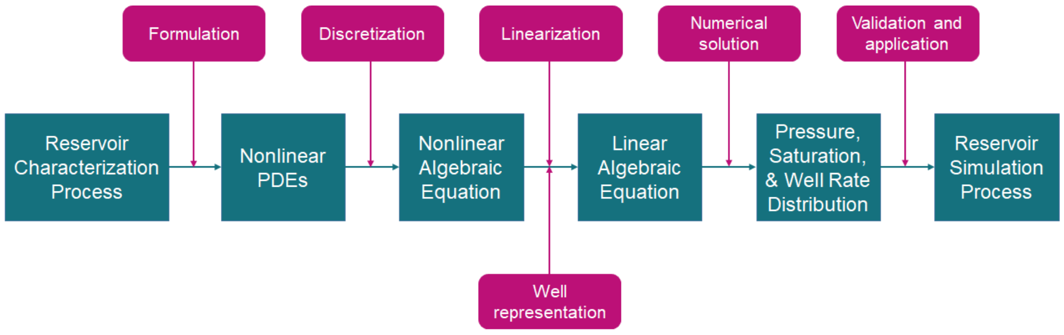
Formulation
Discretization
Linearization
Numerical solution
Validation and application
Reservoir Characterization Process
Nonlinear PDEs
Nonlinear Algebraic Equation
Linear Algebraic Equation
Pressure, Saturation, & Well Rate Distribution
Reservoir Simulation Process
Well representation

Figure 1.1: Major stages of a reservoir simulation, adapted from [1].

1.1 Overview of current developments in reservoir simulation

Reservoir simulation uses conservation laws to describe flow occurring in the reservoir porous media. Fig. 1.1 describes this general procedure in six stages. Firstly, the reservoir engineer characterises the petroleum reservoir to be modeled. A set of nonlinear partial differential equations derived from the chosen model is then discretized in both time and space, resulting in a system of nonlinear discrete algebraic equations (DAEs). Afterward, the simulator initiates iterative processes which linearizes the DAEs, applies boundary conditions, includes source/sink terms to represent wells, uses linear solvers, and leads to pressure and saturation distributions at various timesteps.

Over the years, several techniques have been developed in all stages of the simulation, making the simulators more accurate, realistic, and robust. Among these techniques, we can highlight the following:

PUC-Rio - Certificação Digital Nº 1912763/CA

Gridding

In the beginning, rectangular cartesian grids were predominant in reservoir simulation for general problems, and radial grids for coning problems. To obtain good resolutions in the vicinity of wellbore, highly heterogeneous rocks, or cells with a wide variation of saturation, researchers proposed cartesian and hybrid local grid refinements [2, 3]. Later, corner-point grids improved the capacity to represent complex geological structures and faults [4]. Currently, much effort has been made towards unstructured meshes since, unlike cartesian and corner-point grids, they are able to model fractured systems accurately [5–9].

PUC-Rio - Certificação Digital Nº 1912763/CA

Models

Historically, most simulations were based on the black-oil model, where the system is composed of three pseudo-components and three phases known as water, oil, and gas. It assumes that the flow is isothermal and has a pressure-independent composition of oil and gas; there is no volatility of the oil component in the gas phase, and the hydrocarbons cannot solubilize in water [10]. Later, EOR techniques thrived in response to the 1970's oil crisis, which drove the development of the compositional model. This model describes the reservoir hydrocarbon as a mixture of N-components, where flash behavior is significant and allows the simulation of volatile oil reservoirs [11].

Numerical methods

In general, the set of PDEs derived from a model cannot be solved using analytical methods, requiring numerical approximations. Traditionally, conventional reservoir simulators use the finite difference method (FDM) with single-point upstream weighting. With the popularization of unstructured and non-orthogonal grids and the possibility of using the complete permeability tensor, the finite volume [12] and finite element [13] methods, in addition to multi-point flux approximations [14], have become more common.

Solution schemes

A reservoir model contains thousands or even billions of time-dependent variables [15] . The level of implicitness of the solution scheme to solve this set of equations must have a proper balance between stability and efficiency, i.e., the lowest level of implicitness is the most computationally efficient, and the highest level is the most stable. In this context, fully-explicit methods cannot be efficiently exploited, as field-scale analyses are made for long periods of time (e.g., 20 years) and, at the same time, the Courant–Friedrichs–Lewy (CFL) condition restricts the maximum allowable step-size, leading to a prohibitive number of timesteps [13].

Alternatively, we have the fully-implicit method (FIM), which solves equations implicitly and simultaneously [16]. This scheme is utilized in complex analyses with large timesteps, where the main drawback is the size of the resulting matrix, which is not viable for applications with many chemical species even with today's computing power [13].

Another option is to use a scheme that is more stable than the fully-explicit without significantly sacrificing computation. This can be achieved by using the implicit in pressure and explicit in saturations scheme (IMPES) [17]. This method is widely used in the industry to solve two-phase and intermediate complexity problems.

A middle ground between the two is the adaptive-implicit method (AIM) [18]. This scheme aims to identify the cells that require an implicit treatment, while the remainder are implemented with the IMPES scheme to reduce the computational cost.

Lastly, there is the sequential fully-implicit method (SFI) [19]. It has the objective of having better stability than IMPES and, at the same time, having a lower computational cost than FIM. This is accomplished by constructing and solving the pressure equations first and then solving the transport (saturation or composition) equations, both implicitly. There is a growing interest in this scheme in the context of Multiscale Finite Volume (MSFV) compositional simulations [20, 21].

PUC-Rio - Certificação Digital Nº 1912763/CA

Linear solvers

The matrices arising in numerical reservoir models are sparse, highly ill-conditioned, and non-symmetric. Furthermore, the solution of linear systems is the most computationally expensive process in the simulation. For these reasons, the choice of the best linear solutions has been one of the main areas of research in the past decades.

Direct methods were used at first, but their scalability is considerably limited because of the high number of operations required to obtain the solution, reaching the order of $\mathcal{O}(m^3)$ for a matrix with dimension m. To improve these solvers, some investigations proposed ordering schemes that take advantage of the matrix structure, e.g., Red-Black, D2, and D4 [22, 23]. Even so, the improvements were not satisfactory, leading to the adoption of iterative methods. Among the several iterative solvers studied over the years [24–26], three Krylov subspace algorithms stand out: Orthomin [27], GMRES [28], and BICGSTAB [29]. Although these are the most advanced solvers in the petroleum industry nowadays, their performance depends upon proper preconditioning [13].

There are two main approaches to construct preconditioners. One is based on universally applicable algebraic methods, such as Jacobi, Gauss-Seidel, Incomplete LU factorization, and AMG (Algebraic Multi-grid) [30]. The other is by using available information about the analysis, geometry, and physics of the problem. A variety of alternatives exists for this class of algorithms, which include Nested Factorization (NF) [31], Constrained Pressure Residual (CPR) [32, 33], and Dynamic Row Sum (DRS) [34].

1.2 Survey of software architectures in modern simulation

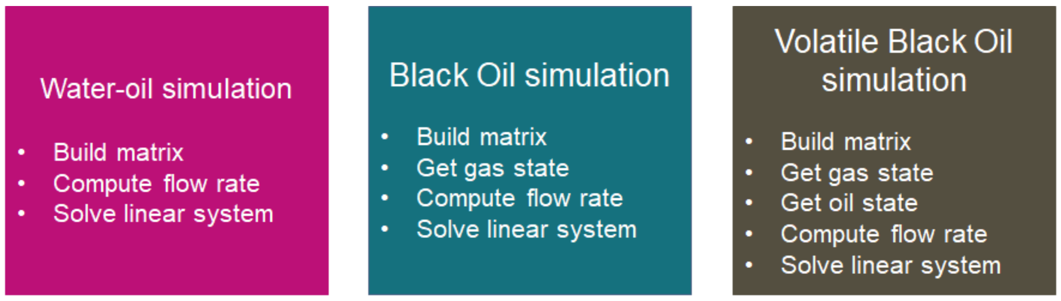

Figure 1.2: Monolithic architecture.

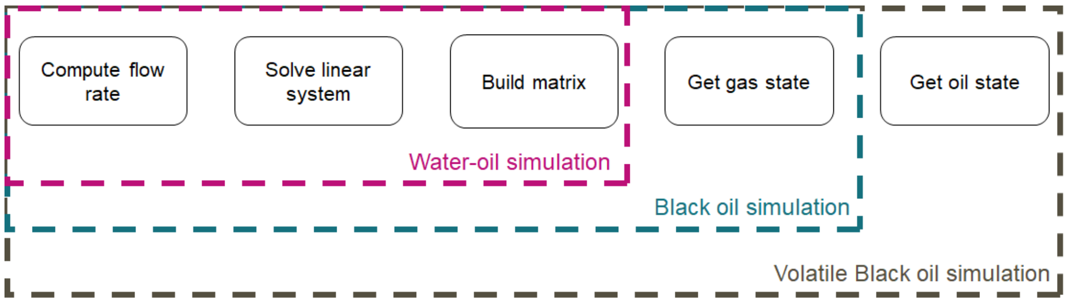

Figure 1.3: Service-oriented architecture.

PUC-Rio - Certificação Digital Nº 1912763/CA

In the early stages of scientific computing, the standard way of designing software was by using a monolithic architecture [35, 36]. Figure 1.2 illustrates this design decision, where an application is a box that performs a pipeline sequence of operations in the data. Although this paradigm seems simple and easy to visualize the whole process, it leads to multiple codes and many executables to treat every variation of the simulated process (i.e., code redundancy); higher executable size; difficult integration, customization, and extension with other software; and the impossibility of performing viable systematic comparisons between programs [37].

Studies have found that 80% of the software costs include adding and modifying features, bug fixing, and design improvements [38, 39]. As a result, developers proposed a new software model based on object-oriented programming, known as service-oriented architecture (SOA) [40–48]. This programming model focuses on breaking each process into smaller blocks of tasks and functions as services to promote loose coupling, reusability, interoperability, and technology-independence. Figure 1.3 shows the SOA, which has many benefits, such as reduced costs, greater return on investment, reduced time to market, reuse, and integration between services and traditional systems [49, 50].

Recently, Duarte [51] introduced Topsim, a plugin-based framework for large-scale numerical simulations. This framework uses a control script to load user-defined software components (in the form of plugins) at runtime. This paradigm has many advantages, namely, smaller executable size than traditional OOP and the capability to compare different numerical procedures changing only specific plugins. With such advantages, we decided to build our simulator upon this framework.

PUC-Rio - Certificação Digital Nº 1912763/CA

1.3 Objectives and contribution

The primary objective of this work is to present a multipurpose reservoir simulator named GERESIM Simulator (GSim). We propose a plugin architecture based on Topsim to provide a modular tool that allows developers to gather different techniques without hindering the maintenance and growth of the application. Our approach does not require an in-depth understanding of each plugin or major changes to the existing code, which facilitates extending and adding functionalities. In particular, this study is concerned with the following features,

- A black-oil model considering phase appearance and disappearance, compressibility, and capillarity effects;
- Fully- and adaptive-implicit schemes;
- Adaptive timestepping;
- Multi-layered wells;
- Support for water and gas injection;
- Support for scheduling opening and closening of wells, in addition to changing its operating conditions according to flow and pressure restrictions;
- Present multiple configurations of linear and nonlinear solvers;

The main contribution of this research is a three-dimensional multiphase reservoir simulator with a wide range of configurations. We present both classic and state-of-the-art methods for different test cases.

1.4 Thesis outline

The remainder of this work is structured as follows. Chapter 2 presents the mathematical formulation of our black-oil implementation. Chapter 3 details the design of GSim, its data structures, and possible plugin combinations. Chapter 4 compares the results of our application against benchmarks and commercial simulators. Also, we examine different configurations to show how one can use our modular approach to test different models and numerical methods for a specific application. Finally, we present our conclusions and suggestions for future research in Chapter 5.

PUC-Rio - Certificação Digital Nº 1912763/CA

2
Multiphase flow and numerical solution

This chapter describes the equations of simultaneous flow of three phases (water, oil, and gas) through the porous medium and its coupling with source/sink models. We explore the computation of rock and fluid properties to build the system of partial differential equations resulting from the black-oil model. Finally, we discuss the different numerical methods for the solution of the system.

2.1
Governing equations and physical properties

The black-oil model consists of, at most, three phases (oil, water, and gas) transporting three reservoir fluids through the porous medium. Figure 2.1 shows this representation, where the oil and water components are immiscible, and there is mass transfer of the gas component between the oil and gas phases.

For the sake of clarity, we identify components with uppercase and phases with lowercase subscripts. Further, we shall use the subscript s to denote standard conditions.

PUC-Rio - Certificação Digital Nº 1912763/CA

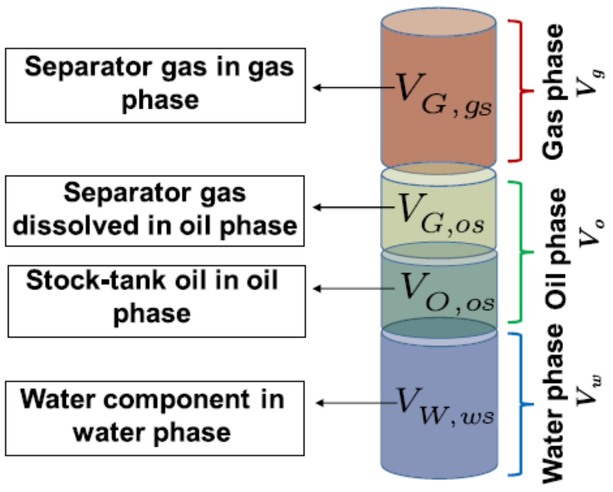
Separator gas in gas phase
$V_{G,gs}$
Gas phase V_g
Separator gas dissolved in oil phase
$V_{G,os}$
Stock-tank oil in oil phase
$V_{O,os}$
Oil phase V_o
Water component in water phase
$V_{W,ws}$
Water phase V_w

Figure 2.1: Conceptual model of the black-oil formulation [52].

2.1.1 Conservation of mass

The continuity equations in a porous medium $\Omega \subset \mathbb{R}^3$ on standard volumes are

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) = -\frac{\partial}{\partial x_i}\left(\frac{A_i}{B_w} u_{wi}\right)\Delta x_i + q_w \tag{2-1}$$

for the water component,

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) = -\frac{\partial}{\partial x_i}\left(\frac{A_i}{B_o} u_{oi}\right)\Delta x_i + q_o \tag{2-2}$$

for the oil component, and

$$V_b \frac{\partial}{\partial t}\left(\phi\left[\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o}\right]\right) = -\frac{\partial}{\partial x_i}\left(A_i\left[\frac{1}{B_g}u_{gi} + \frac{R_{so}}{B_o}u_{oi}\right]\right)\Delta x_i + q_g \tag{2-3}$$

for the gas component, where V_b is the bulk volume of the control volume; ϕ is the rock porosity; S_w, S_o, and S_g are phase saturations; R_{so} is the solution gas-oil ratio; B_w, B_o, and B_g are the formation volume factors (FVFs) for each phase; A_i, Δx_i, u_{wi}, u_{oi}, and u_{gi} are, respectively, the area normal to the flow, the length of the control volume and phase velocities in the x_i-direction; and q_w, q_o, and q_g are component volumetric flow rates. A more detailed description of these quantities will be presented later.

2.1.2 Conservation of momentum

Darcy's law is an empirical equation that relates the phase velocity with the pressure gradient in a porous medium. It can be expressed as

$$u_{\alpha i} = -\frac{k_{ii}k_{r\alpha}}{\mu_\alpha}\frac{\partial \Phi_\alpha}{\partial x_i}, \qquad \alpha = w, o, g \tag{2-4}$$

where k_{ii} is the absolute permeability tensor, $k_{r\alpha}$, u_α and Φ_α are the relative permeability, viscosity and phase potential of phase α, respectively. The latter is defined as

$$\Phi_\alpha = P_\alpha - \rho_\alpha g Z, \qquad \alpha = w, o, g \tag{2-5}$$

where P_α and ρ_α are, respectively, pressure and density of phase α, g is the acceleration of gravity, and Z is the elevation from datum, with positive values downward.

PUC-Rio - Certificação Digital Nº 1912763/CA

2.1.3 Rock and fluid properties

Fluid saturation

Saturation of a fluid is the ratio of the fluid volume to the pore volume

$$S_\alpha = \frac{V_\alpha}{V_p} \qquad \alpha = w, o, g \tag{2-6}$$

where S_α and V_α are, respectively, the saturation and volume of phase α, and V_p is the pore volume.

Considering the interconnected pores of the media to be completely filled with fluid, leads to

$$S_w + S_o + S_g = 1. \tag{2-7}$$

Porosity

Porosity is the volumetric fraction of interconnected pores within the solid matrix (reservoir rock), i.e, the ratio of the pore volume to the bulk volume. Conventionally, the rock is assumed to be slightly compressible. From this, one may define the rock compressibility as follows,

$$c_R = \frac{1}{\phi}\frac{d\phi}{dP}. \tag{2-8}$$

After integration, we have

$$\phi = \phi_0 \exp[c_R(P - P_0)] \tag{2-9}$$

where P_0 is a reference pressure and ϕ_0 is the porosity at P_0.

Using a first-order Taylor series expansion, Equation (2-9) may be approximated by

$$\phi \approx \phi_0[1 + c_R(P - P_0)]. \tag{2-10}$$

Capillary pressure

Because of surface tension between two fluids within pores, the nonwetting fluid pressure is higher than the wetting fluid pressure. The difference between these pressures is the capillary pressure. For a three-phase flow, we use a generic formulation where any phase can be wetting, intermediate, and non-wetting. Thus:

$$P_{cow}(S_w) = P_o - P_w, \tag{2-11}$$

PUC-Rio - Certificação Digital Nº 1912763/CA

$$P_{cgo}(S_g) = P_g - P_o \tag{2-12}$$

where P_{cow} and P_{cgo} are, respectively, the oil-water and gas-oil capillary pressures.

The gas-water capillary pressure is a simple combination of the other two:

$$P_{cgw} = P_g - P_w = P_{cgo} + P_{cow}. \tag{2-13}$$

Moreover, we accept as an empirical fact that these pressures are unique functions of saturation, in other words, $P_{cow} = f(S_w)$ and $P_{cgo} = f(S_g)$ [53].

Absolute and relative permeabilities

Permeability is the capability of the porous media of transmitting fluids through their pores. When a single phase completely fills the medium, this capability is called absolute permeability. The absolute permeability is a directional property, and can be represented as a tensor. For simplicity, we assume the coordinate system to be aligned with the principle axes of this tensor. Thus,

$$diag(k_{ii}) = [k_{xx} \quad k_{yy} \quad k_{zz}]. \tag{2-14}$$

In the case of multiphase flow, Darcy's law must be modified to consider the interference from one fluid in the others. This is represented as the fraction of the single phase that effectively flows through the medium for a given direction, called relative permeability.

For a three-phase flow, relative permeabilities are determined experimentally. From these, the relative permeabilities for the wetting and nonwetting phases are functions of their respective saturations in a two-phase system [13]:

$$k_{rw} = f(S_w), \quad k_{rg} = f(S_g), \tag{2-15}$$

and the intermediate phase (oil) relative permeability is function of two independent saturations

$$k_{ro} = f(S_w, S_g). \tag{2-16}$$

In practice, the form of the function $f(S_w, S_g)$ is rarely known. For this reason, two sets of two-phase relative permeability data (oil-water and gas-oil systems in the presence of irreducible water) are used to estimate k_{ro}. This

PUC-Rio - Certificação Digital Nº 1912763/CA

work adopts Stone's model II [54] for this estimate:

$$k_{ro} = k_{rc}\left[\left(\frac{k_{row}}{k_{rc}} + k_{rw}\right)\left(\frac{k_{rog}}{k_{rc}} + k_{rg}\right) - (k_{rw} + k_{rg})\right] \tag{2-17}$$

where $k_{rc} = f(S_{wc})$ is the irreducible water relative permeability, $k_{row} = f(S_w)$ is the intermediate and wetting system relative permeability, and $k_{rog} = f(S_g)$ is the nonwetting and intermediate system relative permeability.

Fluid properties

Fluid properties are determined from thermodynamic state variables, such as pressure and temperature. These properties are provided through mathematical formulas or interpolated values via PVT (Pressure-Volume-Temperature) tables after characterizing the fluids.

There is a large influence of the states of the hydrocarbon phases on these properties. These states are determined from the bubble-point pressure. This pressure is defined as the pressure at which the first bubble of gas appears at a specific temperature [55]. If the bubble-point pressure is below the reservoir pressure, we have the undersaturated state, where there is no free gas, and the solution gas component is in the oil phase. Otherwise, we have the saturated state, where the three phases coexist, and the reservoir pressure is the bubble-point pressure. To measure the amount of solution gas, we define the solution gas-oil ratio,

$$R_{so} = \frac{V_{Gs}}{V_{Os}} \tag{2-18}$$

which is the ratio of the volume of gas V_{Gs} dissolved at a given reservoir pressure and temperature in a unit volume of stock-tank oil V_{Os}.

Another important property is the formation volume factor (FVF), which converts volumes at reservoir conditions to volumes at standard conditions. That is, FVF is the ratio of the volume that the phase occupies at *in situ* conditions to that at standard conditions:

$$B_\alpha = \frac{V_\alpha}{V_{\alpha s}} \qquad \alpha = w, o, g. \tag{2-19}$$

Considering W_O and W_G as the weight of oil and gas components, the mass fraction of oil and gas components in the oil phase are:

$$C_{Oo} = \frac{W_O}{W_O + W_G} = \frac{\rho_{Os}}{B_o \rho_o} \tag{2-20}$$

PUC-Rio - Certificação Digital Nº 1912763/CA

$$C_{Go} = \frac{W_G}{W_O + W_G} = \frac{R_{so}\rho_{Gs}}{B_o\rho_o}. \tag{2-21}$$

Combining equations (2-20) and (2-21) with $C_{Oo} + C_{Go} = 1$ results in

$$\rho_o = \frac{\rho_{Os} + R_{so}\rho_{Gs}}{B_o}. \tag{2-22}$$

The gas formation volume factor B_g is the ratio of the volume of free gas at reservoir conditions to the volume of the gas component at standard conditions:

$$B_g = \frac{V_g}{V_{Gs}}. \tag{2-23}$$

Using the weight of free gas $W_g = W_G$, $V_g = W_G/\rho_g$ and $V_{Gs} = W_G/\rho_{Gs}$, we obtain

$$\rho_g = \frac{\rho_{Gs}}{B_g}. \tag{2-24}$$

Finally, with the water FVF we can define the water density as

$$\rho_w = \frac{\rho_{Ws}}{B_w}. \tag{2-25}$$

Using the previous definitions, we can classify fluids as incompressible, slightly compressible, and compressible. In multiphase flow in petroleum reservoirs, water and undersaturated oil are treated as slightly compressible. Thereby, their FVFs and viscosities are given by

$$B_w = B_{wi}[1 - c_w(P_o - P_{w0})], \tag{2-26}$$

$$B_o = B_{ob}[1 - c_o(P_o - P_b)], \tag{2-27}$$

$$\mu_w = \mu_{wi} + c_{vw}(P_o - P_{w0}), \tag{2-28}$$

$$\mu_o = \mu_{ob} + c_{vo}(P_o - P_b) \tag{2-29}$$

where B_{wi} and μ_{wi} are the FVF and viscosity at initial formation pressure P_{w0}, c_w and c_{vw} are, respectively, the water compressibility and viscosibility. Also, B_{ob} and μ_{ob} are the oil FVF and viscosity at bubble-point pressure P_b. Finally, c_o and c_{vo} are the oil compressibility and viscosibility.

PUC-Rio - Certificação Digital Nº 1912763/CA

Furthermore, for the saturated state, the quantities B_o, R_{so}, μ_o, u_g, and B_g are established from experimental data.

2.2 Numerical solution of the Black-Oil model

Combining Equations (2-1)-(2-4) with the rock/fluid properties, we obtain

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) = \frac{\partial}{\partial x_i}\left(k_{ii} A_i \frac{k_{rw}}{\mu_w B_w}\frac{\partial \Phi_\alpha}{\partial x_i}\right)\Delta x_i + q_w \tag{2-30}$$

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) = \frac{\partial}{\partial x_i}\left(k_{ii} A_i \frac{k_{ro}}{\mu_o B_o}\frac{\partial \Phi_o}{\partial x_i}\right)\Delta x_i + q_o \tag{2-31}$$

$$V_b \frac{\partial}{\partial t}\left[\phi\left(\frac{S_g}{B_g} + \frac{R_{so} S_o}{B_o}\right)\right] = \frac{\partial}{\partial x_i}\left[k_{ii} A_i \left(\frac{k_{rg}}{\mu_g B_g}\frac{\partial \Phi_g}{\partial x_i} + \frac{R_{so} k_{ro}}{\mu_o B_o}\frac{\partial \Phi_o}{\partial x_i}\right)\right]\Delta x_i + q_g. \tag{2-32}$$

The left-hand side of Equations (2-30)-(2-32) are referred to as the accumulation terms, while the terms on the right-hand side are the flux and source/sink terms, respectively.

Traditionally, commercial simulators such as IMEX [56] and Eclipse [57] use natural variables, i.e., pressures and saturations as primary unknowns. IMEX uses P_o, S_w, and S_o for the saturated state and P_o, S_w, and P_b for the undersaturated state, while Eclipse uses P_o, S_w, and S_g for the saturated state and P_o, S_w, and R_{so} for the undersaturated state. Since we are using Chen et al.'s approach [13], we chose the same set of primary variables as IMEX.

From this point in the text, the indices i, j, k will represent the unit vectors of x, y, and z coordinates, respectively, instead of the previous index notation. Also, we will use the superscript n to represent discrete time levels and subscript n to represent a cell.

2.2.1 Discretization of the accumulation term

The accumulation terms of Equations (2-30)-(2-32) can be discretized over a cell as

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) \approx \frac{V_b}{\Delta t}\left[\left(\frac{\phi S_w}{B_w}\right)^{n+1} - \left(\frac{\phi S_w}{B_w}\right)^{n}\right]_{i,j,k} \tag{2-33}$$

PUC-Rio - Certificação Digital Nº 1912763/CA

$$V_b \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \approx \frac{V_b}{\Delta t}\left[\left(\frac{\phi S_o}{B_o}\right)^{n+1} - \left(\frac{\phi S_o}{B_o}\right)^{n}\right]_{i,j,k} \tag{2-34}$$

$$\begin{aligned} V_b \frac{\partial}{\partial t}\left[\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o}\right)\right] \approx \frac{V_b}{\Delta t}\Bigg\{&\left[\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o}\right)\right]^{n+1} \\ &- \left[\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o}\right)\right]^{n}\Bigg\}_{i,j,k} \end{aligned} \tag{2-35}$$

2.2.2 Discretization of the flux term

The finite difference method for a seven-point stencil (3D) scheme in a block-centered grid (Figure 2.2) leads to

$$\begin{aligned} &\left(\frac{k_{xx}A_x k_{ro}}{\Delta x \mu_o B_o}\right)_{i+1/2,j,k} \left(\Phi_{o_{i+1,j,k}} - \Phi_{o_{i,j,k}}\right) \\ +&\left(\frac{k_{xx}A_x k_{ro}}{\Delta x \mu_o B_o}\right)_{i-1/2,j,k} \left(\Phi_{o_{i-1,j,k}} - \Phi_{o_{i,j,k}}\right) \\ +&\left(\frac{k_{yy}A_y k_{ro}}{\Delta y \mu_o B_o}\right)_{i,j+1/2,k} \left(\Phi_{o_{i,j+1,k}} - \Phi_{o_{i,j,k}}\right) \\ +&\left(\frac{k_{yy}A_y k_{ro}}{\Delta y \mu_o B_o}\right)_{i,j-1/2,k} \left(\Phi_{o_{i,j-1,k}} - \Phi_{o_{i,j,k}}\right) \\ +&\left(\frac{k_{zz}A_z k_{ro}}{\Delta z \mu_o B_o}\right)_{i,j,k+1/2} \left(\Phi_{o_{i,j,k+1}} - \Phi_{o_{i,j,k}}\right) \\ +&\left(\frac{k_{zz}A_z k_{ro}}{\Delta z \mu_o B_o}\right)_{i,j,k-1/2} \left(\Phi_{o_{i,j,k-1}} - \Phi_{o_{i,j,k}}\right) \end{aligned} \tag{2-36}$$

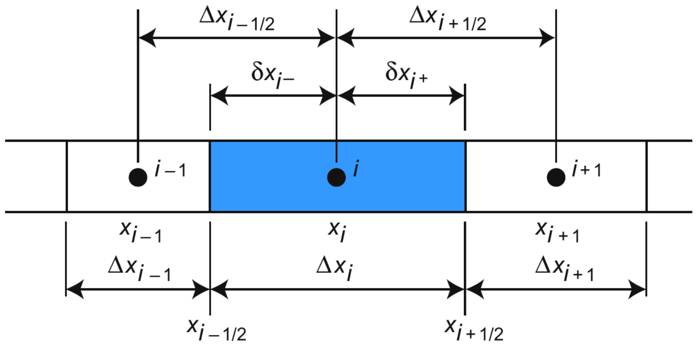

Figure 2.2: Schematic of a block-centered grid [58].

PUC-Rio - Certificação Digital Nº 1912763/CA

for the oil component, where

$$\Phi_{o_{i+1,j,k}} - \Phi_{o_{i,j,k}} = (P_{o_{i+1,j,k}} - P_{o_{i,j,k}}) - g\rho_{o_{i+1/2,j,k}}(Z_{i+1,j,k} - Z_{i,j,k}). \quad (2\text{-}37)$$

This discretization procedure is similar for water and gas.

We define

$$T_{\alpha x_{i\pm1/2,j,k}} = \left(\frac{k_{xx}A_x}{\Delta x}\frac{k_{r\alpha}}{\mu_\alpha B_\alpha}\right)_{i\pm1/2,j,k}, \qquad \alpha = w, o, g \quad (2\text{-}38)$$

as the transmissibility of phase α.

We can write the flow equations in a compact form [58]. Consider ψ_x, ψ_y, and ψ_z as the set of neighboring blocks of block n. Then, the neighborhood ψ_n is defined as the set that contains the neighboring blocks of n in all flow directions; that is, $\psi_n = \psi_x \cup \psi_y \cup \psi_z$. Along with this notation, one may write the phase potential difference as $\Delta\Phi_{ln} = \Phi_l - \Phi_n$ with l and n being the cell n and its neighbor l. Combining Equations (2-36) and (2-38) yields

$$\begin{aligned} &T_{o_{i+1/2,j,k}}\Delta\Phi_{o_{i+1/2,j,k}} + T_{o_{i-1/2,j,k}}\Delta\Phi_{o_{i-1/2,j,k}} + T_{o_{i,j+1/2,k}}\Delta\Phi_{o_{i,j+1/2,k}} \\ &+T_{o_{i,j-1/2,k}}\Delta\Phi_{o_{i,j-1/2,k}} + T_{o_{i,j,k+1/2}}\Delta\Phi_{o_{i,j,k+1/2}} + T_{o_{i,j,k-1/2}}\Delta\Phi_{o_{i,j,k-1/2}} \\ &\qquad = \sum_{l\in\psi_n} T_{o_{ln}}\Delta\Phi_{ln}. \end{aligned} \quad (2\text{-}39)$$

PUC-Rio - Certificação Digital Nº 1912763/CA

2.2.3 Treatment of interblock transmissibilities

In general, petroleum reservoirs are anisotropic and heterogeneous media. These characteristics provide numerical difficulties due to significant variations of certain properties from one block to another, such as the formation thickness, permeabilities, and porosity. For this reason, the computation of the transmissibilities at cell boundaries must be performed with care.

Since these properties are given only at block centers and the transmissibilities are calculated at the interface between two cells, averaging techniques must be employed. To this end, we can split the transmissibility into a geometric factor,

$$G_x \equiv \frac{k_{xx}A_x}{\Delta x}, \quad G_y \equiv \frac{k_{yy}A_y}{\Delta y}, \quad G_z \equiv \frac{k_{zz}A_z}{\Delta z} \quad (2\text{-}40)$$

and a mobility factor for phase α

$$\lambda_\alpha \equiv \left(\frac{k_{r\alpha}}{\mu_\alpha B_\alpha}\right) \qquad \alpha = w, o, g. \tag{2-41}$$

The geometric factor is estimated using harmonic averaging to ensure consistency with Darcy's law [19]:

$$\begin{aligned} G_{i\pm1/2,j,k} &= \frac{2A_{x_{i,j,k}} k_{xx_{i,j,k}} A_{x_{i\pm1,j,k}} k_{xx_{i\pm1,j,k}}}{A_{x_{i,j,k}} k_{xx_{i,j,k}} \Delta x_{i\pm1,j,k} + A_{x_{i\pm1,j,k}} k_{xx_{i\pm1,j,k}} \Delta x_{i,j,k}}, \\ G_{i,j\pm1/2,k} &= \frac{2A_{y_{i,j,k}} k_{yy_{i,j,k}} A_{y_{i,j\pm1,k}} k_{yy_{i,j\pm1,k}}}{A_{y_{i,j,k}} k_{yy_{i,j,k}} \Delta y_{i,j\pm1,k} + A_{y_{i,j\pm1,k}} k_{yy_{i,j\pm1,k}} \Delta y_{i,j,k}}, \\ G_{i,j,k\pm1/2} &= \frac{2A_{z_{i,j,k}} k_{zz_{i,j,k}} A_{z_{i,j,k\pm1}} k_{zz_{i,j,k\pm1}}}{A_{z_{i,j,k}} k_{zz_{i,j,k}} \Delta z_{i,j,k\pm1} + A_{z_{i,j,k\pm1}} k_{zz_{i,j,k\pm1}} \Delta z_{i,j,k}}. \end{aligned} \tag{2-42}$$

The phase mobilities, on the other hand, are averaged using upstream weighting. In this method, the mobility of a phase at the interface between two cells is equal to the mobility of the phase of the upstream cell. For instance, consider Figure 2.3. If there is a $(R_{so}\lambda_o)_{i+1/2,j,k}$ at boundary $< i+1/2, j, k >$ between cells $< i, j, k >$ and $< i+1, j, k >$, then

$$(R_{so}\lambda_o)_{i+1/2,j,k} = (R_{so}\lambda_o)_{i,j,k}, \tag{2-43}$$

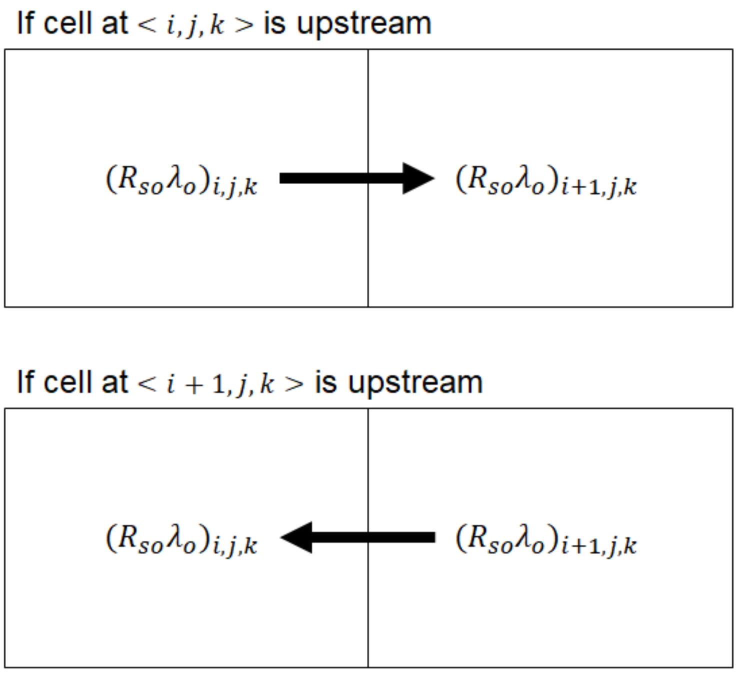
If cell at $< i, j, k >$ is upstream
$(R_{so}\lambda_o)_{i,j,k}$
$(R_{so}\lambda_o)_{i+1,j,k}$
If cell at $< i+1, j, k >$ is upstream
$(R_{so}\lambda_o)_{i,j,k}$
$(R_{so}\lambda_o)_{i+1,j,k}$

Figure 2.3: Example of single-point upstream weighting.

PUC-Rio - Certificação Digital Nº 1912763/CA

if oil flows from cell $< i, j, k >$ to cell $< i+1, j, k >$, and

$$(R_{so}\lambda_o)_{i+1/2,j,k} = (R_{so}\lambda_o)_{i+1,j,k} \tag{2-44}$$

if oil flows from cell $< i+1, j, k >$ to cell $< i, j, k >$ [19].

Among the different methods to determine the flow direction, Phase-Potential Upwind (PPU) schemes are the most widely used. In this approach, the phase potential gradient across the cell boundaries determines the upstream direction for that phase [53]. The mobilities at the cell boundaries are evaluated as

$$\lambda_{\alpha_{ln}} = \begin{cases} \lambda_{\alpha_n}, & \text{if } \Delta\Phi_{ln} < 0 \\ \lambda_{\alpha_l}, & \text{otherwise.} \end{cases} \tag{2-45}$$

Recent studies show that in the presence of strong buoyancy forces, the flow regimes switches between co-current and counter-current, and the upstream direction switches accordingly. As a result, the numerical flux becomes discontinuous, leading to ill-conditioned matrices and serious convergence problems in the nonlinear iterative process [59–62].

Lee et al. [63] proposed a C1-continuous numerical flux scheme named Hybrid Upwind (HU) to address this behavior. They showed that, by treating the viscous, buoyancy, and capillary fluxes differently, the numerical solution becomes smooth, locally conservative, monotonic, and physically consistent. The upwind direction for the viscous flux is always treated as co-current with respect to the total velocity, while the buoyancy and capillary fluxes are always counter-current and independent of the dynamic pressure field.

Given that HU only applies for the fractional flow formulation instead of the traditional formulation, Jiang and Younis [64] developed a different method based on the traditional PPU, called C1-PPU. They found that PPU can be written as

$$\lambda_{\alpha_{ln}} = \frac{\Delta\Phi_{ln}\lambda_{\alpha_n} - \Upsilon_{PPU}(\Delta\Phi_{ln})(\lambda_{\alpha_n} - \lambda_{\alpha_l})}{\Delta\Phi_{ln}} \tag{2-46}$$

where

$$\Upsilon_{PPU}(\Delta\Phi_{\alpha_{ln}}) = \max(\Delta\Phi_{\alpha_{ln}}, 0). \tag{2-47}$$

The flux switching function Υ_{PPU} is a non-differentiable function. From this, one may construct a C1-continuous function that is an approximation of

PUC-Rio - Certificação Digital Nº 1912763/CA

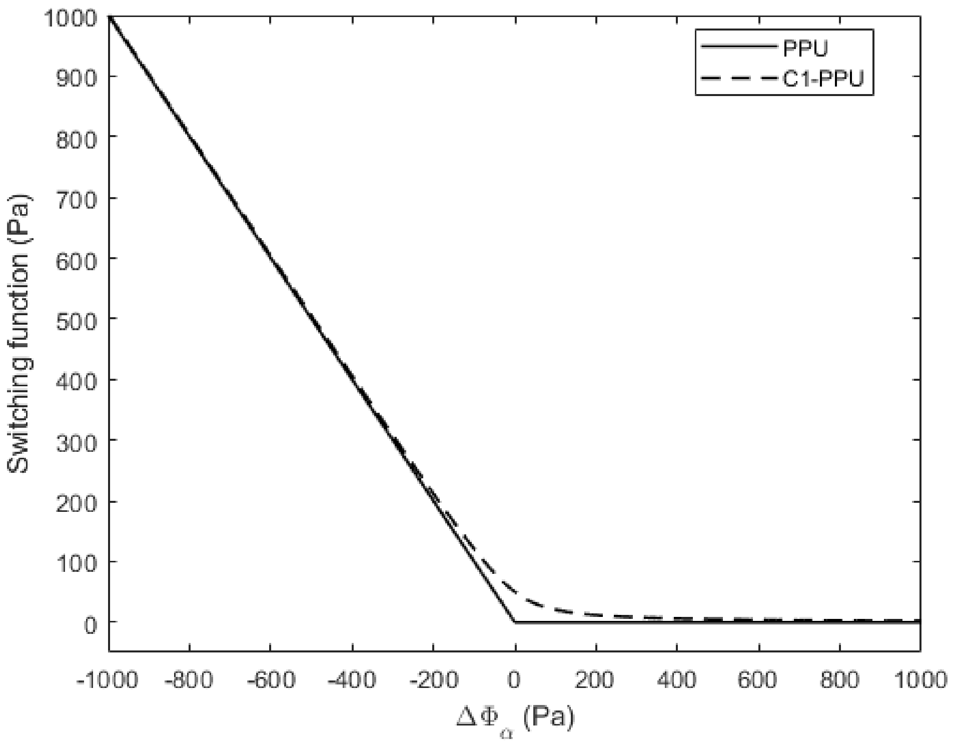

Figure 2.4: Approximation functions for PPU and C1-PPU schemes.

PUC-Rio - Certificação Digital Nº 1912763/CA

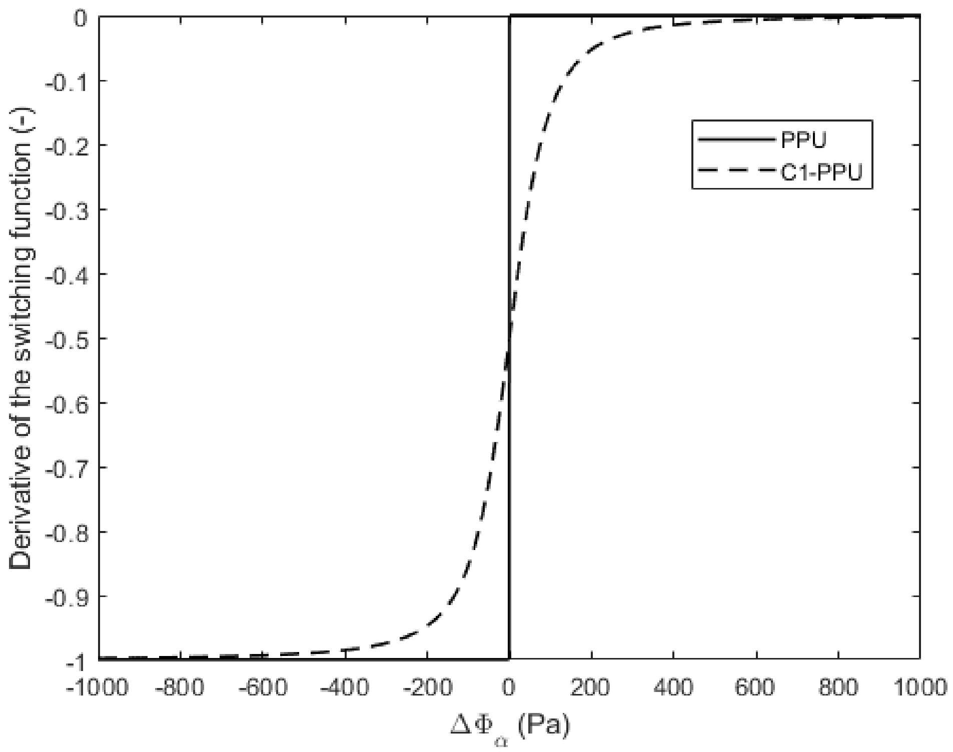

Figure 2.5: Derivatives of the approximation functions with respect to the phase potential for PPU and C1-PPU schemes.

Υ_{PPU}, for instance,

$$\Upsilon_{C1-PPU}(\Delta\Phi_{\alpha_{ln}}, \epsilon) = \frac{\Delta\Phi_{\alpha_{ln}} + \sqrt{\Delta\Phi^2_{\alpha_{ln}} + \epsilon^2}}{2} \tag{2-48}$$

where ϵ is a smoothing coefficient. Figures 2.4 and 2.5 shows the similarities between these two switching functions and their derivatives. The authors concluded that C1-PPU has a superior nonlinear convergence when compared to the PPU due to the smoothness of the numerical flux.

2.2.4 Treatment of source/sink terms

The source and sink terms are numerical representations of the wells. In this research, a simplified vertical wellbore is derived based on Peaceman's equations [65]. The flow rates of this formulation are calculated as follows:

$$q_\beta = -\sum_{\sigma=1}^{M_\omega} J_{\omega,\sigma}(P_o - P_{wf}) \qquad \beta = w, o, g \tag{2-49}$$

PUC-Rio - Certificação Digital Nº 1912763/CA

where P_{wf} is the flowing wellbore bottom-hole pressure (BHP), M_ω is the total number of perforated zones (completions) of the well, and $J_{\omega,\sigma}$ is the productivity or injectivity index of the σ-th perforated zone, given by

$$J_{\omega,\sigma} = (WI)_\sigma \left[\frac{1}{B_\kappa}\left(\frac{k_{rw}}{\mu_w} + \frac{k_{ro}}{\mu_o} + \frac{k_{rg}}{\mu_g}\right)\right] \qquad \kappa = w \text{ or } g \tag{2-50}$$

for injection wells, and

$$J_{\omega,\sigma} = \begin{cases} (WI)_\sigma\lambda_w & \text{if } q_\beta = q_w \\ (WI)_\sigma\lambda_o & \text{if } q_\beta = q_o \\ (WI)_\sigma(R_{so}\lambda_o + \lambda_g) & \text{if } q_\beta = q_g \end{cases} \tag{2-51}$$

for production wells.

The well index WI_σ of the σ-th completion is defined as

$$WI_\sigma = \left(\frac{2\pi h\sqrt{k_x k_y}}{\ln\left(r_{eq}/r_w\right) + s}\right) \tag{2-52}$$

where h is the thickness of the cell's completion, r_w is the wellbore radius, s is the skin factor. Finally, the equivalent radius r_{eq} is estimated as

$$r_{eq} = \frac{0.28\left[(k_y/k_x)^{1/2}(\Delta x)^2 + (k_x/k_y)^{1/2}(\Delta y)^2\right]^{1/2}}{[(k_y/k_x)^{1/4} + (k_x/k_y)^{1/4}]}. \tag{2-53}$$

In reservoir simulation, the flowing bottom-hole pressure of each perforated cell n is computed from a reference BHP. Neglecting the effects of friction and kinetic energy, the relationship between these pressures is defined as

$$P_{wf_n} = P_{wf_{ref}} + g\overline{\rho}_{well}(Z_n - Z_{wfref}) \tag{2-54}$$

where $P_{wf_{ref}}$ is the BHP at a reference depth Z_{wfref}, Z_n is the depth of the cell, and ρ_{well} is the average density in the well, given by

$$\overline{\rho}_{well} = \left(\frac{k_{rw}}{\mu_w B_w}\rho_w + \frac{k_{ro}}{\mu_o B_o}\rho_o + \frac{K_{rg}}{\mu_g B_g}\rho_g\right) / \left(\frac{K_{rw}}{\mu_w B_w} + \frac{K_{ro}}{\mu_o B_o} + \frac{K_{rg}}{\mu_g B_g}\right). \tag{2-55}$$

The wells are considered internal boundaries of the reservoir system, and, as such, one must specify its operational conditions to have a well-posed problem. This internal boundary condition can be specified in the form of bottom-hole pressure (Dirichlet-type boundary condition) or flow rate (Neumann-type boundary condition) [19, 66].

Each operational condition requires a specific treatment for equation (2-49). The injectors are specified either by the reference BHP, water or gas flow rate. Likewise, the producers are specified by the reference BHP, water, oil, gas, liquid, or total flow rates. For Dirichlet boundary conditions, the flow rates are the primary unknowns, and, for Neumann boundary conditions, the BHPs are the primary unknowns. The equations for each operational condition and their respective numerical treatment can be found in Ertekin et al. [19]. More details about how the source/sink terms are used to find the solution will be given in Section 2.2.6 of this chapter.

2.2.5 Initial Conditions

The initial conditions of the model involve the specification of phase pressures and saturations for all cells. Differences of density and capillary pressures segregate the fluids until capillary/gravity equilibrium is reached.

At the equilibrium state, there are five zones in the reservoir, as shown in Figure 2.6. The first zone is the gas cap zone, where only the gas phase is

PUC-Rio - Certificação Digital Nº 1912763/CA

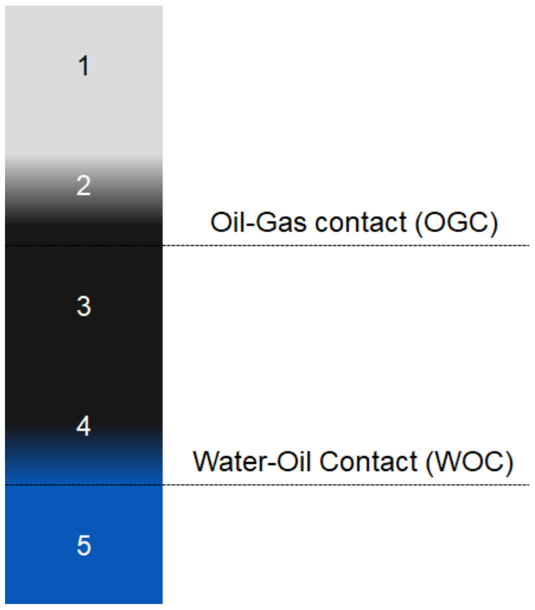

Figure 2.6: Fluid distribution in gravity-capillary equilibrium.

continuous. Therefore, the vertical distribution of the gas pressure is calculated from the hydrostatic gradient:

$$\frac{dP_g}{dz} = \rho_g g. \tag{2-56}$$

Additionally,

$$S_o = 0, \quad S_w = S_{wc}. \tag{2-57}$$

From these saturations expressed in Equation (2-57), other variables can be computed:

$$S_g = 1 - S_w - S_o, \tag{2-58}$$

$$P_o = P_g - P_{cgo}(S_{g,\max}), \tag{2-59}$$

$$P_w = P_o - P_{cow}(S_{wc}). \tag{2-60}$$

The second zone is the gas-oil transition zone, where both oil and gas phases are continuous. The vertical equilibrium depends not only on the gas pressure gradient (Equation (2-56)), but also on the oil pressure gradient:

$$\frac{dP_o}{dz} = \rho_o g. \tag{2-61}$$

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

From these conditions, we use the capillary pressures (Eqs. (2-11) and (2-12)) to determine the saturation distribution.

The oil phase is continuous in the third (oil) zone. Thus, we have $S_w = S_{wc}$, $S_g = 0$, and the pressures follow Equations (2-61), (2-11), and (2-12).

Both the oil and water are continuous in the oil-water transition zone (fourth zone), so the vertical pressure distribution of these phases depends on Equation (2-61) and

$$\frac{dP_w}{dz} = \rho_w g. \tag{2-62}$$

Moreover, we must determine the saturations from capillary pressure relationships.

The last zone is named water zone, where water is the only phase present. Consequently, oil and gas saturations are null, the water pressure is given by the hydrostatic gradient (Equation (2-62)), and oil pressure is found by using the oil-water capillary pressure relationship (Equation (2-11)).

In reservoir simulation, the depths of water/oil and oil/gas contacts are given by the user. Then the initial pressure and saturation at all cells can be determined if a reference pressure and a reference depth are given [67].

2.2.6 Solution of nonlinear equations

As described in Chapter 1, different solution methods are used to solve the system of equations. In this work, we adopt the adaptive- and fully-implicit methods (AIM and FIM). The first was chosen because is a good balance of efficiency and stability, while the second is required for solving more complex problems. In both methods, we use backward Euler for the accumulation and sink/source terms, while for the flux terms we use either backward or forward Euler depending whether the cell is IMPES or FIM, when using the AIM formulation. Since FIM is a subset of AIM, one may write the residual equations only for AIM:

$$R_W = -\frac{V_b}{\Delta t}\left[\left(\frac{\phi S_w}{B_w}\right)^{n+1} - \left(\frac{\phi S_w}{B_w}\right)^{n}\right]_n + \sum_{l\in\psi_n}(T^m_{w_{ln}}\Delta\Phi^{n+1}_{w_l}) + q_{w_n}^{n+1} \tag{2-63}$$

$$R_O = -\frac{V_b}{\Delta t}\left[\left(\frac{\phi S_o}{B_o}\right)^{n+1} - \left(\frac{\phi S_o}{B_o}\right)^{n}\right]_n + \sum_{l\in\psi_n}(T^m_{o_{ln}}\Delta\Phi^{n+1}_{o_l}) + q_{o_n}^{n+1} \tag{2-64}$$

$$R_G = -\frac{V_b}{\Delta t}\left\{\left[\phi\left(\frac{S_g}{B_g}+\frac{R_{so}S_o}{B_o}\right)\right]^{n+1} - \left[\phi\left(\frac{S_g}{B_g}+\frac{R_{so}S_o}{B_o}\right)\right]^{n}\right\}_n + \sum_{l\in\psi_n}\left[T_{gln}^{m}\Delta\Phi_{gl}^{n+1} + (R_{so}T_o)_{ln}^{m}\Delta\Phi_{ol}^{n+1}\right] + {q_g}_n^{n+1} \tag{2-65}$$

$$R_{well} = \begin{cases} (P_{wfref} - P_{wf_{spec}}), & \text{for pressure at a ref. depth specification} \\ (q_{computed} - q_{spec}), & \text{for flow rate specification} \end{cases} \tag{2-66}$$

where R_W, R_O, R_G, and R_{well} are the residual equations for water, oil, gas, and wells, respectively. The superscript m can be either $n+1$ (for FIM cells/neighbors) or n (for IMPES cells/neighbors). For both FIM and AIM formulations, we consider that cells perforated by wells are treated implicitly.

By using the Taylor series expansion in Equations (2-63)-(2-66), we derive the Newton method:

$$[\boldsymbol{J}]\boldsymbol{\delta X} = -\boldsymbol{R} \tag{2-67}$$

where the $[\boldsymbol{J}]$ is the Jacobian matrix, $\boldsymbol{\delta X}$ is the vector with the increments of the unknowns, and $\boldsymbol{R}$ is the vector of residuals.

Due to the dynamic nature of the AIM formulation, the structure of each member of Equation (2-67) changes depending whether the cell is FIM or IMPES. The residual vector for cell n is given by

$$\boldsymbol{R_n} = [R_{W_n} \quad R_{O_n} \quad R_{G_n}]^T \tag{2-68}$$

for FIM fluid equations and

$$\boldsymbol{R_n} = R_{W_n} + R_{O_n} + R_{G_n} \tag{2-69}$$

for IMPES fluid equations.

The well residual vector depends if the well is pressure or flow rate specified. In the former case, the flow rates can be explicitly calculated using Equation (2-49) and, therefore, there is no need of using the well residual of Equation (2-66) to solve the linear system. However, the latter case requires the well residual equation in the residual vector.

That are two types of variables that comprise the vector $\boldsymbol{\delta X}$: reservoir variables and well variables. Reservoir variables are pressures and saturations, while well variables are wellbore pressure and flow rates.

PUC-Rio - Certificação Digital Nº 1912763/CA

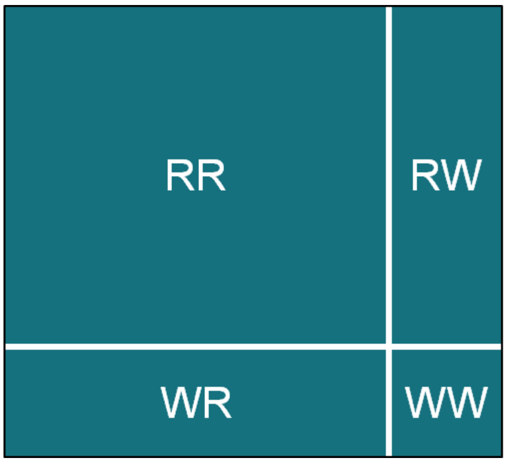

Figure 2.7: Submatrices in Jacobian matrix, adapted from [68].

For cells that are not perforated by wells, the primary unknowns are P_o, if we use the IMPES formulation, or otherwise, they are P_o, S_w, and P_b or S_o. Since cells that contain wells have an implicit treatment, they have one additional unknown if the well is rate specified: P_{wfref}. Finally, well equations have the same set of primary variables as the cells in which they are completed.

The general structure of the Jacobian matrix is shown in Figure 2.7, where we have four submatrices:

– **RR**: matrix of the derivatives of reservoir equations with respect to reservoir variables;

– **RW**: matrix of the derivatives of reservoir equations with respect to well variables;

– **WR**: matrix of the derivatives of well equations with respect to reservoir variables;

– **WW**: matrix of the derivatives of well equations with respect to well variables.

Just as the residual and δX vectors, the structure of the matrix changes depending on well specification and the formulation of the cells at a given timestep. The operating conditions dictate if **RW**, **WR**, and **WW** are required to couple well and flow equations. The formulation of the cells, on the other hand, determines the structure of the **RR** submatrix. For a more detailed description of the structure of this particular submatrix, the reader should refer to [69].

After solving the linear system from Equation (2-67), the primary variables are updated for FIM, IMPES, and well equations. Afterward, IMPES cells must compute the saturations and bubble-point pressures explicitly using

PUC-Rio - Certificação Digital Nº 1912763/CA

the procedure described in [13]. Once all variables are calculated, the Newton iteration continues until convergence.

Implicit-level determination

The level of implicitness of reservoir cells varies in time, depending on stability conditions. Thomas and Thurnau proposed a simple criterion for changing the formulation [18]. It was based on a specified saturation or pressure change threshold from a previous iteration. Given that this criterion performs the switching only from IMPES to FIM, it has limited practical use.

With the development of Courant–Friedrichs–Lewy (CFL) conditions for three-phase flow [70, 71], one can switch from IMPES to FIM and FIM to IMPES using the CFL value, which is defined as

$$CFL = \frac{1}{2}\frac{\Delta t}{V_b \phi}\left| f_{0,0} + f_{1,1} + \sqrt{(f_{0,0}+f_{1,1})^2 - 4(f_{0,0}f_{1,1} - f_{0,1}f_{1,0})}\right| \tag{2-70}$$

where the coefficients $f_{0,0}$, $f_{1,1}$, $f_{0,1}$, and $f_{1,0}$ are calculated according to [71]. In this work, we use $CFL \geq 1$ to switch from IMPES to FIM, and $CFL \leq 0.8$ to switch from FIM to IMPES [72].

PUC-Rio - Certificação Digital Nº 1912763/CA

2.2.7 Phase appearance and disappearance

In the black-oil model, the gas phase may appear or disappear under certain conditions. Proper handling of this phenomenon is important for the convergence of the nonlinear solver. In this work, we are using the variable-substitution method, where for the undersaturated state, the primary variables are P_o, S_w, and P_b, while for the saturated state, the primary variables are P_o, S_w, and S_o. The main idea of this formulation is to identify what triggers the phase transition and then determine the appropriate formulation for each cell [13, 19].

Figure 2.8 illustrates the algorithm to treat this phenomenon numerically. If the cell is undersaturated, it stays undersaturated while the oil pressure is greater than the bubble-point pressure. If the oil pressure becomes lower than the bubble-point pressure after Newton's iteration, the state transition is triggered, and the cell becomes saturated. The saturated state is maintained while the gas saturation is higher than zero. If this saturation becomes negative, the gas component is composed only by the solution gas in the oil phase, triggering the state transition to an undersaturated state.

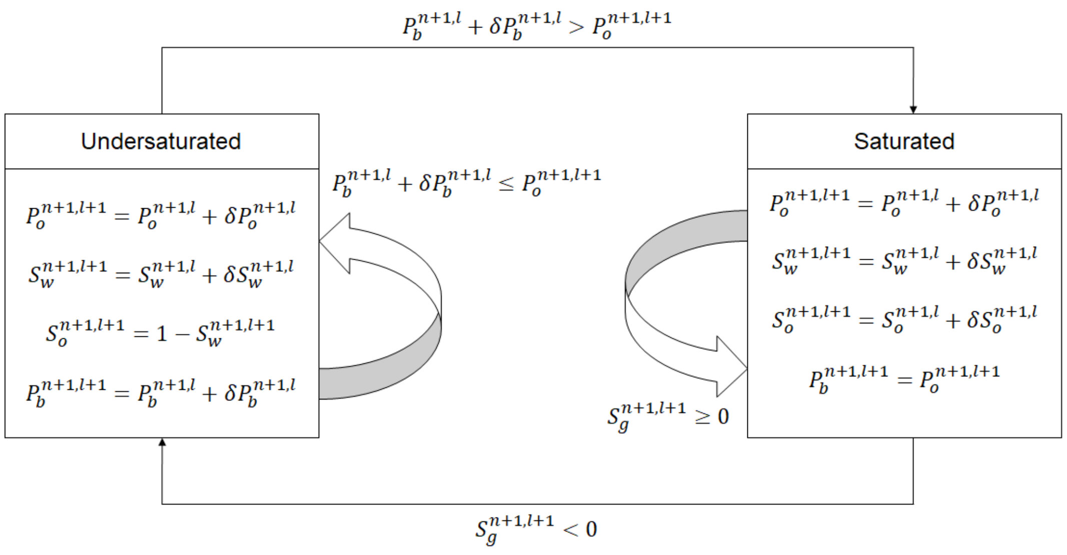
$P_b^{n+1,l} + \delta P_b^{n+1,l} > P_o^{n+1,l+1}$
Undersaturated
$P_o^{n+1,l+1} = P_o^{n+1,l} + \delta P_o^{n+1,l}$
$S_w^{n+1,l+1} = S_w^{n+1,l} + \delta S_w^{n+1,l}$
$S_o^{n+1,l+1} = 1 - S_w^{n+1,l+1}$
$P_b^{n+1,l+1} = P_b^{n+1,l} + \delta P_b^{n+1,l}$
$P_b^{n+1,l} + \delta P_b^{n+1,l} \leq P_o^{n+1,l+1}$
Saturated
$P_o^{n+1,l+1} = P_o^{n+1,l} + \delta P_o^{n+1,l}$
$S_w^{n+1,l+1} = S_w^{n+1,l} + \delta S_w^{n+1,l}$
$S_o^{n+1,l+1} = S_o^{n+1,l} + \delta S_o^{n+1,l}$
$P_b^{n+1,l+1} = P_o^{n+1,l+1}$
$S_g^{n+1,l+1} \geq 0$
$S_g^{n+1,l+1} < 0$

Figure 2.8: Numerical treatment for phase appearance and disappearance, adapted from [13].

2.2.8 Nonlinear methods for the black-oil model

In the context of scientific computing, Newton method is one of the most popular for solving nonlinear equations [73]. Although it has a local quadratic convergence, its computational cost is expensive, in particular for very large problems, which is the case of reservoir simulation [74]. For each timestep, it is necessary to perform Newton iterative process, illustrated in Algorithm 2.1, with ν representing the Newton iteration number.

Algorithm 2.1: Newton method

```
Given an initial guess $X^0$.
for $\nu = 0, 1, 2, ...$ until $X^\nu$ convergence do
    Assemble the residual vector R and Jacobian matrix J.
    Solve the linear system given in Equation (2-67) and obtain the
      increment $\delta X$.
    Let $X^\nu = X^{\nu-1} + \delta X$.
end
```

To reduce the computational cost, Dembo et al. [76] observed that solving the linear system of Equation (2-67) with a direct or an iterative linear solver with a very tight tolerance may result in *oversolving*. Figure 2.9 shows this phenomenon, where several linear iterations are performed in order to reach the desired linear tolerance, but its solution demands several Newton iterations to converge. They addressed this problem by proposing a generalization of the

PUC-Rio - Certificação Digital Nº 1912763/CA

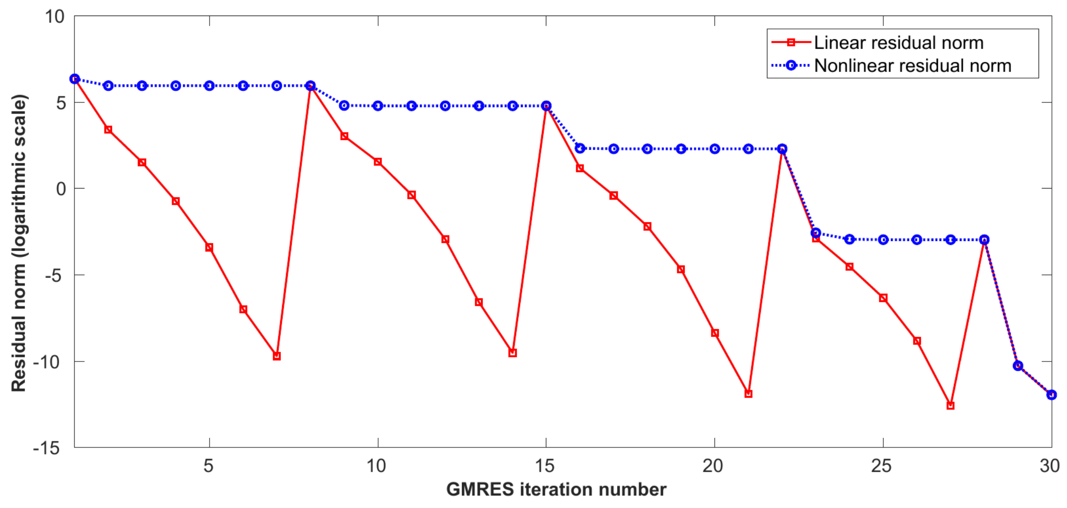

Figure 2.9: Example of an *oversolving* problem [75].

Newton method, called Inexact Newton. This method is described in Algorithm 2.2.

In each iteration of the Inexact Newton algorithm, a forcing term η^ν is chosen to control the accuracy to which Newton equation needs to be solved. In particular, with we choose $\eta^\nu = 0$, we obtain the standard Newton method. This method has been used in various scientific computing applications, including reservoir simulators [74].

The choice of the forcing term is what dictates the performance of the Inexact Newton algorithm, because not only it determines its speed of convergence, but it also affects its robustness and accuracy [73]. The most adopted choices for this term were proposed by Eisentat and Walker [77],

Algorithm 2.2: Inexact Newton method

```
Given an initial guess $X^0$.
**for** $\nu = 0, 1, 2, ...$ *until* $X^\nu$ *convergence* **do**
    Assemble the residual vector R and Jacobian matrix J.
    Choose **some** $\eta^\nu \in [0, 1)$.
    Inexactly solve the Newton Equation (2-67) and obtain the
      increment $\delta X$, such that

          $$\|R(X^\nu) + J(X^\nu)\delta X\| \leq \eta^\nu \|R(X^\nu)\|. \quad (2\text{-}71)$$

    Let $X^\nu = X^{\nu-1} + \delta X$.
**end**
```

PUC-Rio - Certificação Digital Nº 1912763/CA

which are listed as follows:

$$\eta_\nu = \begin{cases} \frac{\|R(X^\nu) - R(X^{\nu-1})J(X^{\nu-1})\delta X^{\nu-1}\|}{\|R(X^{\nu-1})\|} \\ \\ \frac{\|R(X^\nu)\| - \|R(X^{\nu-1})J(X^{\nu-1})\delta X^{\nu-1}\|}{\|R(X^{\nu-1})\|} \\ \\ \Theta\left(\frac{\|R(X^\nu)\|}{\|R(X^{\nu-1})\|}\right)^\psi \end{cases} \qquad \nu = 1, 2, ..., \tag{2-72}$$

where $\psi \in (1, 2]$ and $\Theta \in [0, 1)$. The third option is employed in our simulator, based on [74, 75, 78]. This work follows the Inexact Newton proposed by Sheth and Moncorgé [75], where $\psi = \frac{\sqrt{5}+1}{2}$ and Θ as monotonically decaying function of the Newton iteration count ν:

$$\Theta(\nu) = \max\left(\xi_0, \Theta_0 \exp\left(1 - \nu\right)\right) \tag{2-73}$$

with $\xi_0 = 1.0 \times 10^{-6}$ and $\Theta_0 = 0.5$.

PUC-Rio - Certificação Digital Nº 1912763/CA

2.2.9 Adaptive timestep control

Choosing a suitable timestep is a critical task for carrying out large-scale reservoir simulations, as using very small timesteps may lead to prohibitive runtimes. Alternatively, using large timesteps can increase the error in the solution and, even using unconditionally stable methods (i.e., FIM), there is no guarantee that Newton's method will converge [1]. Many techniques address this problem by automatically adjusting the timestep size [79–82].

Among these strategies, Todd et al.'s [79] procedure is the most commonly used in reservoir simulation [13, 19]. Their approach uses the changes of the primary unknowns to limit the growth and reduction of successive timesteps. Todd et al.'s heuristic is given by

$$\Delta t^{n+1} = \Delta t^n \min\left(\zeta_{\Delta_t}, \zeta_w, \zeta_o, \zeta_g\right) \tag{2-74}$$

where

$$\zeta_w = \frac{\Delta S_{w desired}}{\max(|{S_w}^{n+1} - {S_w}^n|)_{i,j,k}} \tag{2-75}$$

$$\zeta_o = \frac{\Delta P_{o desired}}{\max(|{P_o}^{n+1} - {P_o}^n|)_{i,j,k}} \tag{2-76}$$

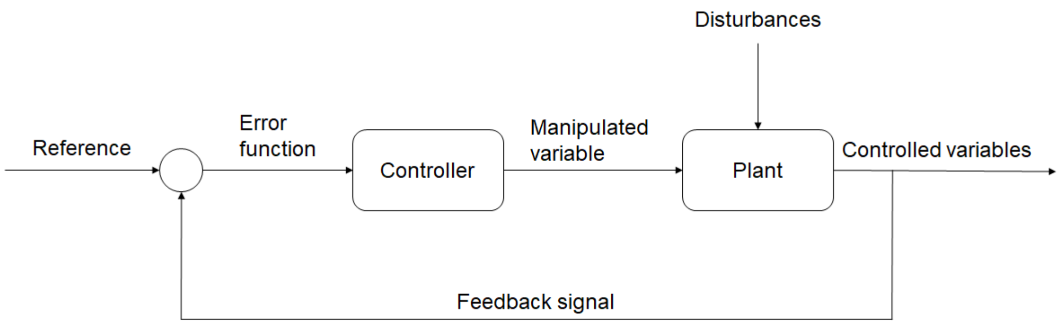

Figure 2.10: Block diagram of feedback control loop, adapted from [82].

$$\zeta_g = \frac{\Delta P_{bdesired}}{\max(|{P_b}^{n+1} - {P_b}^{n}|)_{i,j,k}} \quad \text{or} \quad \frac{\Delta S_{odesired}}{\max(|{S_o}^{n+1} - {S_o}^{n}|)_{i,j,k}}. \tag{2-77}$$

The numerators of Equations (2-75)-(2-77) are the desirable changes of the primary unknowns and the denominators are the maximum changes of these unknowns. Additionally, $\zeta_{\Delta t}$ is a user-defined maximum allowable ratio of $\Delta t^{n+1}/\Delta t^{n}$.

Since step size selection is a control problem, we can design a timestep control strategy based on linear feedback control theory. Söderlind [83] pointed out several advantages of using this approach: the algorithms can be analyzed in terms of stability and adaptivity, and they can be designed to produce smoother step-size sequences resulting in significantly improved regularity and numerical stability. Figure 2.10 illustrates the feedback control system, where the reference input is the value at which the user desires to maintain the controlled variable, the controller is the algorithm utilized to adjust the manipulated variable, the manipulated variable is the variable to adjust, and the plant is the engineering process that outputs the controlled variable. The main idea of the control system process is to build an error function from reference and feedback signals to adjust the manipulated variable to ensure that the control variable is kept below the tolerance [82].

In the case of reservoir simulation, the feedback control loop is represented in Figure 2.11. The reference values are the desired changes in the primary variables, the manipulated variable is the timestep, the plant is the reservoir simulation, the controlled variables are the primary unknowns, the feedback signal is the maximum change of those variables across the grid, and the disturbances include changes of boundary conditions and state transitions.

There are various controllers in the literature for this specific problem [82, 83]. In our study, we consider two controllers: an integral controller and a PID controller. The integral controller is a subset of the PID controller

PUC-Rio - Certificação Digital Nº 1912763/CA

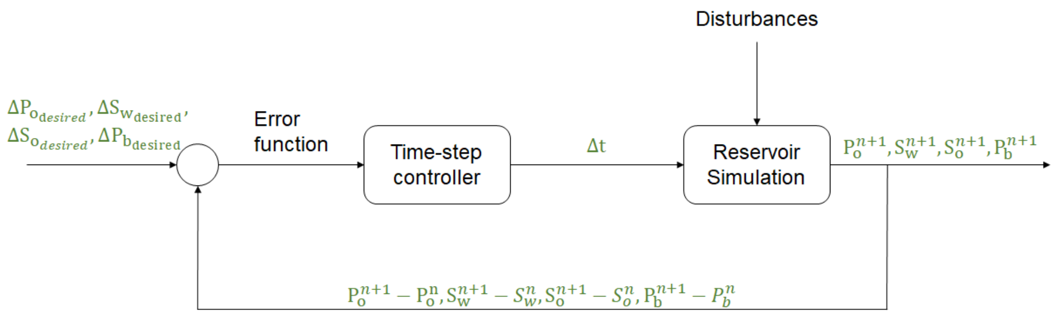
Disturbances
$\Delta P_{o_{desired}}, \Delta S_{w_{desired}}, \Delta S_{o_{desired}}, \Delta P_{b_{desired}}$
Error function
Time-step controller
Δt
Reservoir Simulation
$P_o^{n+1}, S_w^{n+1}, S_o^{n+1}, P_b^{n+1}$
$P_o^{n+1} - P_o^n, S_w^{n+1} - S_w^n, S_o^{n+1} - S_o^n, P_b^{n+1} - P_b^n$

Figure 2.11: Adaptive timestepping block diagram, adapted from [82].

and will thus be explained next. The PID controller adjusts the timestep changes in the estimated errors of the last three timesteps through three controlling terms: the proportional, integral, and derivative. The proportional part controls the sudden changes of the system, the integral part controls the relationship between the error in the present and past time, and the derivative anticipates the output values of the plant to correct them, preventively. These two controllers can be written as

$$\Delta t^{n+1} = \Delta t^n \left(\frac{r^{n-1}}{r^n}\right)^{k_P} \left(\frac{1}{r^n}\right)^{k_I} \left[\frac{(r^{n-1})^2}{r^n r^{n-2}}\right]^{k_D} \tag{2-78}$$

where k_P, k_I, and k_D are the proportional, integral, and derivative gains, respectively, and r^n is defined as

$$r^n = \max\left(\zeta_o^{-1}, \zeta_w^{-1}, \zeta_g^{-1}\right). \tag{2-79}$$

Additionally, $\zeta_{\Delta t}$ is used as a low-cut filter for the timestep resulting from Equation (2-78). That is

$$\Delta t^{n+1} = \min(\zeta_{\Delta t}\Delta t^n, \Delta t^{n+1}). \tag{2-80}$$

The integral controller is obtained using $k_P = 0$ and $k_D = 0$, with $k_I \neq 0$. Note also that Todd et al.'s procedure is an integral controller with $k_I = 1$.

PUC-Rio - Certificação Digital Nº 1912763/CA

3
Basic aspects of the simulator

Understanding and modeling the various phenomena in a particular hydrocarbon reservoir is crucial for selecting the most suitable recovery strategy. Since each reservoir requires a different approach, oil companies developed general-purpose simulators instead of developing one for each recovery process. In this class of simulators, the solution for any recovery process comes from a generic model, where only the necessary equations to obtain the results are used [84]. For instance, this type of simulator must be able to dynamically change the number of FIM and IMPES equations for AIM formulation, keeping track of the number of phases, components, and well equations, while assembling the Jacobian accordingly.

The design of a multipurpose simulator demands flexibility to incorporate all the physics and options to give the reservoir engineer as much information about the reservoir as possible. Simultaneously, such simulator must be developed aiming for maintainability, reusability, and scalability.

For this purpose, Topsim framework [51] was chosen as the central piece of our application. It provides tools to connect several specialized algorithms and services in the form of plugins, enabling the user to simulate different physical problems with varying levels of complexity. Also, it has proven to perform well for large-scale numerical analysis, which is one of the future goals of this research.

This chapter describes the development of our simulator using Topsim. We present all information about the implementation of the models described in Chapter 2 and present all services available in GSim.

PUC-Rio - Certificação Digital Nº 1912763/CA

3.1
Framework and model representation

The Topsim framework is a tool for the development of scientific numerical simulations [51]. Its design philosophy is based on plugins connected through general APIs, where the user has complete control to select the desired layout to run simulations with different ranges of complexity. Using the framework, the user may create, load, and link plugins accordingly for a specialized task that compose a bigger system, in many cases, without the necessity of an in-depth understanding of what every plugin of the layout does. Instead, the user may load the required plugin dynamically, making the application compact and with small memory usage.

The framework has a straightforward infrastructure based on a small kernel, which contains two components: the TopS [85] data structure and a Plugin Manager.

TopS is a compact adjacency-based topological data structure for mesh representation. It supports both two- and three-dimensional meshes with any type of cells (in this library, named elements). The data structure is capable of retrieving all topological adjacency relationships with a low memory footprint. This is achieved by explicitly representing only the *elements* and *nodes* of the *model* (mesh representation). The elements have references to all their nodes, and the nodes have references to all elements connected to them.

TopS also supports the creation and association of attributes to the nodes, the elements, and the model. These allow the developer to associate attributes related to the simulation, such as wells, completions, physical quantities, and results in the entities that comprise the mesh. Due to these functionalities, TopS is responsible for managing the communication of data between plugins since it provides APIs to create and manipulate attributes associated with the *model*.

The Plugin Manager is responsible for loading, linking, and registering plugins at runtime in the host system. It uses a Lua [86] configuration script as the communication protocol between the user and the application. This script is where the user describes the layout of a simulation, i.e., the information about which plugins will be loaded and how they should be connected with each other.

Plugins are the core of Topsim's applications. They are entities with all the features required to perform a service for the numerical analysis without modifying the kernel or other plugins. The interaction between plugins is given by an interface. Interfaces are software elements which formally define all services offered by a class of plugins.

Figure 3.1 illustrates an example of how plugins are combined to provide a linear system service. The service provides means of solving the linear system of equations by connecting the Orthomin, CPR, and Compressed Sparse Row (CSR) plugins through the *LinearSolver*, *Preconditioner*, and *SparseMatrix* interfaces. For the whole service to work and solve the request, the plugins exchange information, such as the RHS, Bo, and Bg using TopS. This is done by querying and saving data through TopS attributes.

PUC-Rio - Certificação Digital Nº 1912763/CA

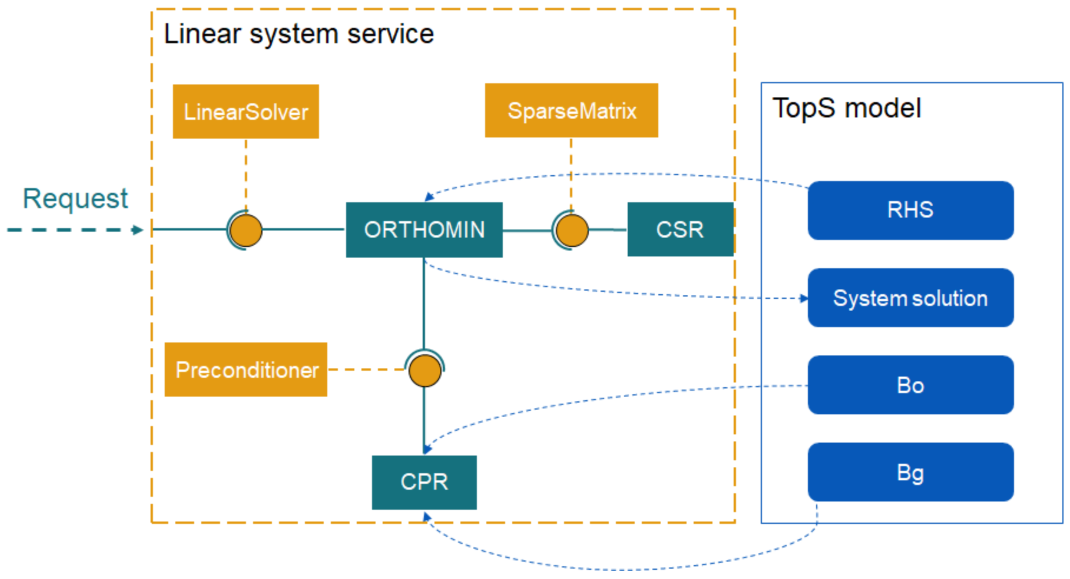

Figure 3.1: Example of linear system service, where CPR is the Constrained Pressure Residuals preconditioner, CSR is the Compressed Sparse Row matrix storage format, and RHS is the vector b from a $Ax = b$ linear system of equations.

3.2 Reservoir simulator architecture

Figure 3.2 illustrates a combination of plugins to perform a black-oil simulation. In this section, we provide a detailed description of all interfaces and plugins implemented in this work. For the sake of simplicity, we will divide the explanation of these services into five subsystems:

- Nonlinear solving (Figure 3.3)
- Input/Output (Figure 3.4)
- Geometry (Figure 3.5)
- Numerical solution (Figure 3.6)
- Linear solving (Figure 3.7)

PUC-Rio - Certificação Digital Nº 1912763/CA

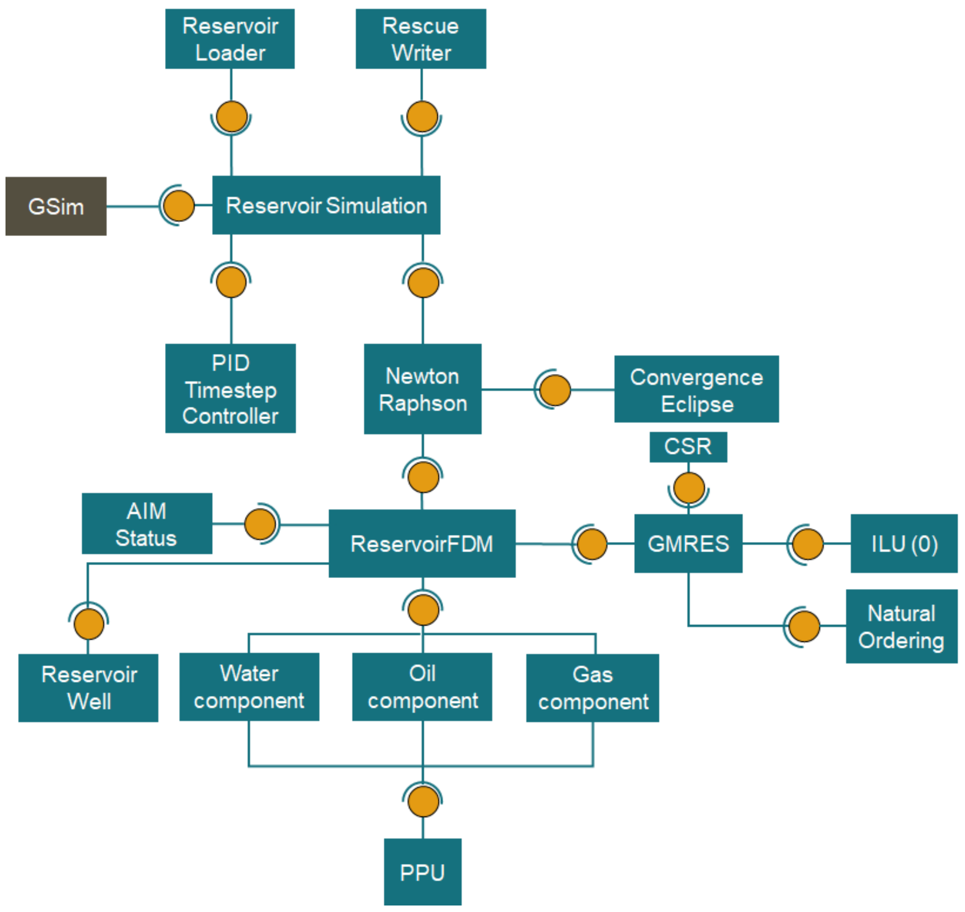

Figure 3.2: Example of a configuration to solve a reservoir simulation problem.

3.2.1 Nonlinear solving subsystem

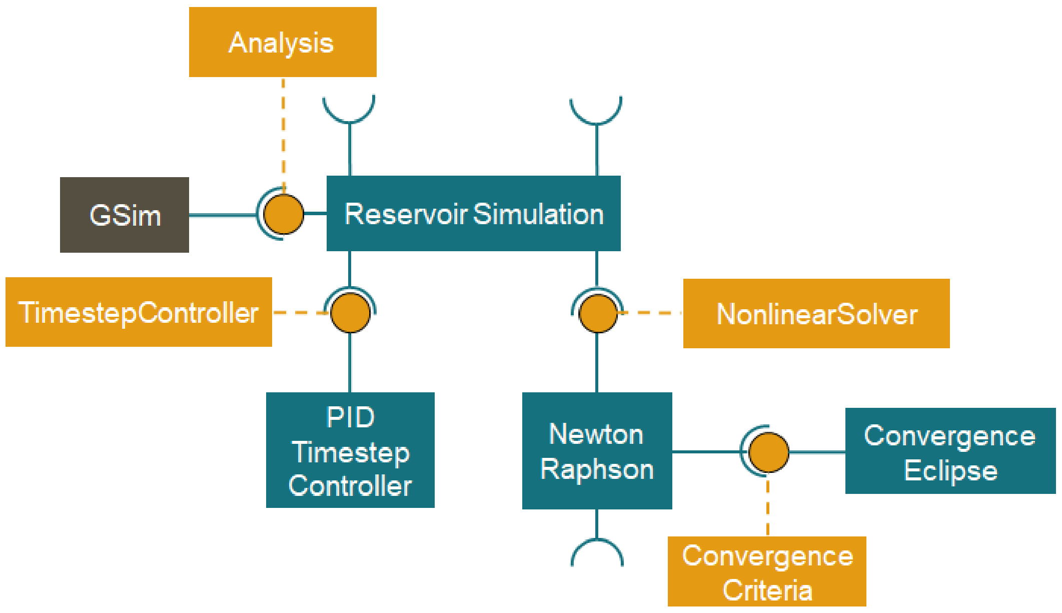

Figure 3.3: Example of a configuration for the nonlinear solving subsystem

PUC-Rio - Certificação Digital Nº 1912763/CA

A reservoir simulation may be divided into three layers: time marching, iterative nonlinear process, and linear solution of the linearized set of equations. For each timestep several nonlinear iterations are performed until convergence to obtain the behavior of the system. Inside each nonlinear iteration, the model equations are linearized to build a matrix-vector system, solved by a direct or iterative linear solver. This section describes the design of the first two layers of the simulation.

Analysis

The simulation starts from the plugin responsible for running the analysis in the time domain, named *ReservoirSimulation*. It initializes by asking a *Loader* plugin to populate TopS. Once the mesh is generated and all attributes are successfully loaded, the plugin starts the time marching loop.

The first part of the loop consists of a setup of the current timestep. In this process, a request is sent for a *WellManager* to verify if there are scheduled well events to be handled, for a *TimestepController* to compute the timestep size, and a *NumericalMethod* to set up the equations for the current timestep. The details about this setup process inside each plugin will be given further in this chapter.

After that, the plugin asks for a *NonlinearSolver* to obtain the solution of the current timestep. With the convergence of the nonlinear solver, the *NumericalMethod* updates the primary variables and checks if there is any violation of AIM'S stability criteria, operating conditions of the wells, or maximum allowed variation of the primary variables.

The coordination of these checks is made through other plugins. If any violation is triggered, there is an attribute available in TopS that warns the *ReservoirSimulation* plugin. Then, the *TimestepController* repeats the step and shrinks it depending on the type of violation. By repeating the timestep, the *NumericalMethod* resets the values of the primary variables of the current step to the ones from the previous step as a new initial guess for the *NonlinearSolver*.

Before moving to the next timestep, the *ReservoirSimulation* asks the Writer to write the results into the output file.

TimestepController

As previously mentioned, the *TimestepController* is responsible for computing the timestep size. We implemented two plugins for this task, called *Basic* and *PID*.

PUC-Rio - Certificação Digital Nº 1912763/CA

Their algorithm gets three flags from TopS, one for each of the violations explained in the previous section. If none of these flags is activated, the controller computes the step size using Equation (2-74) for the *Basic* controller and Equation (2-78) for the *PID*.

In the case of violation of AIM's stability criteria, we decided to raise the level of implicitness of the reservoir cells instead of shrinking the step size. This choice allows the simulation to keep taking larger steps. In this case, the timestep is repeated, but its value remains the same.

In the case of violation of well operating conditions, the step is repeated. Then, the value of the timestep is set to either a user-defined value or one day. This is important for maintaining the convergence of the nonlinear solution because changes of the boundary conditions lead to numerical instabilities.

For maximum allowable changes in the primary variables violation, the step is repeated, and its timestep size is divided by the maximum allowable ratio ($\zeta_{\Delta t}$).

After calculating the step size, the result is stored in TopS so that other plugins can query it. The implementation of all those conditions is straightforward in the *Basic* plugin. However, since the *PID*'s algorithm uses information about the previous two timesteps, a resetting strategy is required when repeating the timestep. Our strategy comprises of setting r^{n-2}, r^{n-1}, and r^{n} (from Eq. (2-79)) equal one.

PUC-Rio - Certificação Digital Nº 1912763/CA

NonlinearSolver and ConvergenceCriteria

The *NonlinearSolver* is responsible for finding the solution of the non-linear PDEs for a given timestep. We implemented the *NewtonRaphson* and *InexactNewton* plugins, given by Algorithms 2.1 and 2.2, respectively.

These solvers iterative procedure consist in requesting the *Numerical-Method* to ask the *FluidComponent*s and *WellManager* plugins to compute their properties, such as relative permeabilities, porosities, densities, FVFs, viscosities, and mobilities. Then, it calls the *NumericalMethod* to request these same plugins to compute their residual vectors.

Afterward, the nonlinear solver calls a *ConvergenceCriteria* plugin to verify whether the residual satisfies the tolerance. If the procedure did not converge, the *NumericalMethod* is asked to request the *FluidComponent*s and *WellManager* plugins to assemble the Jacobian matrix. With the residual and matrix, the linear system from Equation (2-67) is solved by asking the *NumericalMethod* to call a *LinearSolver* plugin. Finally, the *NonlinearSolver* tells *NumericalMethod* to update the primary variables with the increment obtained by solving the system.

The only difference between the *NewtonRaphson* and the *InexactNewton* plugins is that the latter computes the forcing term η^{ν} from Equations (2-72) and (2-73). For computing this value, we need to query the residual vector, the current iteration number, and the tolerance of the iterative linear solver from TopS.

By using an iterative algorithm, we must provide a suitable *Convergence-Criteria*. Two criteria are available in our simulator, inspired by the ones found in CMG's IMEX [56] and Eclipse [57] commercial simulators. The first criterion uses a maximum average residual, a maximum well residual, and a maximum residual. The second criterion uses a maximum material balance error, given by

$$\mathrm{MB}_{\alpha} = \overline{B}_{\alpha}\Delta t\left(\frac{\sum_n (R_{\alpha})_n}{\sum_n (\phi V_b)_n}\right) \qquad \alpha = w, o, g \tag{3-1}$$

and a maximum normalized saturation

$$\mathrm{CNV}_{\alpha} = \overline{B}_{\alpha}\Delta t \max\left|\frac{(R_{\alpha})_n}{(\phi V_b)_n}\right| \qquad \alpha = w, o, g \tag{3-2}$$

where n represents one cell and $\overline{B}_{\alpha}$ is the average value of the FVF of phase α in the reservoir.

The implementation of these criteria is simple, requiring the plugins to read some attributes to perform their calculations and maintain an attribute in TopS that tells the nonlinear solver if the iteration converged or not.

3.2.2 I/O subsystem

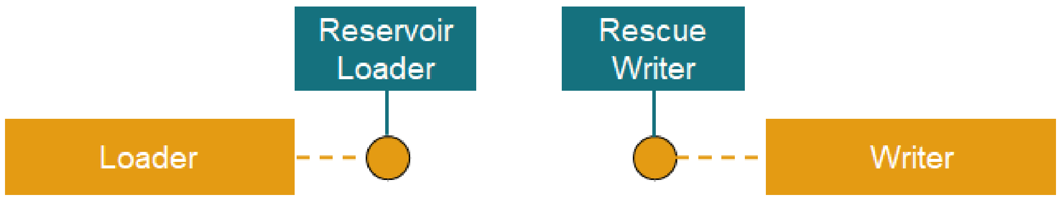

Figure 3.4: Example of a configuration for the I/O subsystem

As well as several simulators, GSim requires the communication between pre- and post-processing tools. Two services are used for these process, named writer and loader. Plugins developed for these purposes must be completely exchangeable to allow the integration of GSim with multiple file formats or

PUC-Rio - Certificação Digital Nº 1912763/CA

software. At the same time, they need to ensure compatibility with all other plugins.

The loader is responsible for reading an input file, creating the reservoir mesh, and populating the TopS model with reservoir attributes, e.g., PVT and SWT tables, reference pressures, number of cells, and user preferences for controlling the numerical methods. During this process, the loader converts all physical quantities from the input file to SI units. We implemented the *ReservoirLoader* plugin to read the same input file format as IMEX [57].

The writer is responsible for getting user-defined attributes of the model and writing them in an output file. The vast majority of this research results is visualized in GERESIM, which is an integrated environment for reservoir simulation management and reservoir geomechanics developed by our research group in cooperation with Petrobras. Therefore, the output file formats implemented in GSim were implemented aiming compatibility with GERESIM. Currently, two plugins are available: the *RESCUEWriter* and the *GRMWriter*. The former is a plugin that wraps the RESCUE standard library [87], while the latter writes the results in the GRM format, developed by our research group as a joint project with Petrobras.

PUC-Rio - Certificação Digital Nº 1912763/CA

3.2.3 Geometry subsystem

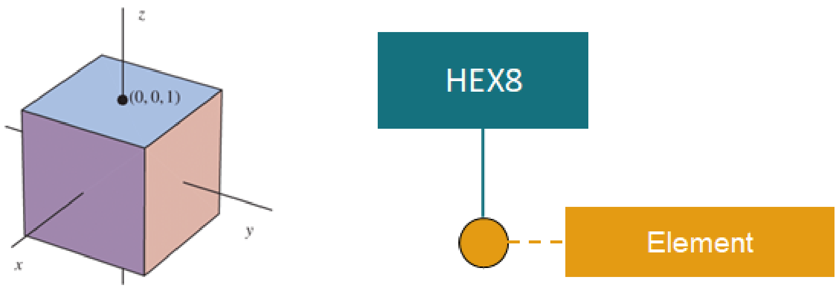

Figure 3.5: Example of a configuration for the geometry subsystem

Several geometrical quantities appear in reservoir simulation equations, such as bulk volumes, lengths and areas perpendicular to certain directions. GSim has a specific plugin for providing these information, named *HEX8*, which implements the classical 8-node hexahedron. This plugin gets the information about the mesh representation from TopS, computes the requested geometrical information and sends them to other plugins. This plugin works differently as the others shown in Figure 3.8, because it is not connected directly to any specific plugin. Instead, it can be accessed by any plugin through the Plugin Manager.

3.2.4
Numerical solution subsystem

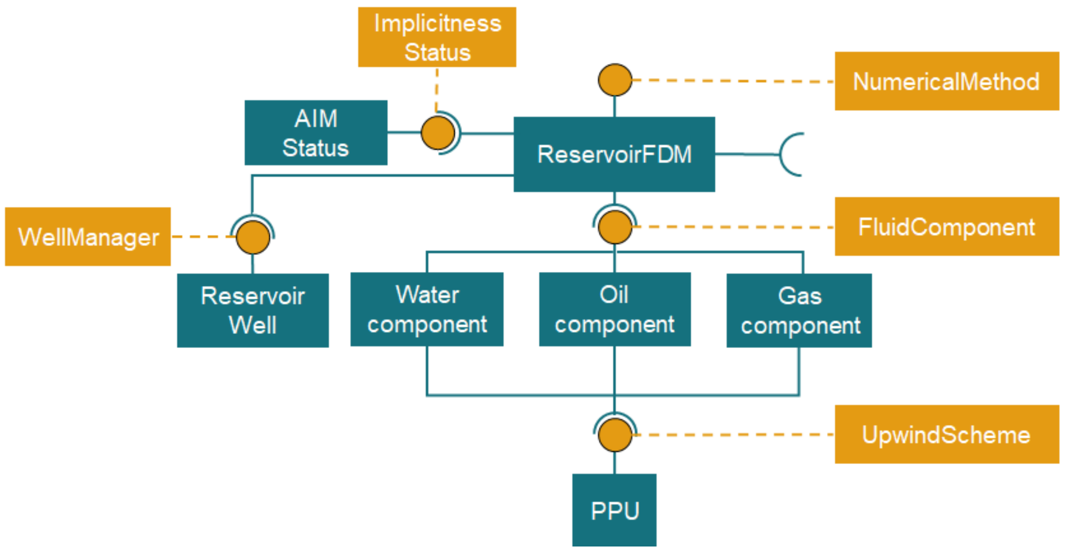

Figure 3.6: Example of a configuration for the numerical solution subsystem

PUC-Rio - Certificação Digital Nº 1912763/CA

NumericalMethod

Up to this point, none of the described subsystems enforced the use of any specific discretization method, like finite differences. We made this decision because by isolating the general simulation procedure from the model equation, it is possible to increase code reuse while maintaining flexibility.

The *NumericalMethod* is the service responsible for the communication between the general procedure plugins and plugins which implement the equations. In this work, this service is implemented using a plugin specialized in the finite differences method, named *ReservoirFDM*.

During the initialization of the simulation, it asks the *FluidComponent*s to compute pressures and saturations of the model using capillary/gravity equilibrium conditions, given by Equations (2-56)-(2-62). At this same stage, we compute and store the height difference between each cell and its neighbors, and the geometric factors of the transmissibilities (from Equation (2-42)) since they do not change during the simulation. The last stage of the initialization procedure involves allocating memory for the Jacobian matrix.

When programming a simulator, the developer does not know *a priori* the dimension of the problem that users want to solve. For this reason, we must allocate the Jacobian dynamically, as the number of rows and columns of the matrix depends on the number of cells, wells, and fluids, as well as the formulation of each cell (IMPES and FIM). Compared to static allocation, dynamic allocation is computationally expensive and should be used as little as possible [88]. Thus, we allocate the matrix considering the worst case for the problem coming from the user input file; that is, all cells are assumed to use FIM, and if the wells may operate with flow rate specification, we allocate **RW**, **WR**, and **WW** submatrices (Fig. 2.7). After the initialization, the *ReservoirFDM* receives requests from other plugins until the end of the analysis.

The first request is made in each timestep by the *ReservoirSimulation* plugin, consisting in a setup of the equations of the current step. The idea of this stage is to perform the computations that determine how to fill the Jacobian and the residual correctly by using two TopS attributes: one that computes the number of implicit equations, which are the equations that effectively goes into the linear system, and a list with information about the formulation of all cells of the mesh. Firstly, the *ReservoirFDM* tells the *ImplicitnessStatus* plugin to check which cells contain active completions and sets them as fully-implicit. Then, it asks the *WellManager* plugin to compute the number of equations required by the wells. Afterward, the *ReservoirFDM* sends a request to an *Ordering* plugin to determine the position of each equation in the residual vector and Jacobian matrix. Finally, it sets up a *SparseMatrix* plugin. The configuration of the *SparseMatrix* plugin consists in getting information about the number of implicit equations and the formulations of the cells to compute the number of non-zeros and fill the row/column pointer array of the user-defined compressed matrix format. We will give more details about this array at the linear solving subsystem section.

PUC-Rio - Certificação Digital Nº 1912763/CA

With the Jacobian and residual vector sizes setup, the nonlinear iteration procedure begins. For each nonlinear iteration, there is a setup inside the *NumericalMethod*, already mentioned at the nonlinear solving section. After that, it receives a request to ask the *FluidComponent*s and *WellManager* plugins to compute their residual vectors and their contribution to the matrix. For example, the *OilComponent* plugin computes the residual for the oil equations and the derivatives of the oil equations with respect to all primary variables. The *OilComponent* itself is responsible for filling the result of this computation in the residual vector and in the Jacobian matrix, according to the numbering of the equations. Similarly, the *WaterComponent*, *GasComponent*, and *ReservoirWell* plugins perform the same procedure with water, gas, and well equations, respectively. Lastly, the nonlinear solver asks the *ReservoirFDM* to request the *LinearSolver* to calculate the solution of Equation (2-67) and asks the *FluidComponent*s and *WellManager* to update the primary variables.

After the nonlinear iteration convergence is reached, it is time for the *ReservoirSimulation* to ask the *ReservoirFDM* to update the variables with respect to the current timestep. This is done by asking the *FluidComponent*s and *WellManager* to copy the values of the primary variables from the step $n+1$ to n. During this process, the maximum changes of the primary variables and well restrictions violations are checked inside each of the fluid and well plugins. After updating the variables, the *ImplicitnessStatus* is called by the *ReservoirFDM* to check if any cell needs to change its formulation.

ImplicitnessStatus

By using an adaptive-implicit formulation, we can simulate FIM and AIM without implementing them in separate plugins. By default, if there is no plugin to check the implicitness status of the cells, the simulation is carried out using FIM in all cells. Otherwise, a plugin named *AIMStatus* maintains a cell attribute in TopS to keep the information about the current formulation (i.e., IMPES or FIM) of each cell.

The *AIMStatus* plugin initializes non-perforated cells using the IMPES formulation and perforated cells using FIM. Nevertheless, there is a user-defined radius of the neighborhood attribute to initialize cells in the region near the wells with FIM formulation. This choice is given because numerical instabilities occur in regions with very high flow rates, which is usually the case for well regions. During the simulation, it checks the current status of each cell at each timestep to verify if it is necessary to change the formulation from IMPES to FIM or vice versa. These checks are performed by using the stability criteria of threshold [18] (described in Chapter 2) and CFL (Equation (2-70)).

PUC-Rio - Certificação Digital Nº 1912763/CA

WellManager

The *ReservoirWell* plugin is responsible for handling all tasks related to wells and its completions. In GSim, wells and completions are special data structures, containing several information about their behavior and operating conditions. These structures are kept in TopS as arrays of wells/completions structures.

The well data structure contains information about geometrical/physical properties, status, and specifications of a well. The geometrical/physical properties information comprises the average density ($\bar{\rho}_{well}$), skin factor (s), wellbore radius (r_w), and reference depth (Z_{wref}). The status of the well comprises in a name (for writing into the output file), an id, if the well is open or not, a well type (injector or producer), the number of completions, well equation number (if flow rate specified), and a list containing the ids of each completion of the well. Finally, the specifications comprises a number of constraints, injection fluid (if it is an injection well), BHP at the reference depth (P_{wref}), well specified flow rate (if flow rate specified), and a list containing the values of the operating conditions constraints.

The completion data structure holds the values of geometrical/physical properties and specifications of a completion. The geometrical/physical data comprises a difference of height between the completion and the reference depth of the well, the equivalent radius (r_{eq}), the well index (WI) of the cell containing the completion, the productivity/injectivity index J_{ω}, the wellbore pressure (P_{wf}), and the flow rate of each fluid (q_w, q_o, and q_g). The specification data contains information about its id, open/closed status, and the ids of the cell and well it belongs.

As other plugins, the *ReservoirWell* performs an initialization procedure. The initialization consists in looping the wells and completions structure arrays for computing the difference in height between the reference depth of the well and the cell containing the completion, the well index, and the equivalent radius using Equations (2-52) and (2-53). After finishing this process, this plugin is ready for answering the requests from other plugins.

The first request it receives comes from the *ReservoirSimulation* plugin, from the setup process of the current timestep. This stage checks if there is any user-defined scheduled changes in the operating conditions of the wells. If there is any change on the well operating conditions, this plugin warns the *TimestepController* plugin via a TopS attribute, and re-initializes the wells with the new operating conditions.

At each nonlinear iteration, there is a setup stage where three calculations are performed. Firstly, the *ReservoirWell* calculates the average density for each well data structure using Equation (2-55). Then, it asks each *FluidComponent* to compute the injectivity/productivity index to store in the completion data structure. Finally, it computes the flow rate and the completion BHP using Equations (2-49) and (2-54).

To compute the well contribution to the residual vector, the *ReservoirWell* gets the information about the well equation number from the well data structure. This information is assigned when an *Ordering* plugin is requested by the *NumericalMethod* to assign the location of each equation number of the current step. With the information about the specification and where the well residual will be placed in the residual vector attribute, we use Equation (2-66) to assign its value in this array (for flow rate specification, as explained in Chapter 2). Computing the coefficients of the Jacobian matrix is an anal-

PUC-Rio - Certificação Digital Nº 1912763/CA

ogous procedure. It comprises building the **RW**, **WR**, and **WW** submatrices by differentiating Equations (2-63), (2-64), (2-65), and (2-66) with respect to P_o, S_w, P_b or S_o, and P_{wref}. The last stage of the nonlinear iteration consists of updating the well primary variables (i.e., P_{wref}), which is straightforward.

The last request of the *ReservoirWell* comes from the *NumericalMethod* when the *ReservoirSimulation* requires the update of the variables to start a new timestep and check for violations. The well constraints violations are monitored in this part of the code. The algorithm consists in checking the restrictions from the most critical to less critical well constraints. If any constraint is violated, a flag is written in TopS for other plugins to take the necessary actions, and the most critical operating condition is assigned as the current one in the well data structure to continue the simulation.

PUC-Rio - Certificação Digital Nº 1912763/CA

FluidComponent and UpwindScheme

A *FluidComponent* is responsible for implementing the conservation equations of a fluid. This work implemented three plugins for this task, named *WaterComponent*, *OilComponent*, and *GasComponent*. Their initialization comprises a capillary/gravity equilibrium, as stated in the *Numerical-Method* section.

During the setup process requested by the *NumericalMethod*, they compute the variables dependent on the primary variables, that is, FVFs, viscosities, capillary pressures, and relative permeabilities. In addition, each component computes the mobilities of the transmissibility terms (Eq. (2-41)) by sending a request to an *UpwindScheme* plugin.

Two plugins are available in GSim to perform the *UpwindScheme* service, named *PPU* and *C1-PPU*. They get the phase potential from TopS and compute the mobility of the interface between each cell and its neighbors using Equation (2-46), where the switching functions Υ are given by Equations (2-47) and (2-48) for the PPU and C1-PPU plugins, respectively. After the setup stage, the *NumericalMethod* requests the *FluidComponent* to compute its residual vector.

Filling the right entries of residual vector depends on having the appropriate data structures. The implementation of this feature uses two arrays already described in the *NumericalMethod* section: an array which determines the position of each equation in the residual vector and an array that tells the current formulation of each cell. With these arrays available, we loop through all cells of the mesh. If the cell is FIM, there is an entry in the residual for each component. Thus, the water component fills its entry in the residual vector by using Equation (2-63), the oil component uses Equation (2-64), and the gas

PUC-Rio - Certificação Digital Nº 1912763/CA

component uses Equation (2-65). In the case where the cell is IMPES, there is only one entry per cell in the residual vector, given by Equation (2-69).

The Jacobian matrix uses a similar approach for filling its entries. For each cell of the mesh, it checks the formulation and the location of each coefficient using the same arrays. If the cell is IMPES, we compute the derivatives of Equation (2-69) with respect to P_o. Otherwise, we compute the derivatives of each of the Equations (2-63), (2-64), and (2-65) with respect to P_o, S_w, and S_o or P_b.

After the solution of the linear system, *FluidComponent*s are requested to update the primary variables for the new nonlinear iteration. Updating the oil pressure is trivial, since it is always a primary variable. However, the other properties must check if they should be updated using IMPES or FIM approach. For FIM updates, the update of water saturation is as trivial as the oil pressure, and the update of bubble-point pressure and oil saturation are performed using the algorithm of Figure 2.8. On the other hand, IMPES updates are calculated according to the procedure described in [13].

Finally, the last request for the *WaterComponent*, *OilComponent*, and *GasComponent* plugins comprises updating the variables for a new timestep. This is performed by copying each array of the primary variables of the converged step to arrays of the primary variables of the old timestep. As well as several simulators, the initial guess for the next Newton iteration consists in using the values of the primary unknowns from the last timestep.

3.2.5 Linear solving subsystem

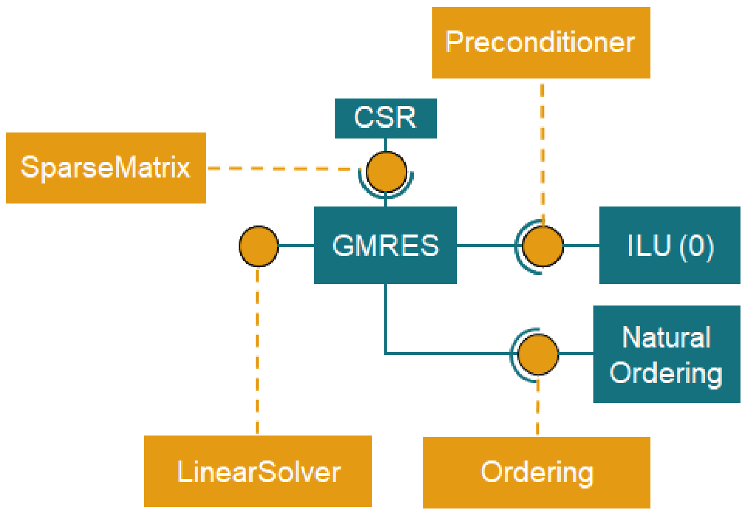

Figure 3.7: Example of a configuration for the linear solving subsystem

The matrices arising from reservoir simulation are, in most cases, very sparse. Taking advantage of this property is essential for reducing memory consumption and the number of operations required by the linear solver algorithm, especially in practical problems. We use different data structures to represent the matrices without storing the zero coefficients by means of a *SparseMatrix* service. Two plugins are available for performing this service, using two different formats: Compressed Sparse Row (CSR) and Compressed Sparse Column (CSC). The CSR format stores a $(m \times n)$ matrix using three arrays: Row Pointer, Column Index, and Values. These arrays are shown in the example of Equation (3-3),

$$\begin{aligned} [\boldsymbol{A}] &= \begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 3 \\ 4 & 5 & 6 \end{bmatrix} \\ \text{Values} &= [1,\ 2,\ 3,\ 4,\ 5,\ 6] \\ \text{Row Pointer} &= [0,\ 2,\ 3,\ 6] \\ \text{Column Index} &= [0,\ 2,\ 2,\ 0,\ 1,\ 2] \end{aligned} \tag{3-3}$$

where the Values array holds the values of the nonzero entries of the matrix from left to right and from top to bottom. The Row Pointer is of length $(m+1)$ and is responsible for holding the start of each row within the array of nonzero entries. The Column Index is of the same size as the Values array and holds the column index of each nonzero entry of $[\boldsymbol{A}]$.

The CSC format is similar to the CSR format, except that the values are read by the first column, a Row Index is stored for each value, and Column Pointers are stored. We will not explore other formats in this research, but it is worth mentioning that the Block Sparse Row [89] is another popular matrix compression scheme in the context of reservoir simulation. We developed our simulator thinking in cases like this, where experimenting a new method for a certain class of problems or comparing the performance of different methods requires little modifications from the developer using the framework.

The main bottleneck of the simulation resides in solving the large sparse linear system derived from multiphase flow. Since the most efficient linear solver is problem-dependent, providing multiple solvers is mandatory. For this purpose, four linear solvers are available in GSim: Pardiso, Biconjugate Gradient Stabilized (BICGSTAB), Generalized Minimal Residual (GMRES), and Orthogonal Minimization (ORTHOMIN).

Pardiso is an efficient parallel direct solver for large symmetric and unsymmetric matrices developed by Intel Corporation [90]. This solver was

PUC-Rio - Certificação Digital Nº 1912763/CA

wrapped by Duarte et al. as a plugin [51].

On the other hand, we use Piazza's [91] implementation of BICGSTAB, ORTHOMIN, and GMRES as iterative solvers. Along with these linear solvers, he also implemented preconditioning services in the context of reservoir simulation. The preconditioners available are Nested Factorization, Incomplete LU Factorization with no fill (ILU(0)), and CPR.

Finally, we implemented an ordering scheme service. This service determines the row and column where the coefficients of the Jacobian matrix are positioned. Likewise, it allocates the order of the equations in both residual and vector with the updated unknowns arrays of Equation (2-67). This allocation occurs by storing the position of the coefficients in TopS.

The ordering scheme plays an important role in the performance of the preconditioner and linear solver and, for this reason, it is important to provide a specialized algorithm. Currently, the algorithm available for this task is the *NaturalOrdering* plugin, which positions the equations following the lexicographical order of the cells.

In Figure 3.8, we show a summary of all interfaces and plugins available for our simulator. In the next Chapter, we will validate GSim and compare different plugin configurations to verify the extensibility and flexibility of our approach.

PUC-Rio - Certificação Digital Nº 1912763/CA

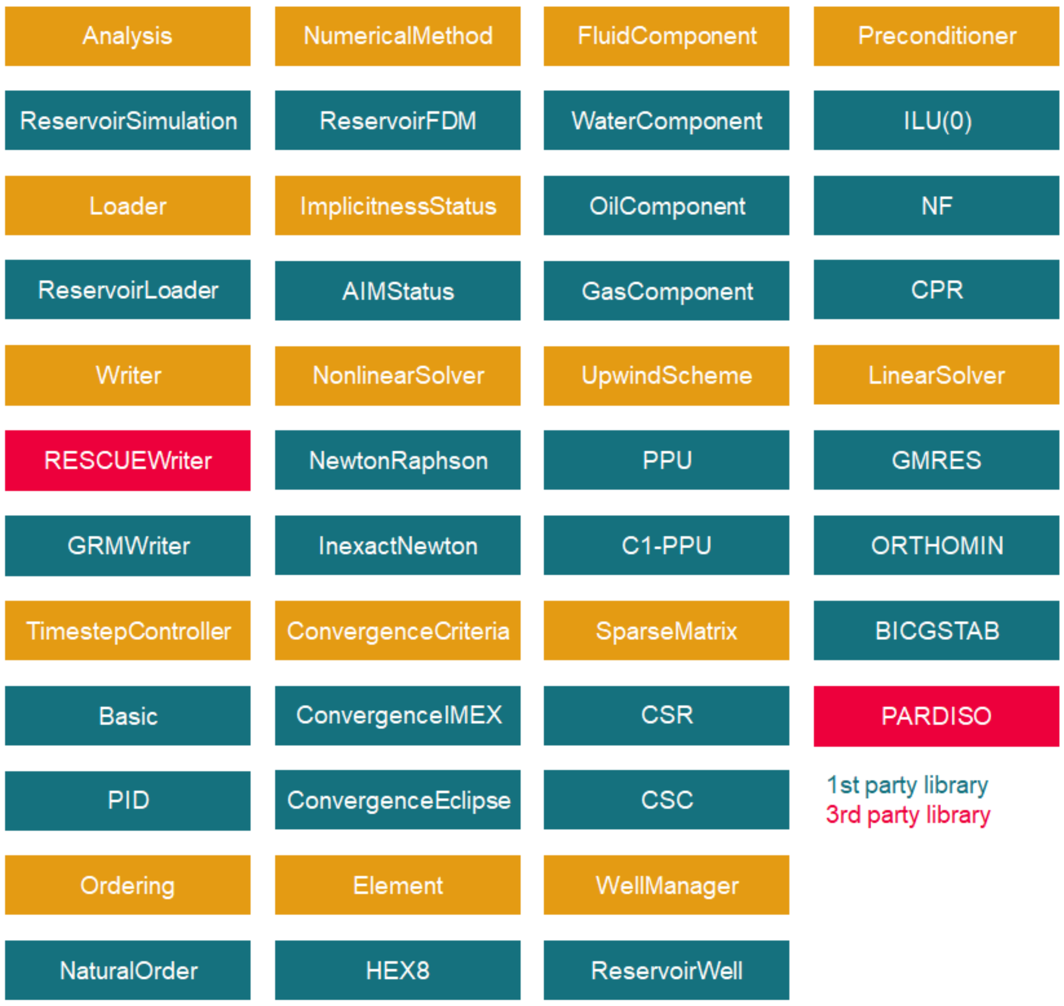

Figure 3.8: Summary of all interfaces and plugins available in GSim.

PUC-Rio - Certificação Digital Nº 1912763/CA

4
Results

This chapter performs several reservoir simulations to validate the mathematical models and simulator architecture described in Chapters 2 and 3. First, we give a brief description of each test case and which features it validates. The results are compared with CMG-IMEX commercial simulator using the same timesteps.

Finally, we use the test cases to compare the classical and state-of-the-art methods available in GSim to show how one can use this tool to test and develop different numerical procedures. All the examples presented in this Section were simulated using the machine specification listed in Table 4.1.

Table 4.1: Computing platform used in the simulations.

Operating system	Windows 10 Pro (64-bit)
RAM	16.0 GB
Compiler	Intel C++ 19.1
CPU	Intel Core i7-8700 3.20GHz

PUC-Rio - Certificação Digital Nº 1912763/CA

4.1 Five-spot

The first example is a water injection five-spot reservoir model with dimensions of size $8136 \times 8136 \times 492$ ft in the x, y, and z directions, respectively. The reservoir possesses a uniform initial porosity of 30% and is initially saturated with connate water and undersatured oil. The original reservoir pressure at the top depth is 2845 psia, while the initial bubble-point is 1422 psia. The characterization of the hydrocarbon phases is given in Table 4.2. The gas-oil and oil-water relative permeabilities are given in Figures 4.3 and 4.2, respectively.

Figure 4.1 shows the permeability map and well configuration, with one producer well in each corner of the reservoir and a injector at the center. All wells are controlled by their wellbore pressures, with the producers set to operate at a minimum 2560 psia and the injector is limited to 3129 psia. All

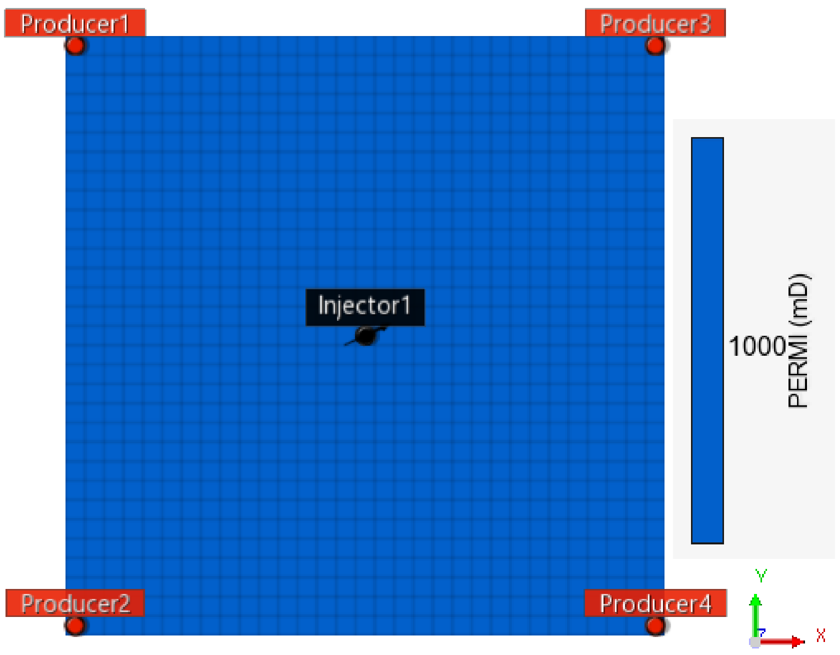

Figure 4.1: Permeability map for the five-spot case.

PUC-Rio - Certificação Digital Nº 1912763/CA

wells have only one completion, at the top layer of the reservoir. This model uses a $31 \times 31 \times 12$ mesh, with a total number of 11532 cells.

This problem was designed to validate the most basic features of the reservoir, such as the AIM formulation and the values of the primary variables. Figure 4.4 shows the symmetry of the pressure distribution, which is expected since the reservoir is homogeneous and there is a symmetric well configuration with all producers set to the same operating conditions.

As seen in Figures 4.5 to 4.9, all numerical results of our simulator matches the ones from CMG-IMEX. Figures 4.7 and 4.8 shows the cause-effect relationship between water occupying the pores of cell (16, 16, 1) and the decrease of the oil relative permeability. Figures 4.6 and 4.9 illustrates that, as pressure drops in the perforated cell, the oil flow rate decreases due to a lower pressure gradient between the wellbore and the perforated cell.

After the validation of the results, we verified the implicitness level of our simulation. Figures 4.10 and 4.11 show the formulation of each cell of the mesh, where the blue cells are calculated using FIM and red cells are calculated using IMPES. In addition, it is worth remembering that cells perforated by wells are always FIM, as discussed in Chapter 2. Since this is a simple problem, only a few cells near the injector changed to FIM over the simulation because of the rapid increase in pressure in the vicinity of the wellbore.

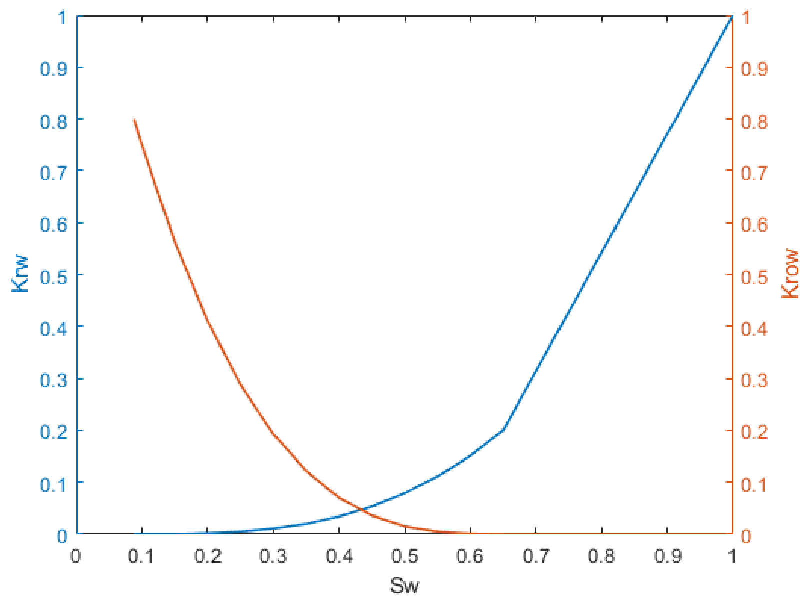

Figure 4.2: Oil-water relative permeabilities as functions of water saturation.

PUC-Rio - Certificação Digital Nº 1912763/CA

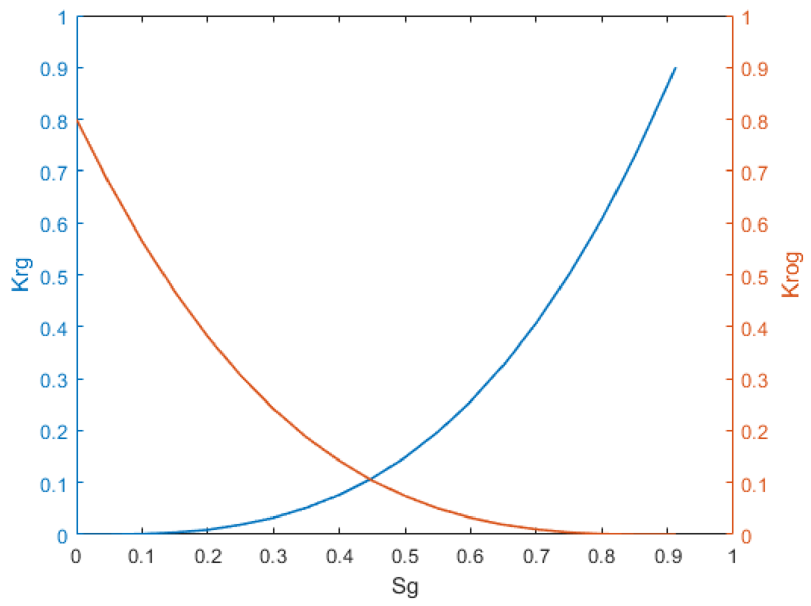

Figure 4.3: Gas-oil relative permeabilities as functions of gas saturation.

Table 4.2: PVT properties of oil and gas.

P_o [psia]	R_s [scf/STB]	B_o [RB/STB]	E_g [bbl/RB]	μ_o [cp]	μ_g [cp]
14.6500	0.0000	1.0344	0.8239	2.7695	0.01092
370.234	112.579	1.1013	21.7978	1.7277	0.01330
441.350	126.001	1.1073	26.1419	1.6713	0.01346
583.584	152.963	1.1193	34.9780	1.5707	0.01373
725.817	180.060	1.1311	43.9965	1.4838	0.01400
868.051	207.573	1.1430	53.1816	1.4074	0.01428
953.248	224.265	1.1502	58.7501	1.3658	0.01445
1137.868	260.471	1.1657	71.2296	1.3351	0.01484
1422.334	316.250	1.1896	90.8276	1.2611	0.01549
1706.801	372.029	1.2135	110.5960	1.2006	0.01619
2133.502	455.698	1.2494	139.8711	1.1266	0.01735
2560.202	539.367	1.2852	167.8947	1.0661	0.01863
2844.669	595.146	1.3091	185.5790	1.0311	0.01955
3555.836	734.594	1.3689	225.7614	0.9571	0.02209
4267.003	874.042	1.4287	260.2866	0.8966	0.02497
4978.170	1013.49	1.4884	289.8959	0.8454	0.02819

PUC-Rio - Certificação Digital Nº 1912763/CA

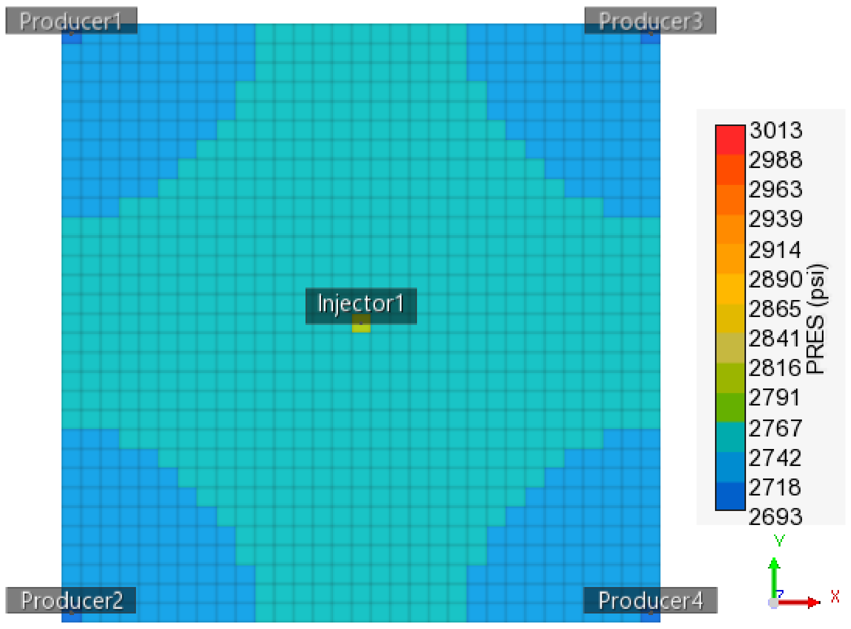

Figure 4.4: Pressure distribution on the top layer at t = 279 days.

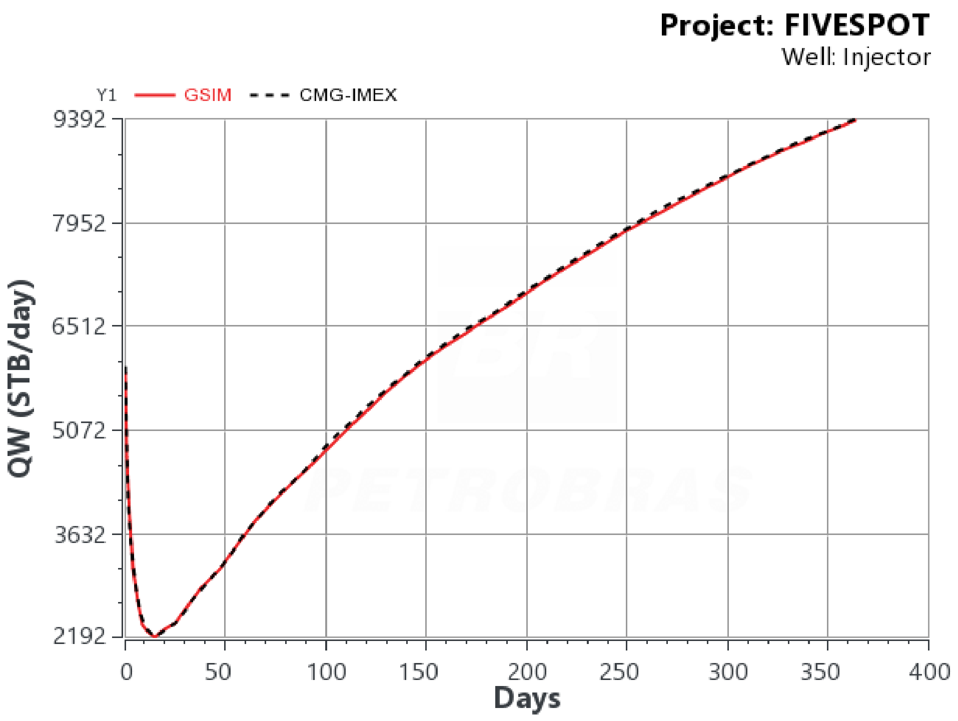

Figure 4.5: Water flow rate vs. time at Injector 1.

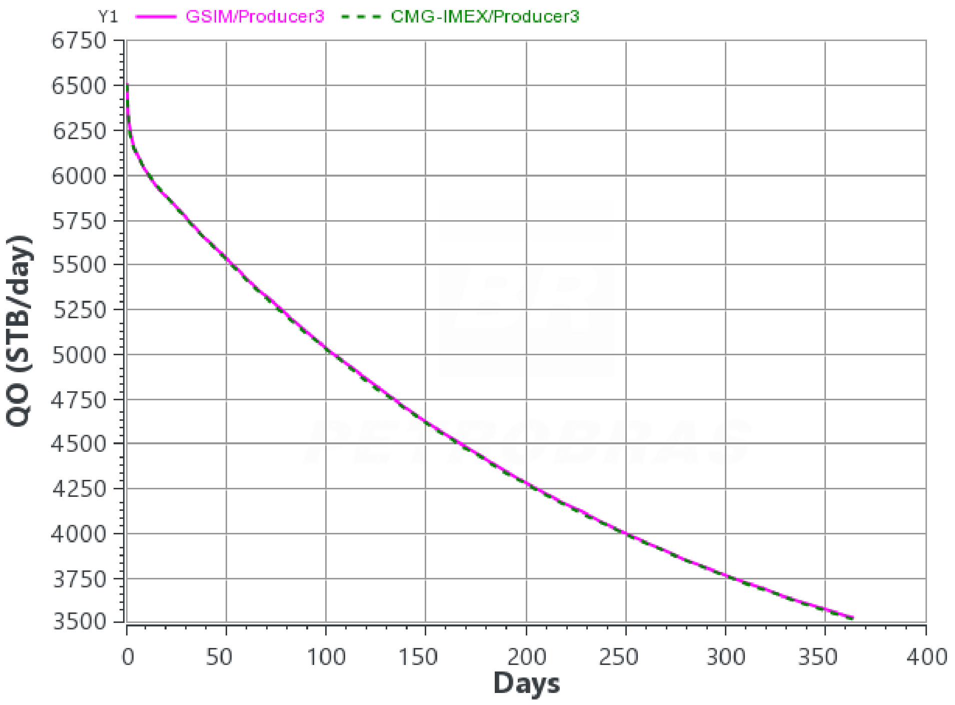

Figure 4.6: Oil flow rate vs. time at Producer 3.

PUC-Rio - Certificação Digital Nº 1912763/CA

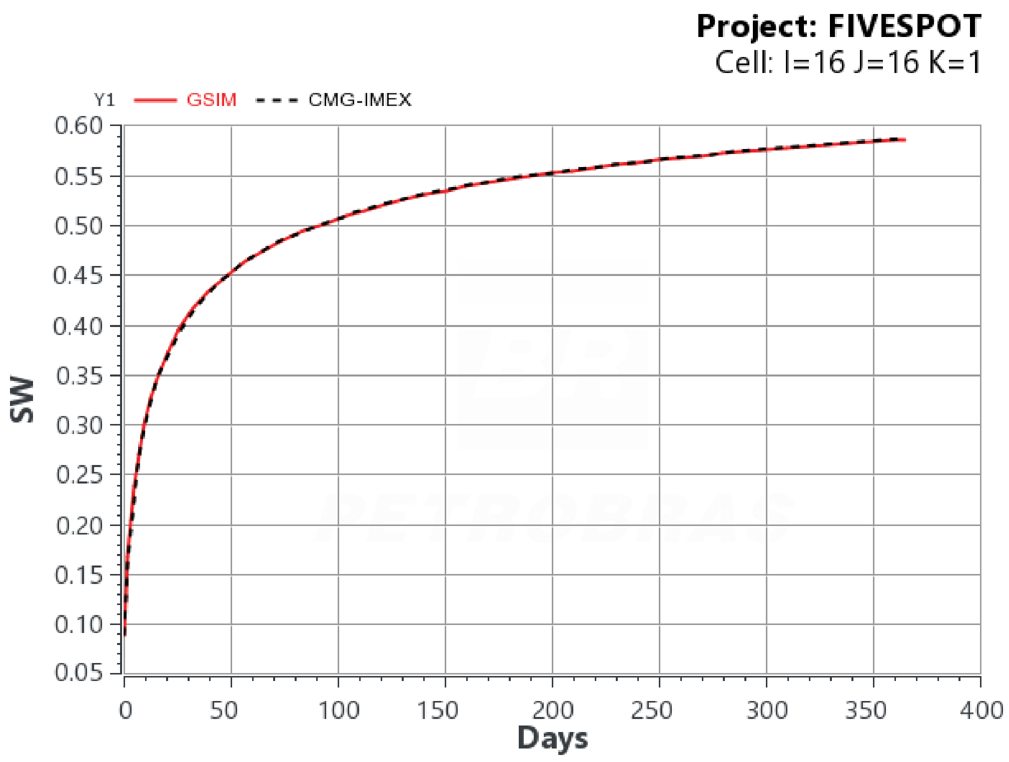

Figure 4.7: Water saturation vs. time at cell (16,16,1).

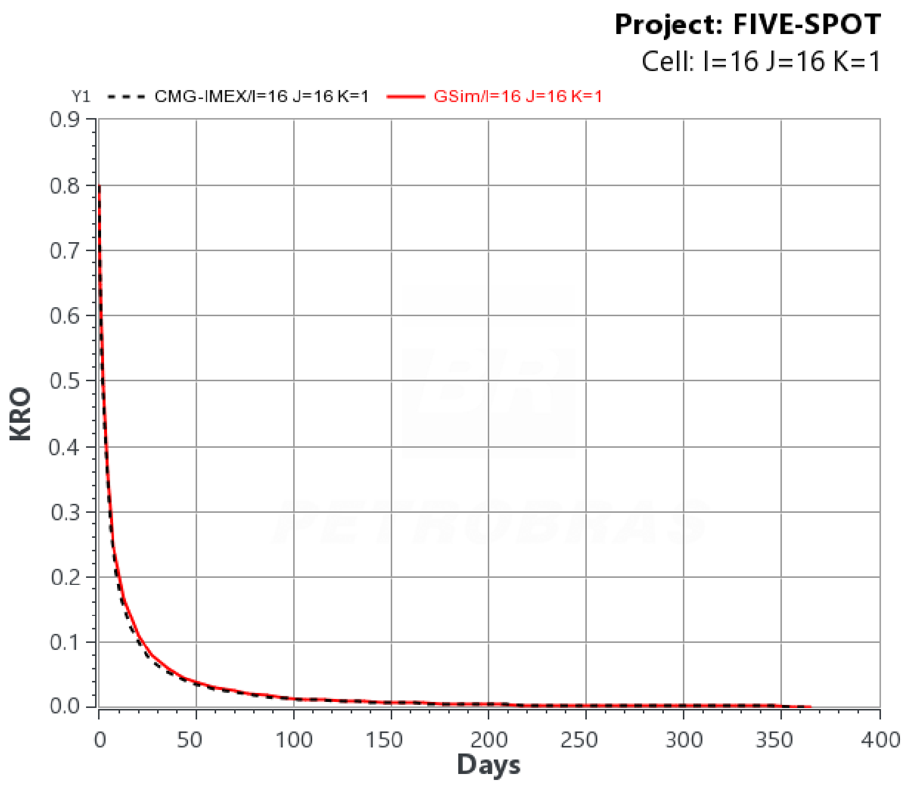

Figure 4.8: Oil relative permeability vs. time at cell (16,16,1).

PUC-Rio - Certificação Digital Nº 1912763/CA

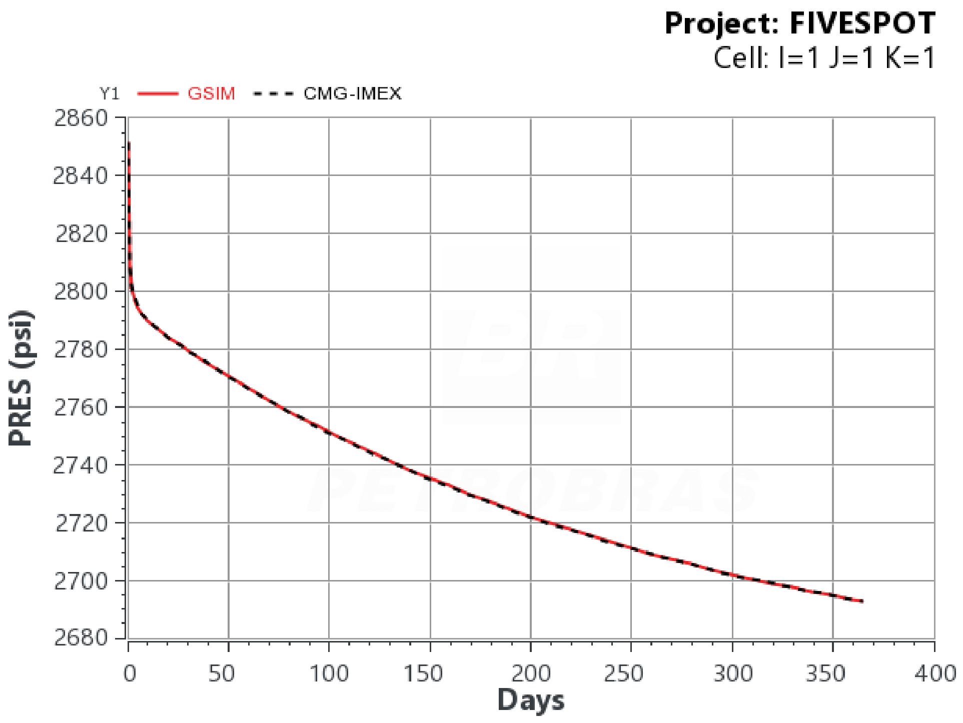

Figure 4.9: Oil pressure vs. time at cell (1,1,1).

PUC-Rio - Certificação Digital Nº 1912763/CA

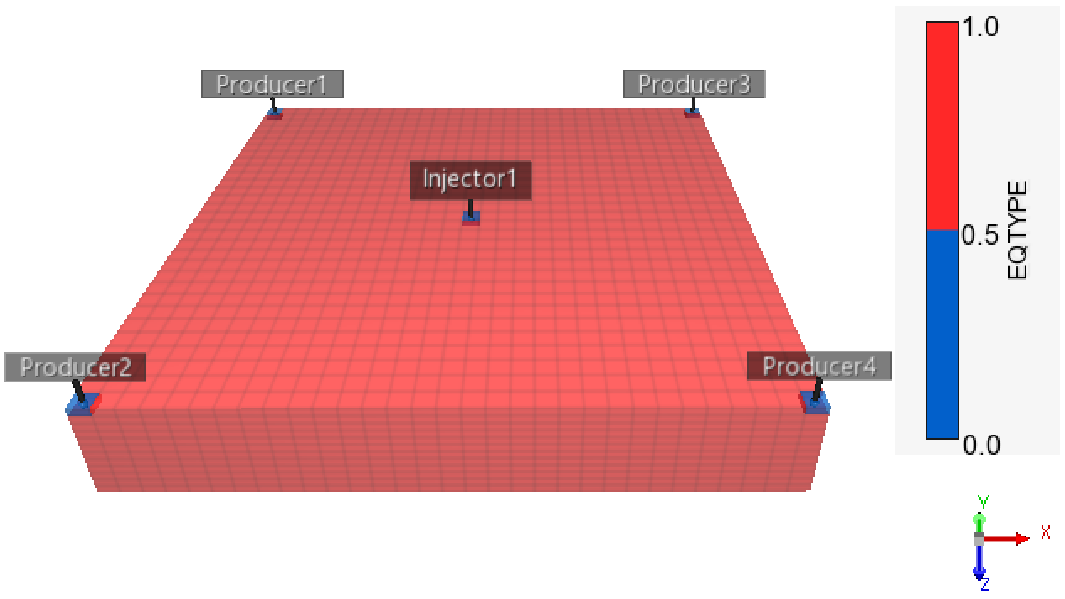

Figure 4.10: Implicitness map at t = 1 day, where red cells are IMPES and blue cells, FIM.

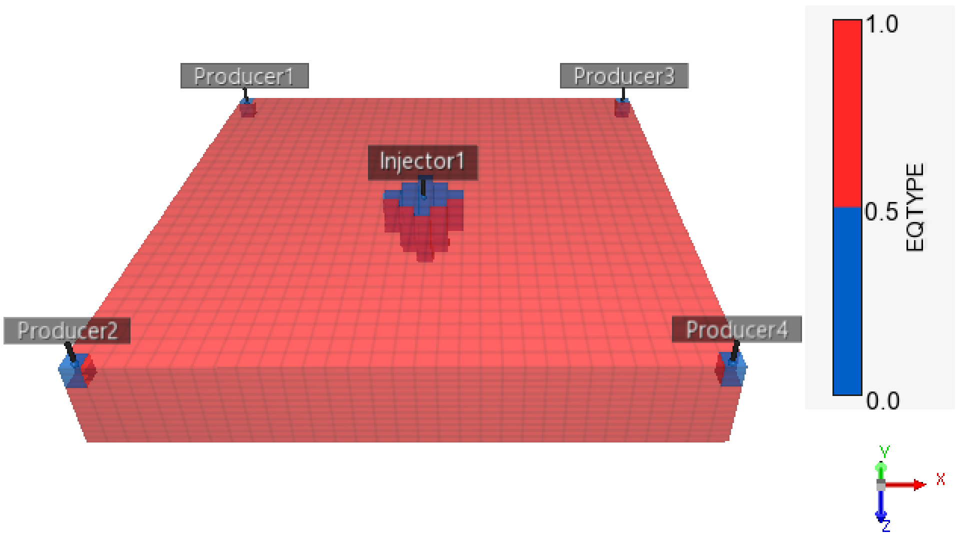

Figure 4.11: Implicitness map at t = 365 days, where red cells are IMPES and blue cells, FIM.

4.2 Amalgamated channels

The next validation example is a water injection in an heterogeneous reservoir model with rectangular dimensions of size $8202 \times 8202 \times 328$ ft in the x, y, and z directions, respectively. The absolute permeability map for the top layer is shown in Figure 4.12. Also, the placement of the wells and porosity map are given in Figure 4.13. All layers of the model follow the same pattern as the top layer for both absolute permeability and porosity maps.

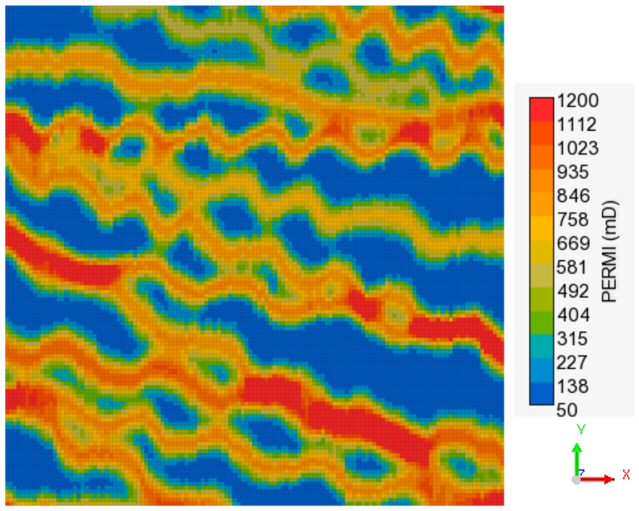

Figure 4.12: Absolute permeability of the top layer for the amalgamated channels case.

This example uses the same initial conditions as the previous model, as well as PVT (Table 4.2) and relative permeabilities (Figures 4.3 and 4.2) data.

PUC-Rio - Certificação Digital Nº 1912763/CA

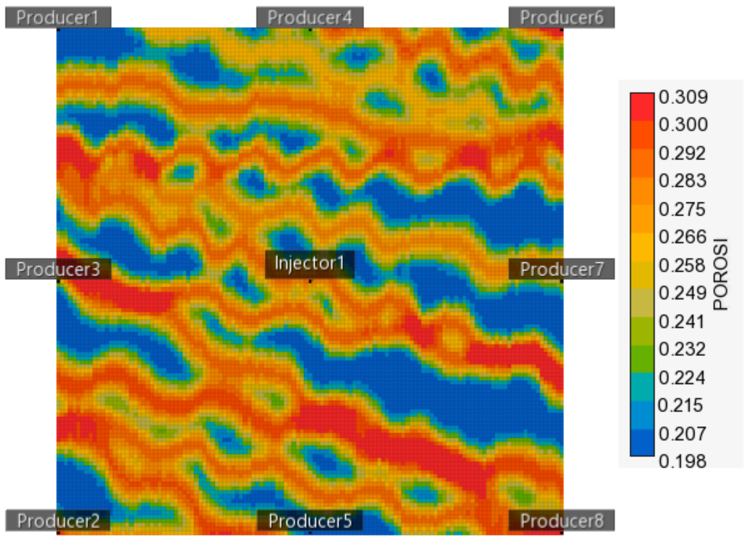

Figure 4.13: Porosity of the top layer for the amalgamated channels case.

This model uses a $125 \times 125 \times 4$ mesh, with a total number of 62500 cells. The well configuration consists of a nine-spot pattern, illustrated in Figure 4.13. All wells perforate multiple layers of the reservoir, where the producers perforate the two upper layers and the injector, the two lower layers.

This example was tailored for validating the features of multi-layer wells, different operating conditions, and scheduling well events. The events and operating conditions for each well are listed as follows:

- Producers 1, 2, 6, and 8: initially with BHPs of 2560 psia;
- Producers 3 and 7: initially with BHPs of 2560 psia. These wells are shut at $t = 30$ days and reopened at $t = 233$ days;
- Producers 4 and 5: initially with BHP of 2560 psia. These wells are shut at $t = 30$ days and reopened at $t = 60$ days;
- Injector 1: maximum water flow rate of 12579.6 STB/day and a maximum BHP of 3129 psia. This well is shut at $t = 106$ days and reopened at $t = 120$ days.

Figure 4.14 illustrates that, due to the heterogeneity of this example, the pressure distribution is unsymmetrical even though all producers are using the same operating conditions at $t = 25$ days.

Figure 4.15 shows the results of the water flow rate at each completion of the Injector and Figures 4.16 to 4.18 show the results of the oil flow rates at each completion of Producer 3, 4, and 8. Besides the results being similar to those of CMG-IMEX, the treatment of all scheduled well events works as expected.

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

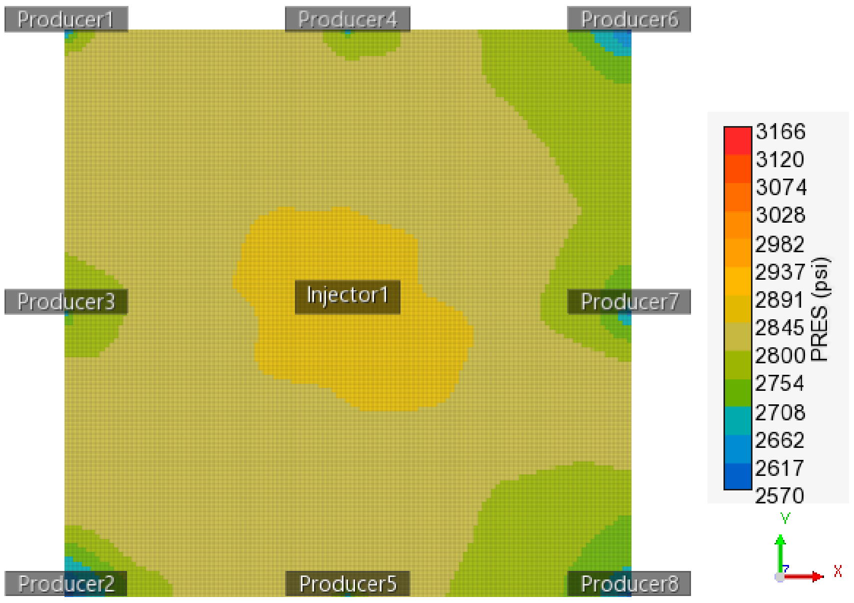

Figure 4.14: Pressure distribution, time = 25 days, top layer.

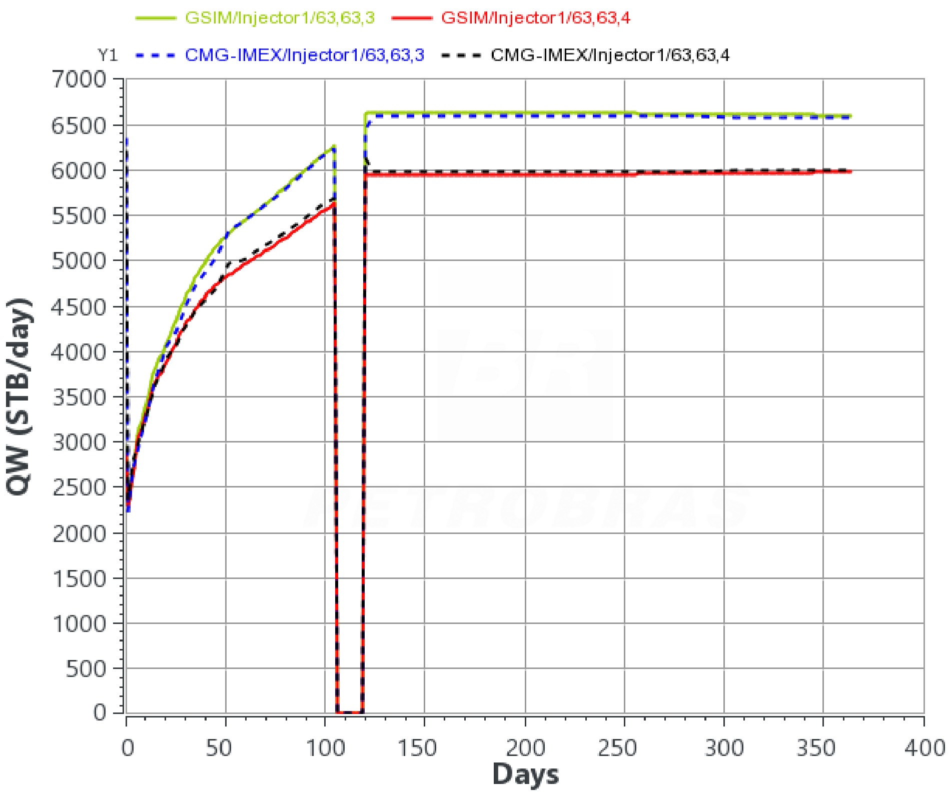

Figure 4.15: Water flow rate vs. time at Injector 1.

PUC-Rio - Certificação Digital Nº 1912763/CA

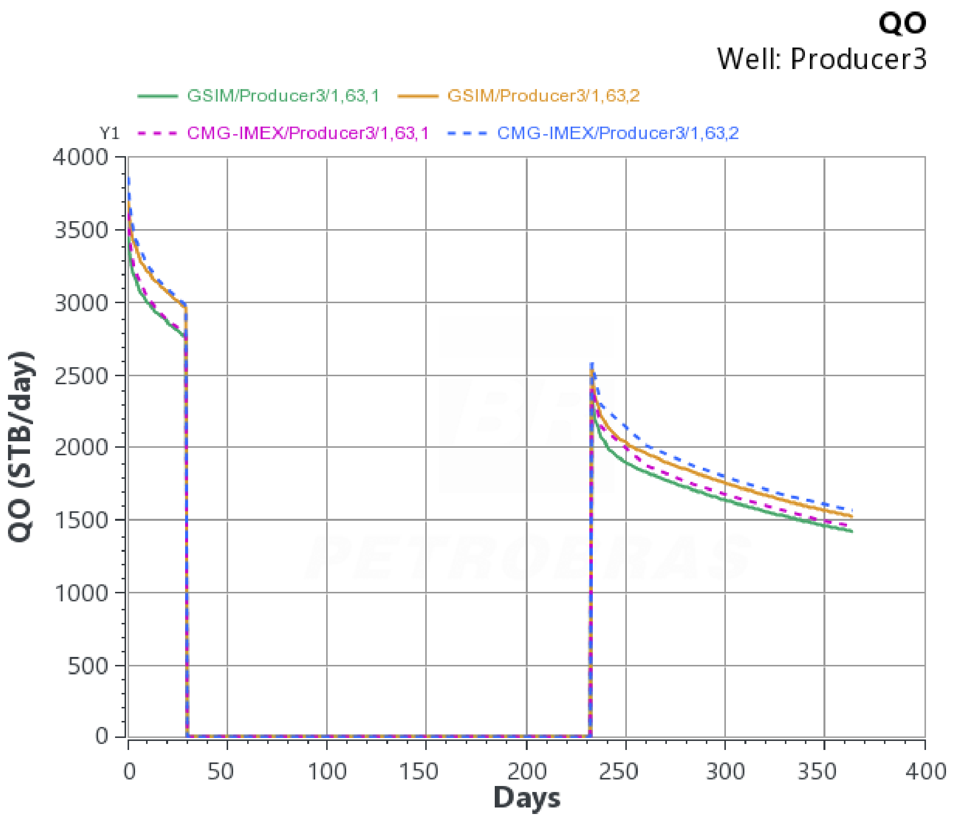

Figure 4.16: Oil flow rate vs. time at Producer 3.

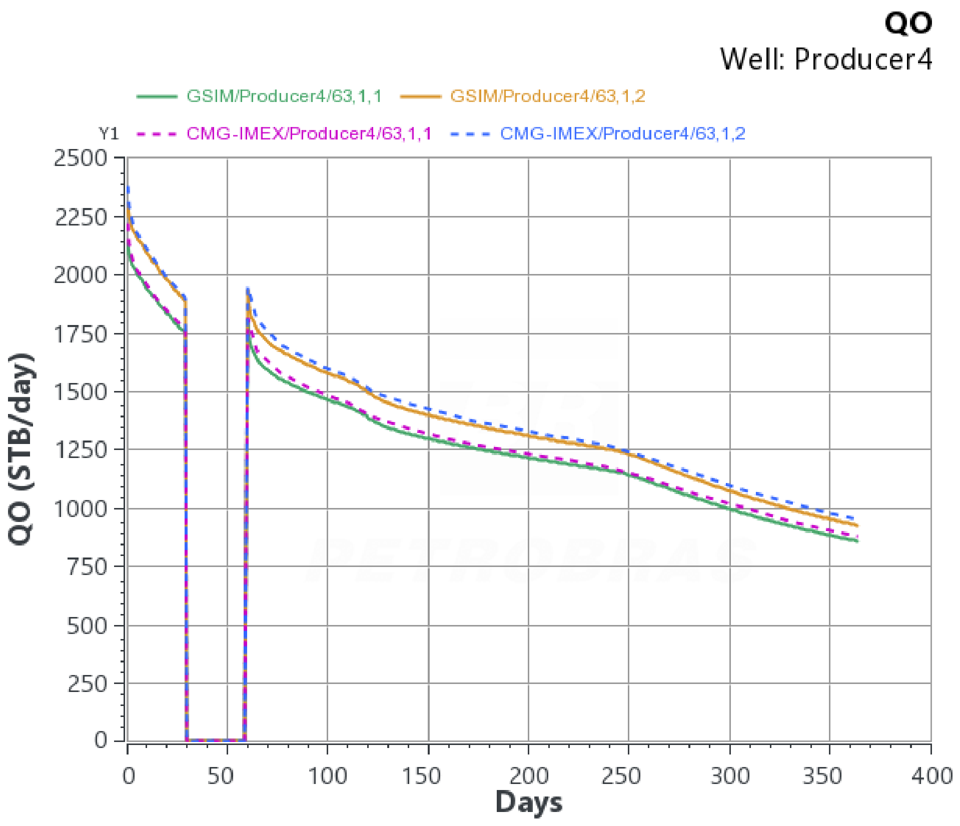

Figure 4.17: Oil flow rate vs. time at Producer 4.

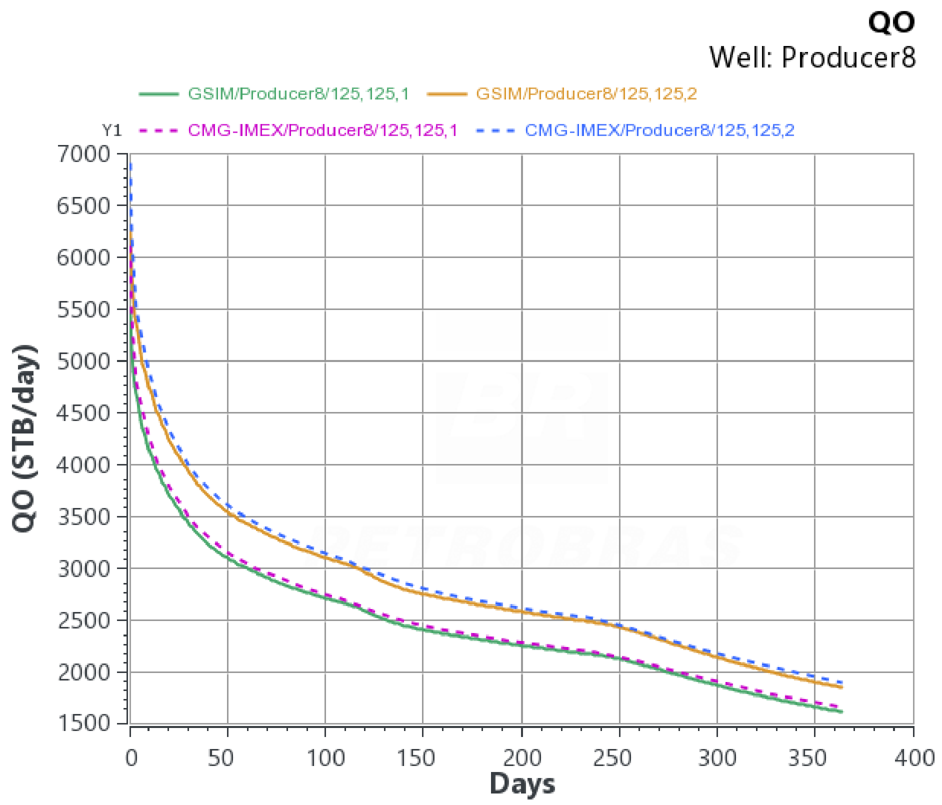

Figure 4.18: Oil flow rate vs. time at Producer 8.

PUC-Rio - Certificação Digital Nº 1912763/CA

4.3 First SPE-CSP

The third validation test is Odeh's first comparative-solution project [92]. This project involves a three-layer reservoir simulation with gas injection, as shown in Figure 4.19. The reservoir is assumed to be undersaturated, with PVT properties and relative permeabilities available in Tables 4.3, 4.4, and 4.5. The reservoir has a uniform initial porosity of 30% and is initially with a pressure of 8335 psia at top depth, while the initial bubble-point is 4014.7 psia.

Figure 4.20 shows the location of the wells. The injector is located at cell (1,1,1) and the producer at cell (10,10,3). The injector operates with a maximum gas flow rate of 100 MMscf/day and the producer operates with a maximum oil flow rate of 20000 STB/day and a minimum BHP of 1000 psia.

This model was chosen to validate two features of our simulator: gas injection and phase appearance/disappearance. Figures 4.21 to 4.24 compare the results from GSim against those from CMG-IMEX. It is possible to verify that all results are similar for this example. In addition, Figure 4.24 shows the gas saturation at the producer well increasing and decreasing over the whole simulation, validating our implementation of the phase appearance/disappearance treatment (Fig. 2.8).

PUC-Rio - Certificação Digital Nº 1912763/CA

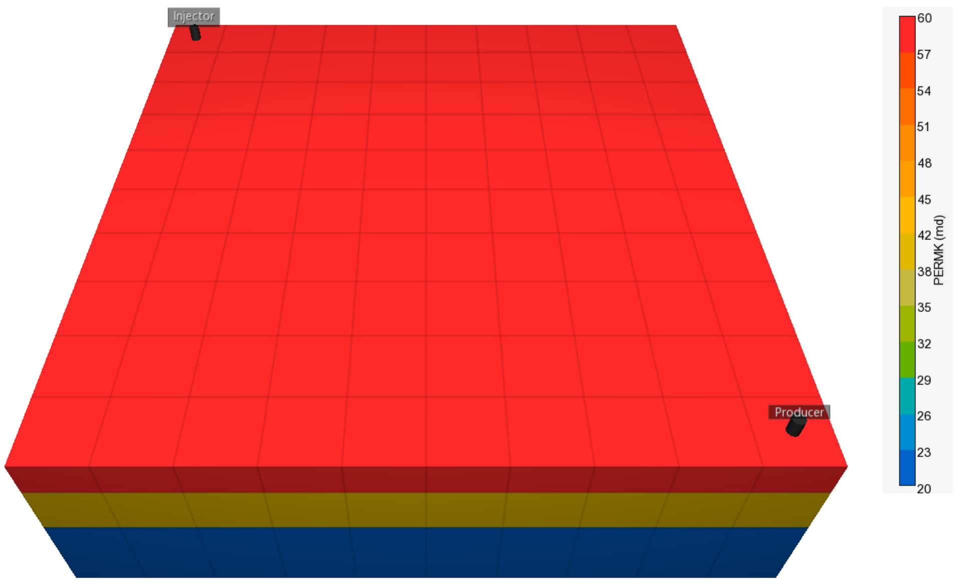

Figure 4.19: 3D view of the reservoir

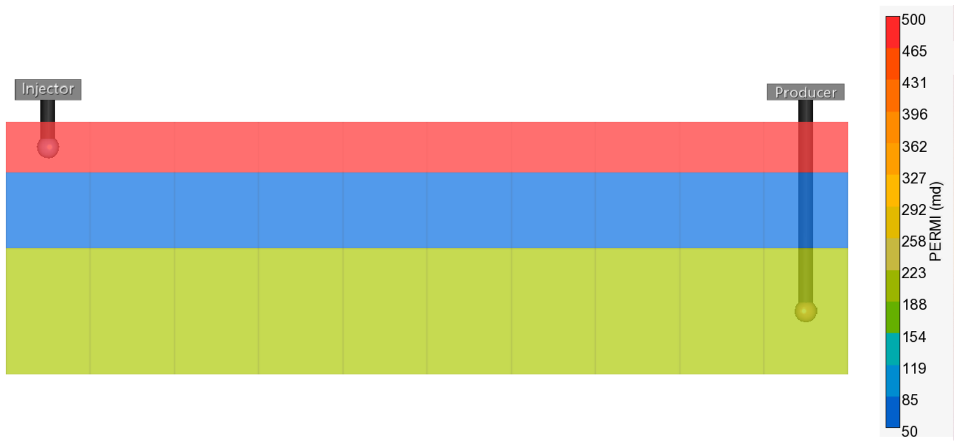

Figure 4.20: SPE 01 reservoir cross-section

Table 4.3: Gas-oil permeabilities as functions of gas saturation.

S_g	K_{rg}	K_{rog}
0.000	0.000	1.0000
0.001	0.000	1.0000
0.020	0.000	0.9970
0.050	0.005	0.9800
0.120	0.025	0.7000
0.200	0.075	0.3500
0.250	0.125	0.2000
0.300	0.190	0.0900
0.400	0.410	0.0210
0.450	0.600	0.0100
0.500	0.720	0.0010
0.600	0.870	0.0001
0.700	0.940	0.0000
0.850	0.980	0.0000
0.880	0.984	0.0000

Table 4.4: Water-oil permeabilities as functions of water saturation.

S_w	K_{rw}	K_{row}
0.12	0.0	1.0
0.82	0.0	0.0

Table 4.5: PVT properties of oil and gas.

P_o [psia]	R_s [scf/STB]	B_o [RB/STB]	E_g [bbl/RB]	μ_o [cp]	μ_g [cp]
14.7	1.0	1.0620	6.000	1.0400	0.008000
264.7	90.5	1.1500	82.700	0.9750	0.009600
514.7	180.0	1.2070	159.000	0.9100	0.011200
1014.7	371.0	1.2950	313.000	0.8300	0.014000
2014.7	636.0	1.4350	620.000	0.6950	0.018900
2514.7	775.0	1.5000	773.000	0.6410	0.020800
3014.7	930.0	1.5650	926.000	0.5940	0.022800
4014.7	1270.0	1.6950	1233.000	0.5100	0.026800
5014.7	1618.0	1.8270	1541.000	0.4490	0.030900
9014.7	2984.0	2.3570	2591.000	0.2030	0.047000

PUC-Rio - Certificação Digital Nº 1912763/CA

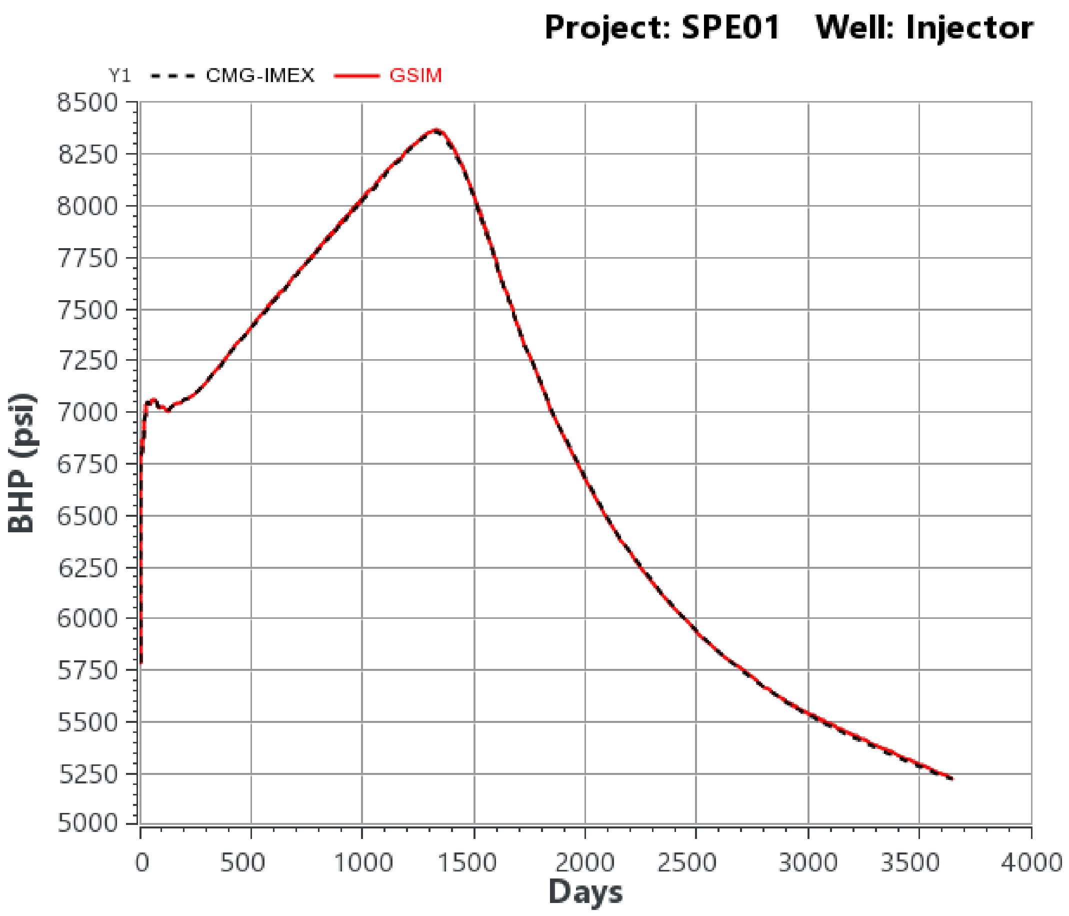

Figure 4.21: BHP of the injection well vs time.

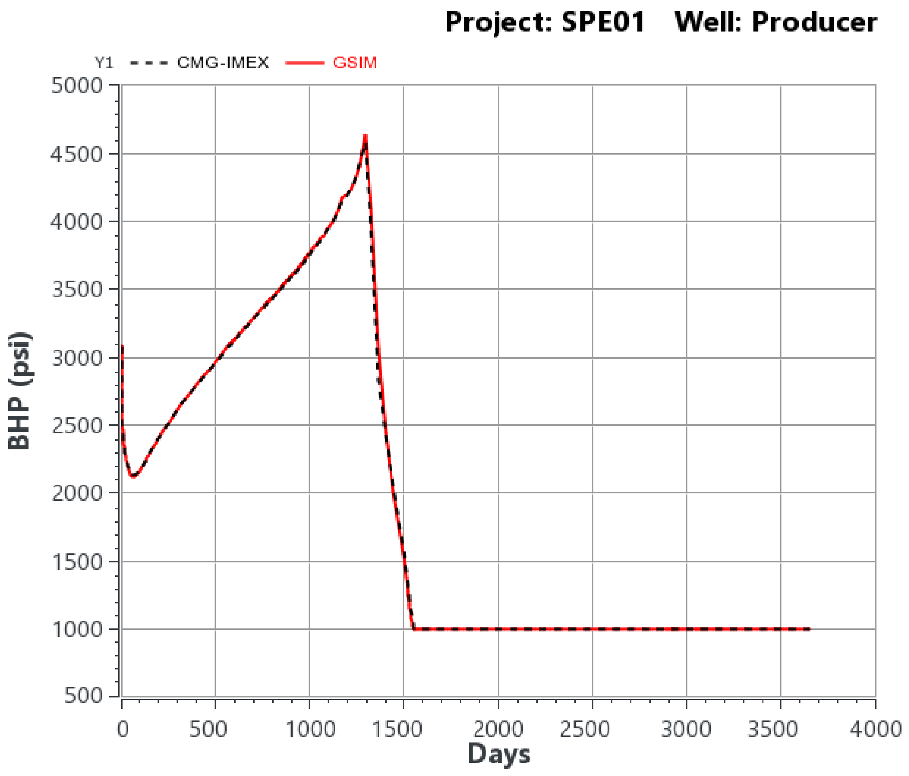

Figure 4.22: BHP of the producer well vs time.

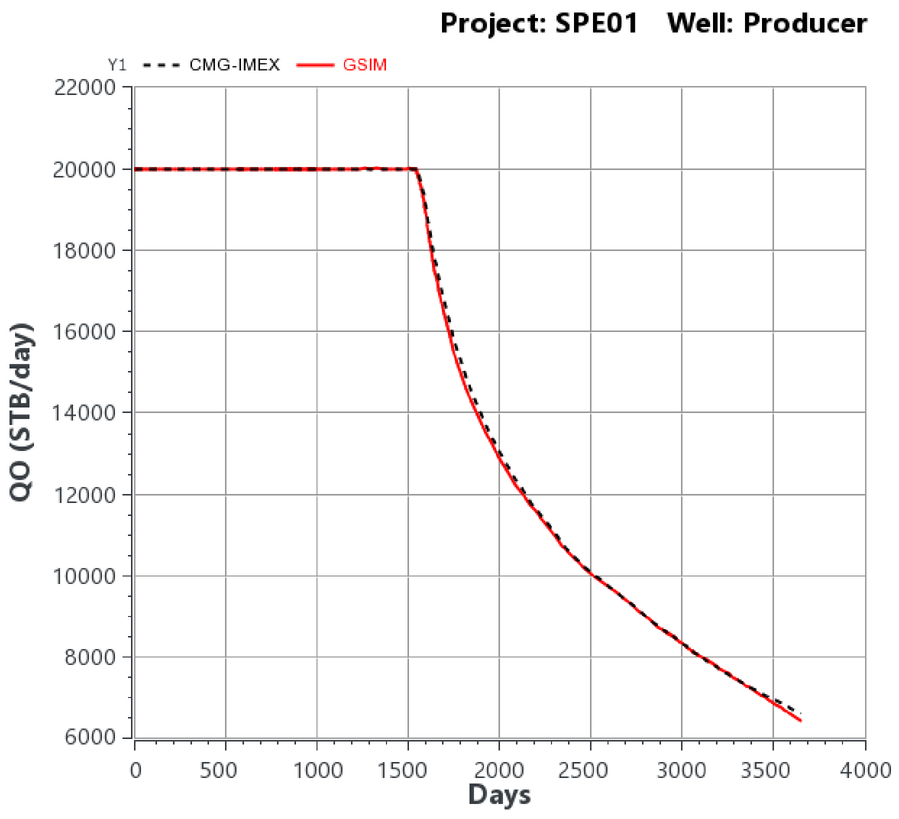

Figure 4.23: Oil flow rate vs time.

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

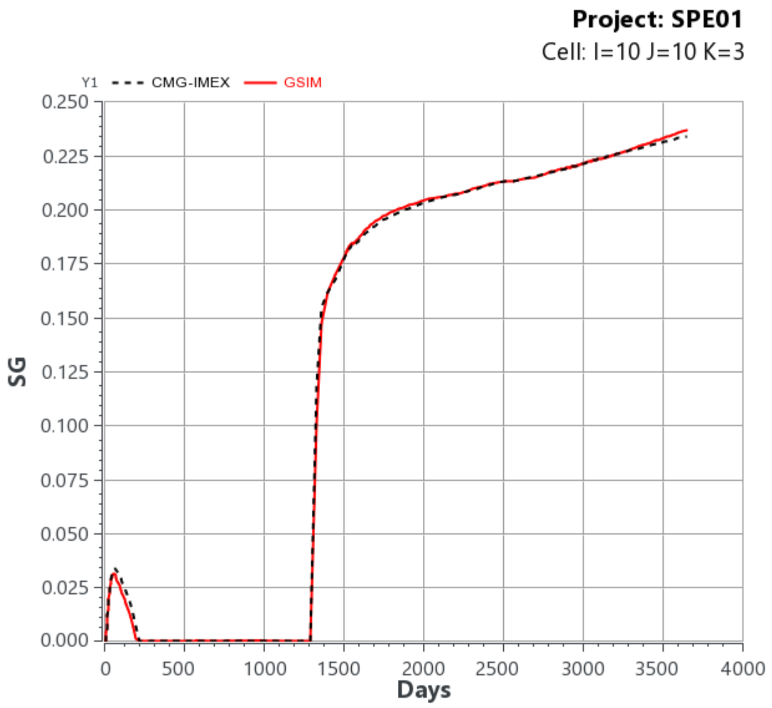

Figure 4.24: Gas saturation vs time at cell (10,10,3).

Besides validating against CMG-IMEX, we tried to compare GSim with the data from the participants of Odeh's paper. Figures 4.25 and 4.26 shows the comparison of the results for the simulators of different oil companies. From these figures, we concluded that our results are within the same range as those found in the original paper.

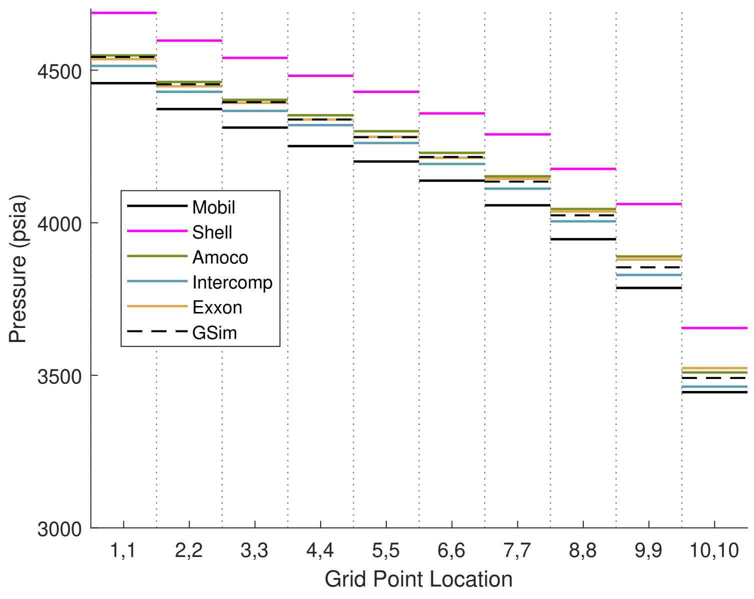

Figure 4.25: Pressure vs. grid-point location, time = 8 years, top layer.

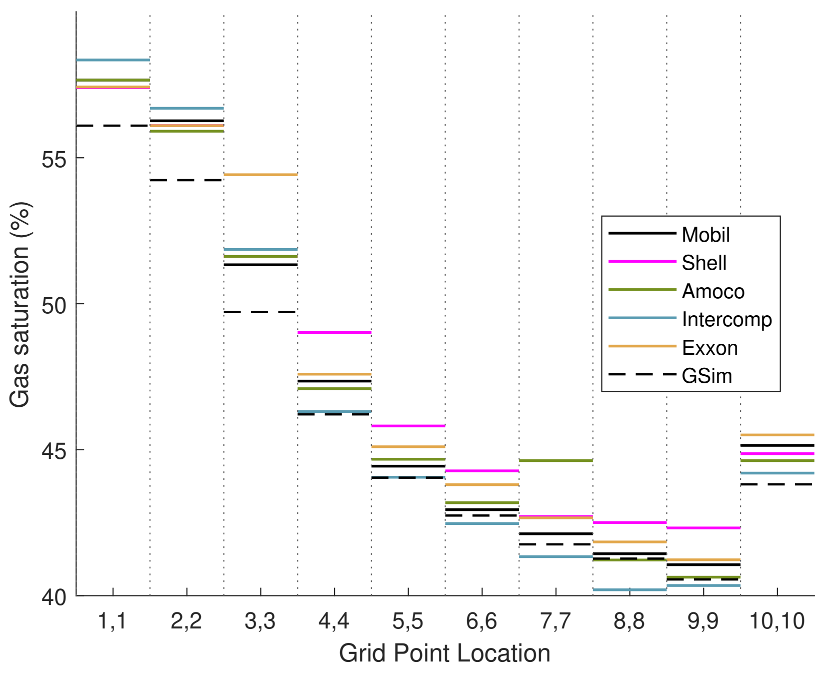

Figure 4.26: Gas saturation vs. grid-point location, time = 8 years, top layer.

PUC-Rio - Certificação Digital Nº 1912763/CA

4.4 Ninth SPE-CSP

Our final validation example is Killough's ninth CSP [93]. This example is described by a $24 \times 25 \times 15$ mesh with a $10°$ dip-angle in the x-direction, as shown in Figure 4.27. The reservoir is initially with undersaturated oil and water, where the water-oil contact is 915 ft below the top of the reservoir.

The well configuration is shown in Figure 4.28. The single water injector operates at a maximum rate of 5000 STB/day with a maximum BHP of 4000 psia. The 25 producers are initially set to a maximum rate of 1500 STB/day, which is lowered to 100 STB/day from day 300 to 360, and raised up again to its initial value until the end of simulation at 900 days. All producers are completed in layers 2 to 4 and the water injector is completed in layers 11 to 15.

Relative permeabilities and capillary pressure functions are shown in Figures 4.29 to 4.31. There is an interesting feature in the water-oil capillary pressure, which is the discontinuity at about 35% water saturation. This discontinuity can lead to difficulties in Newton method convergence for cases in which water saturations change significantly. Values for PVT properties for the oil and gas are given in Table 4.6.

PUC-Rio - Certificação Digital Nº 1912763/CA

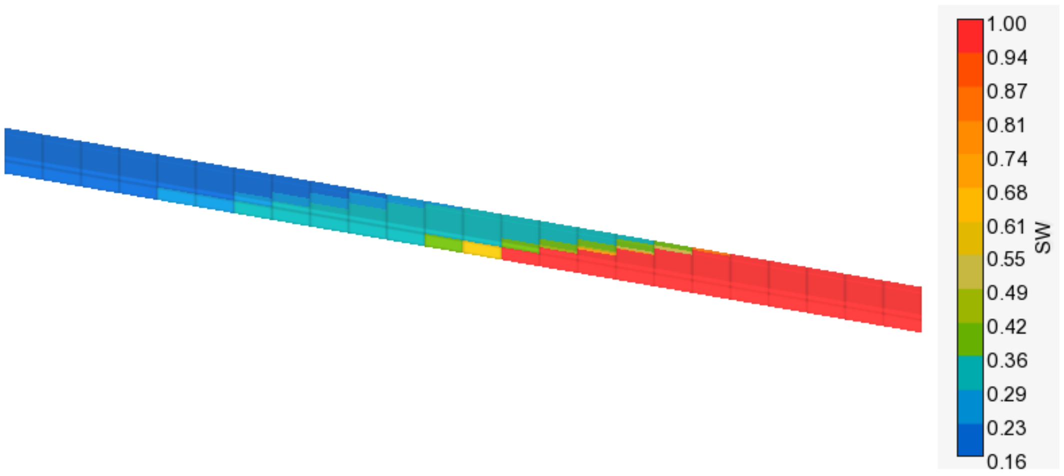

Figure 4.27: Side view of the ninth SPE-CSP

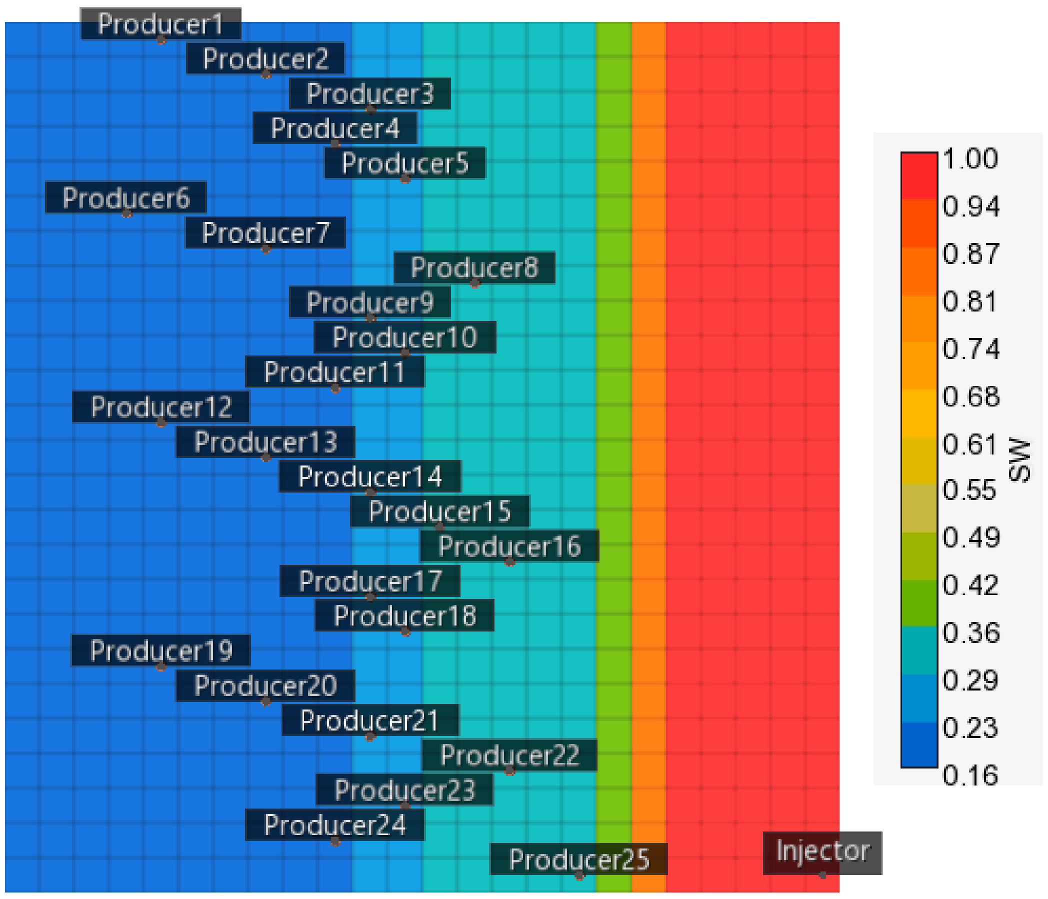

Figure 4.28: Top view of the ninth SPE-CSP

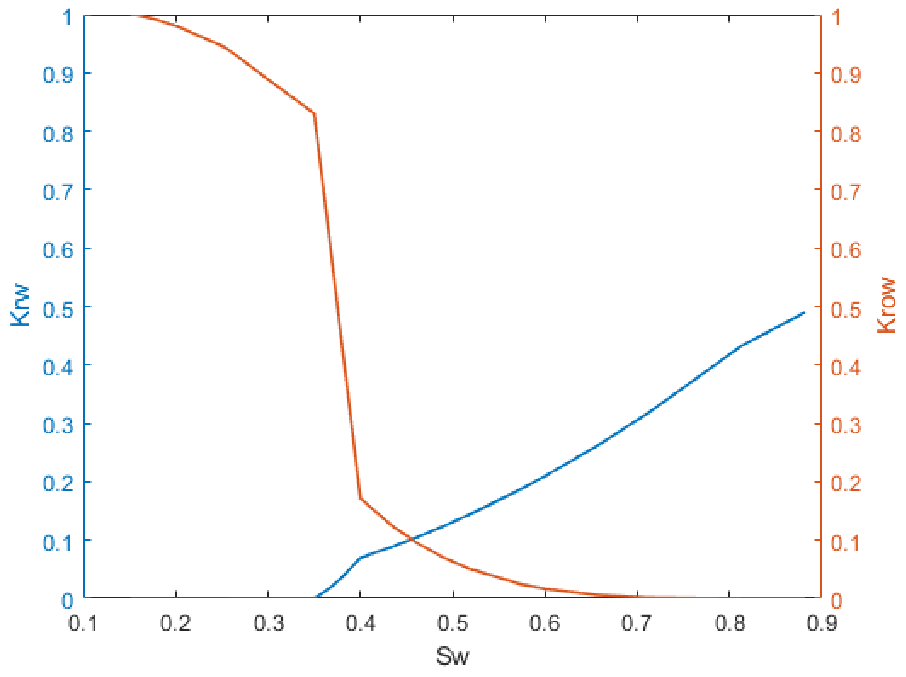

Figure 4.29: Water-oil relative permeabilities as functions of water saturation. These curves are non-smooth, as presented in Killough's paper [93].

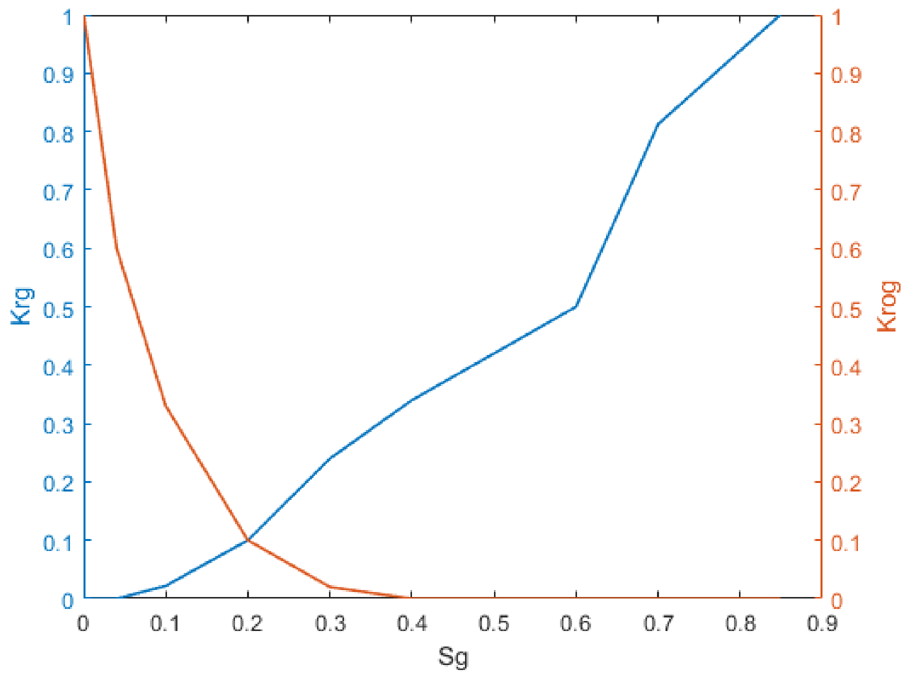

Figure 4.30: Gas-oil relative permeabilities as functions of gas saturation. These curves are non-smooth, as presented in Killough's paper [93]

PUC-Rio - Certificação Digital Nº 1912763/CA

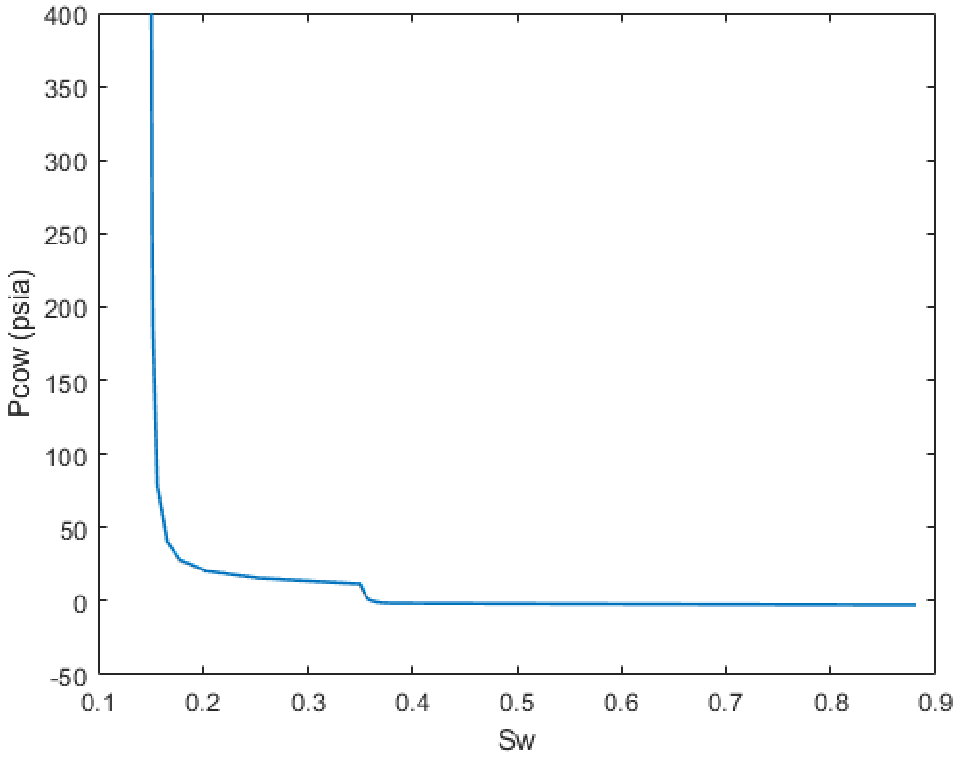

Figure 4.31: Water-oil capillary pressure as function of water saturation. This curve is non-smooth, as presented in Killough's paper [93]. There is a capillary pressure jump at $S_w = 0.35$, which can lead to numerical difficulties.

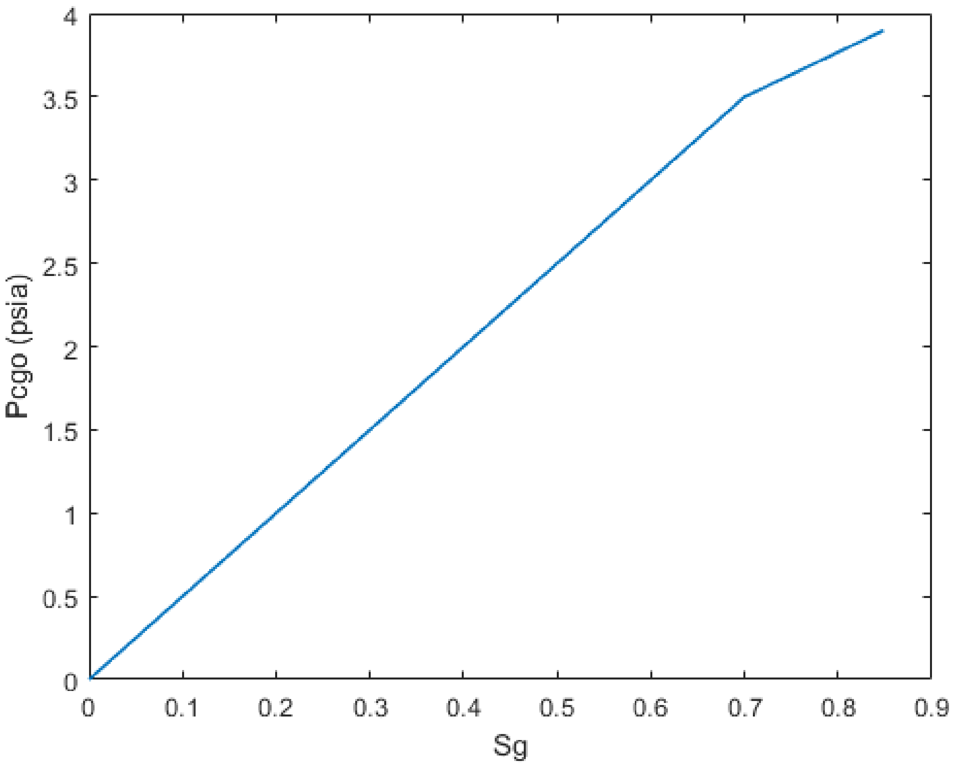

Figure 4.32: Gas-oil relative capillary pressure as function of gas saturation.

PUC-Rio - Certificação Digital Nº 1912763/CA

Table 4.6: PVT properties of oil and gas.

P_o [psia]	R_s [scf/STB]	B_o [RB/STB]	z_g [-]	μ_o [cp]	μ_g [cp]
14.7	0	1.0000	0.9999	1.20	0.0125
400	165	1.0120	0.8369	1.17	0.0130
800	335	1.0255	0.8370	1.14	0.0135
1200	500	1.0380	0.8341	1.11	0.0140
1600	665	1.0510	0.8341	1.08	0.0145
2000	828	1.0630	0.8370	1.06	0.0150
2400	985	1.0750	0.8341	1.03	0.0155
2800	1130	1.0870	0.8341	1.00	0.0160
3200	1270	1.0985	0.8398	0.98	0.0165
3600	1390	1.1100	0.8299	0.95	0.0170
4000	1500	1.1200	0.8300	0.94	0.0175
9000	1510	1.1210	0.8301	0.93	0.0176

PUC-Rio - Certificação Digital Nº 1912763/CA

The ninth SPE-CSP is the most complete example of this work, containing capillary pressures, phase appearance/disappearance, scheduled well events, different operating conditions, multi-layered wells, and a tilted reservoir. Figure 4.33 validates the oil-water capillary pressure. Also, it validates the robustness of GSim since it succeeded in simulating days 180 to 360, which have the discontinuity in capillary pressure.

As free gas is formed at cell (18,18,1), the gas-oil capillary pressure increases (Fig. 4.34), validating both gas-oil capillary pressure and phase appearance. Finally, Figures 4.35 and 4.36 show that our results are very close of those from CMG-IMEX for the flow rates at Injector 1 and Producer 2, respectively. In particular, Figure 4.36 shows the well events and restrictions forcing the operating conditions to change. At $t \approx 190$ days the Producer 2 changes from constant flow rate to BHP specification. Afterward, two well events happen: at $t = 300$ days, the well switches from BHP to a constant flow rate of 100 STB/day specification and at $t \approx 360$ it increases the constant flow rate, and at $t \approx 380$ it violates the restrictions, becoming BHP specified until the end of the analysis.

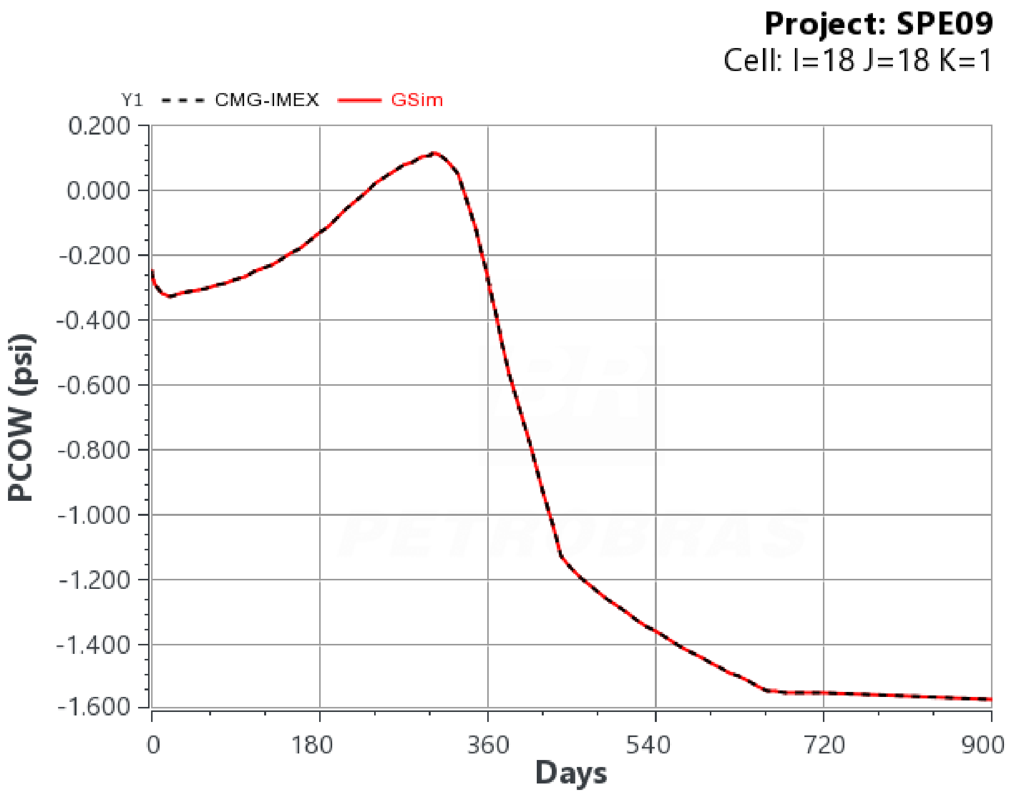

Figure 4.33: Oil-water capillary pressure vs time at cell (18,18,1).

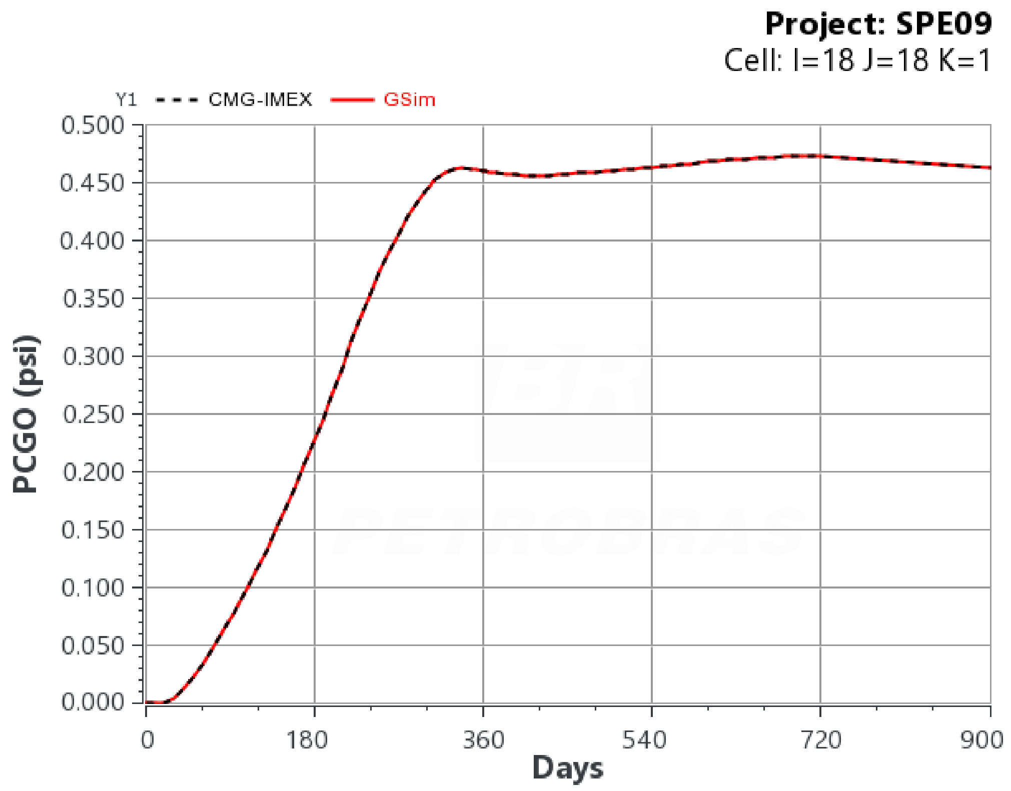

Figure 4.34: Gas-oil capillary pressure vs time at cell (18,18,1).

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

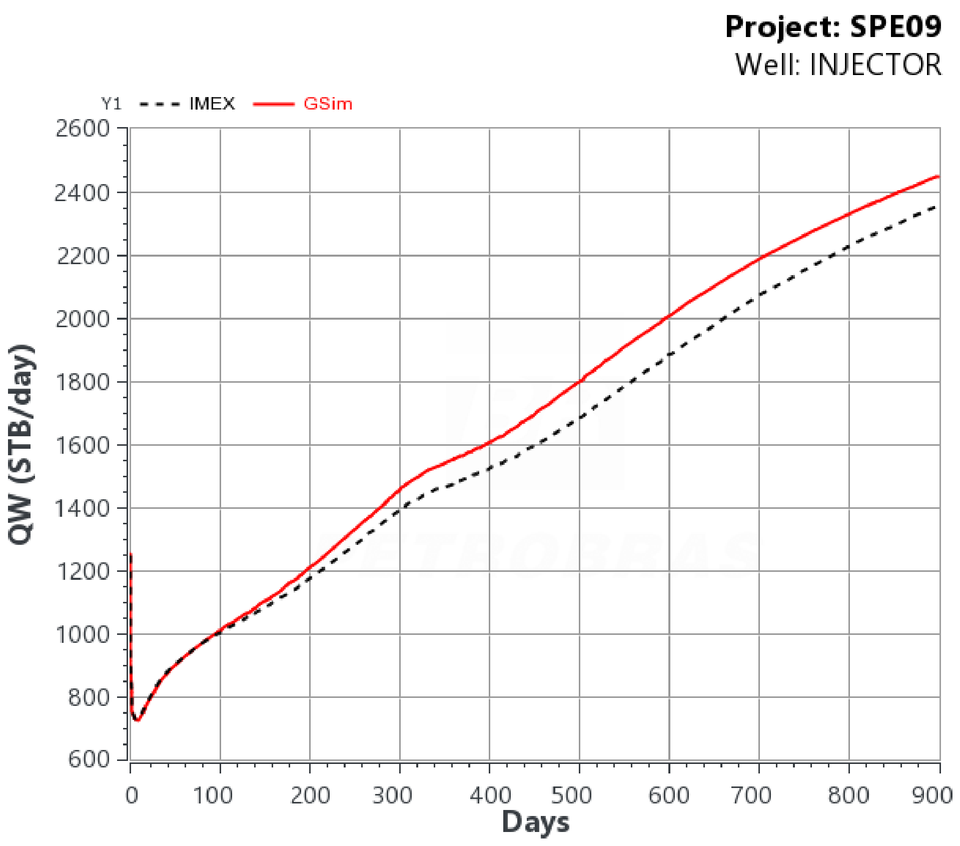

Figure 4.35: Water injection flow rate vs time at Injector 1.

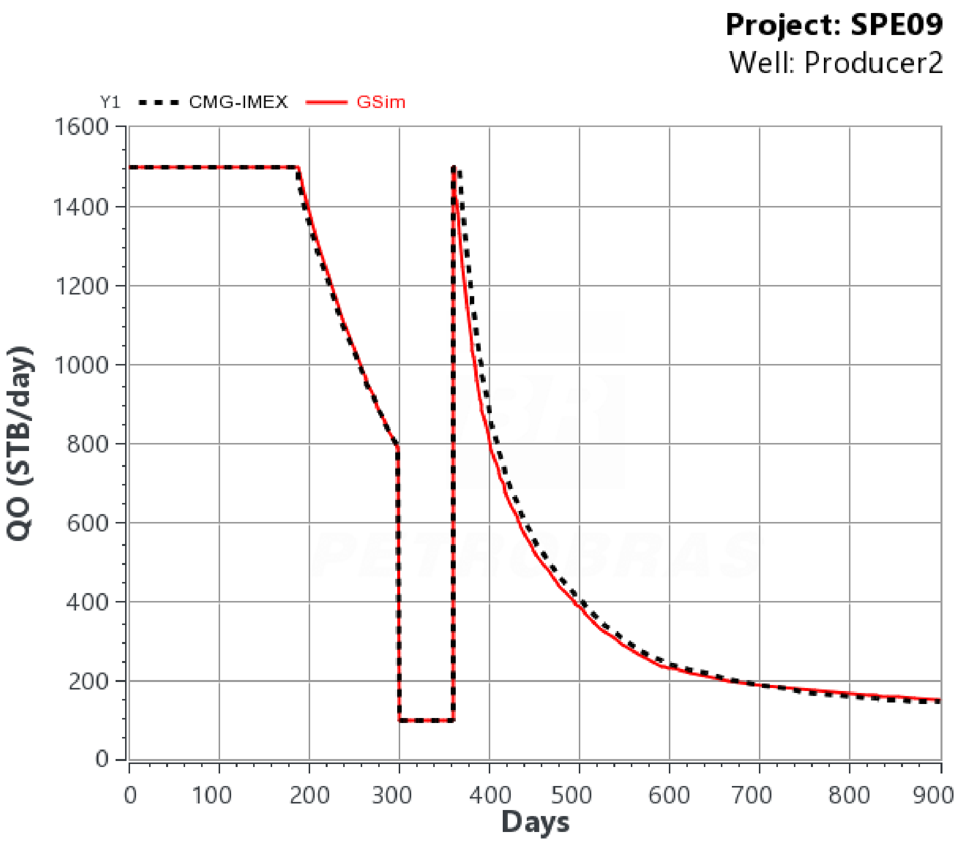

Figure 4.36: Oil flow rate vs time at Producer 2.

PUC-Rio - Certificação Digital Nº 1912763/CA

4.5 GSim showcase

All characteristics related to the use of GSim as a tool for predicting the behavior of petroleum reservoirs were validated. The final study of this thesis consists of showing that our modular approach allows us to use the simulator as a performance assessment tool for different numerical procedures.

In this section, we compare the traditional methods used in reservoir simulation with those from state-of-the-art implemented in this work, i.e., PPU vs. C1-PPU, Newton method vs. Inexact Newton, and Todd et al's [79] basic timestep controller vs. PID.

PPU vs. C1-PPU

Choosing PPU to treat the transmissibilities leads to non-differentiable flux functions when the flux direction changes during the nonlinear iteration process. These generate sudden changes in the behavior of the flux terms of the residual equations, creating oscillations in Newton's iterations that can lead to convergence failure [64]. The aim of C1-PPU is to mitigate this problem by approximating the original PPU by a C1-continuous function, as discussed in Chapter 2.

We chose the first SPE-CSP problem to perform the comparison between these methods, as the gas component appearance and disappearance causes changes in the flow direction within the nonlinear iterations. Three metrics are used in this comparison: the number of nonlinear iterations, number of timestep cuts, and number of nonlinear iterations discarded due to the timestep cuts.

Table 4.7: Comparison of PPU and C1-PPU for the First SPE-CSP.

		PPU	C1-PPU
First SPE-CSP	Nonlinear iterations	155	142
	Time-step cuts	10	9
	Wasted nonlinear iterations	8	8
Refined First SPE-CSP	Nonlinear iterations	347	246
	Time-step cuts	59	39
	Wasted nonlinear iterations	32	16

Table 4.7 shows the results for two versions of the first SPE-CSP, the original one and a refined version with a $10 \times 10 \times 60$ mesh. For the original

SPE-CSP, there is a reduction of 9% in the number of nonlinear iterations using C1-PPU, while for the refined version we have a reduction of 30% in the number of nonlinear iterations, 34% in the number of timestep cuts, and 50% in the number of wasted nonlinear iterations. These results agree with those from Jiang and Younis [64], indicating that C1-PPU is a promising alternative for increasing the nonlinear performance of a reservoir simulation.

Classical Newton vs. Inexact Newton methods

The problem of *oversolving* arises when the initial guess is not sufficiently close to the solution. This numerical phenomenon makes the linear solver to perform several iterations unnecessarily. For this reason, the Inexact Newton has been widely used, as it finds the nonlinear solution by changing the tolerance of the linear solver depending on how close to the nonlinear solution it is.

Two scenarios were chosen to compare these methods. The first scenario analyzes the performance of both algorithms for an example that simulates a long production time. The second scenario, on the other hand, analyzes their performance when solving the linear system of equations at each timestep is computationally expensive. Thus, the first SPE-CSP was chosen for the first scenario, while for the second we chose the amalgamated channels example.

PUC-Rio - Certificação Digital Nº 1912763/CA

Table 4.8: Comparison of Newton and Inexact Newton methods.

		Classical Newton	Inexact Newton
First SPE-CSP	Nonlinear iterations	115	160
	Linear iterations	3389	945
	Total runtime (s)	0.6	0.5
Amalgamated channels	Nonlinear iterations	52	53
	Linear iterations	5808	435
	Total runtime (s)	155.2	34.5

Table 4.8 shows the results for both scenarios using BICGSTAB iterative solver with ILU preconditioner. They indicate that, besides having an increase in the number of nonlinear iterations, the runtime of the simulation is 17% smaller for the first SPE-CSP and 77% smaller for the amalgamated channels, when using Inexact Newton. This is due to a huge decrease of the total number of linear iterations, which is the cause of the *oversolving* phenomenon.

Besides comparing these different nonlinear solvers, we investigated the influence of the choice of a linear solver when using the Inexact Newton method. We made this tests for the amalgamated channels examples using BICGTAB and GMRES, both with ILU as a preconditioner.

Table 4.9: Comparison of different linear solvers when using Inexact Newton.

	BICGSTAB	GMRES
Nonlinear iterations	53	61
Linear iterations	435	952
Total runtime (s)	34.5	51.3

Table 4.9 shows that BICGSTAB uses 32% less runtime and 13% less nonlinear iterations than GMRES. Thus, different linear solvers not only influence in the number of linear iterations and simulation execution time, but also influence the number of Inexact Newton iterations.

Basic vs. PID adaptive timestepping

The final comparison aims to study the influence of different techniques in adaptive timestepping for reservoir simulation. We investigated the impact of different timestep controllers when using FIM and AIM formulations using the First SPE-CSP case.

Table 4.10: Comparison of Basic and PID time-step controllers for different formulations.

		Basic	PID
First SPE-CSP AIM	Nonlinear iterations	253	251
	Total runtime (s) per nonlinear solver iter.	5.1×10^{-3}	5.05×10^{-3}
	Timestep cuts	0	0
	Wasted nonlinear iterations	0	0
First SPE-CSP FIM	Nonlinear iterations	115	106
	Total runtime (s) per nonlinear solver iter.	1.042×10^{-2}	1.2×10^{-2}
	Timestep cuts	11	17
	Wasted nonlinear iterations	9	11

Table 4.10 shows the results for the first comparison. We see that the PID controller is a promising alternative for adaptive timestepping in reservoir simulations, as its results led to a few number of nonlinear iterations for both formulations. In addition, it is worth mentioning that the computational cost of AIM is lower than FIM, as expected.

Finally, we have tested both timestep controllers for the ninth SPE-CSP. Table 4.11 shows the results of this comparison. Unfortunately, there is no

PUC-Rio - Certificação Digital Nº 1912763/CA

significant difference between the two approaches in this case due to the number of changes of the operating conditions, which forces the simulator to reset the timestep size. Therefore, we conclude that the efficiency of these techniques is highly dependent on the case which one wants to solve.

Table 4.11: Comparison of Basic and PID time-step controllers for ninth SPE-CSP problem.

		Basic	PID
	Nonlinear iterations	270	269
Ninth SPE-CSP FIM	Wasted nonlinear iterations	4	4
	Timestep cuts	11	11

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

5 Conclusion

This work presents a multipurpose reservoir simulator based on a plugin architecture. The simulator provides an extensible modular tool to forecast and analyze different aspects of oil and gas recovery. A three-dimensional black-oil model was implemented in the simulator with a wide range of configurations, including traditional and state-of-the-art methods.

We discussed the plugin architecture of the simulator, presenting the tool on which it is built upon, data structures, and implementation details. The implementation of different formulations, adaptive timestepping strategies, nonlinear solvers, and treatments of transmissibilities demonstrated the flexibility of the plugin approach. Also, the simulator supports several functionalities related to recovery strategies such as the injection of different fluids, multi-completion wells, and changes in the operating conditions.

The results were obtained for different test cases and compared against CMG-IMEX simulator, which demonstrated the correctness and feasibility of the proposed approach. Finally, these test cases showed how one could use GSim to easily compare different numerical methods.

5.1 Future work

We present some suggestions for future research, as follows:

– **Extending the simulator for compositional models.** The black-oil model is sufficiently accurate for reservoirs which have simple types of fluid and phase behaviors, e.g., viscosities, densities, FVFs, and gas-oil ratio are empirical functions of pressure. However, this model is not representative for reservoirs and recovery processes which require a thermodynamic treatment, specially when EOR is utilized. In this context, K-values and flash calculations are required to compute the gas-oil equilibrium and the physical properties that were dependent only on pressure in the isothermal black-oil model become dependent on temperature and chemical composition as well [94].

- **Automatic control in the context of adaptive timestepping.** In reservoir simulation, most adaptive timestep controllers require manually adjusting constant empirical factors in order to have a satisfactory performance [79, 82]. Taking advantage of system identification techniques for estimating the changes in the simulation to adapt such factors automatically may not only decrease the runtime of the simulation, but also improve the convergence behavior of the nonlinear iterations [95].
- **Implementation of globalization nonlinear solving methods.** Since multiphase flow in porous media is a strong nonlinear problem, the standard Newton method is convergent only when "sufficiently small" timesteps are taken and the initial guess is close to the solution. Thus, studying globally convergent methods, where convergence is independent of the initial guess, is needed to improve the convergence behavior of the simulator [96–98].
- **Distributed computing on clusters of CPUs and GPUs.** The computational cost of a reservoir simulation is prohibitive when high resolution models are required. For this reason, using massive parallel units for simulating reservoirs is a common practice in the industry.

PUC-Rio - Certificação Digital Nº 1912763/CA

Bibliography

[1] DEB, P. K.; AKTER, F.; IMTIAZ, S. A.; HOSSAIN, M. E. **Nonlinearity and solution techniques in reservoir simulation: a review**. Journal of Natural Gas Science and Engineering, 46:845–864, 2017.

[2] NACUL, E. C.; LEPETRE, C.; PEDROSA, O. A.; GIRARD, P.; AZIZ, K. **Efficient use of domain decomposition and local grid refinement in reservoir simulation**. In: SPE ANNUAL TECHNICAL CONFERENCE AND EXHIBITION, p. 245–256, New Orleans (LA, USA), 1990.

[3] PEDROSA, O. A.; AZIZ, K. **Use of hybrid grid in reservoir simulation**. SPE Reservoir Engineering, 1(6):611–621, 1986.

[4] DING, Y.; LEMONNIER, P. **Use of corner point geometry in reservoir simulation**. In: INTERNATIONAL MEETING ON PETROLEUM ENGINEERING, p. 451–461, Beijing (China), 1995.

[5] HEINEMANN, Z. E.; BRAND, C.; MUNKA, M.; CHEN, Y. M. **Modeling reservoir geometry with irregular grids**. In: SPE SYMPOSIUM ON RESERVOIR SIMULATION, p. 37–54, Houston (TX, USA), 1991.

[6] KARIMI-FARD, M.; DURLOFSKY, L. J.; AZIZ, K. **An efficient discrete fracture model applicable for general purpose reservoir simulators**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 1–11, Houston (TX, USA), 2003.

[7] OLORODE, O. M.; FREEMAN, C. M.; MORIDIS, G. J.; BLASINGAME, T. A. **High-resolution numerical modeling of complex and irregular fracture patterns in shale-gas reservoirs and tight gas reservoirs**. SPE Reservoir Evaluation & Engineering, 16:443–455, 2013.

[8] SUN, J.; SCHECTER, D. **Optimization-based unstructured meshing algorithms for simulation of hydraulically and naturally fractured reservoirs with variable distribution of fracture aperture, spacing, length, and strike**. SPE Reservoir Evaluation & Engineering, 18:463–480, 2015.

PUC-Rio - Certificação Digital Nº 1912763/CA

PUC-Rio - Certificação Digital Nº 1912763/CA

[9] YANG, C.; KING, M.; DATTA-GUPTA, A. **Rapid simulation of naturally fractured unconventional reservoir with unstructured grids using the fast marching method**. In: SPE RESERVOIR SIMULATION CONFERENCE, p. 1–21, Montgomery (TX, USA), 2017.

[10] ISLAM, M. R.; MOUSSAVIZADEGAN, S. H.; MUSTAFIZ, S.; ABOU-KASSEM, J. H. **Advanced Petroleum Reservoir Simulation**. Scrivener, Salem (USA), 2010.

[11] COATS, K. H. **Reservoir simulation: State of the art**. Journal of Petroleum Technology, 34:1633–1642, 1982.

[12] HURTADO, F. S. V.; MALISKA, C. R.; SILVA, A. F. C.; CORDAZZO, J. **A quadrilateral element-based finite-volume formulation for the simulation of complex reservoirs**. In: LATIN & CARIBBEAN PETROLEUM ENGINEERING CONFERENCE, p. 1–10, Buenos Aires (Argentina), 2007.

[13] CHEN, Z.; HUAN, G.; MA, Y. **Computational Methods for Multiphase Flows in Porous Media (Computational Science and Engineering 2)**. Society for Industrial and Applied Mathematics, USA, 2006.

[14] HJEIJ, D.; ABUSHAIKHA, A. **Comparing advanced discretization methods for complex hydrocarbon reservoirs**. In: SPE RESERVOIR CHARACTERISATION AND SIMULATION CONFERENCE AND EXHIBITION, p. 1–11, Abu Dhabi (UAE), 2019.

[15] DOGRU, A. H.; LARRY, S. K.; AL-SHAALAN, T. M.; MIDDYA, U.; PITA, J. A. **From mega cell to giga cell reservoir simulation**. In: SPE ANNUAL TECHNICAL CONFERENCE AND EXHINBITION, p. 1–19, Denver (CO, USA), 2008.

[16] DOUGLAS, J.; PEACEMAN, D. W.; RACHFORD, H. H. **A method for calculating multi-dimensional immiscible displacement**. Transactions of AIME, 216(1):297–308, 1959.

[17] SHELDON, J. W.; ZONDEK, B.; CARDWELL, W. T. **One-dimensional, incompressible, noncapillary, two-phase fluid flow in a porous medium**. Transactions of AIME, 216(1):290–296, 1959.

[18] THOMAS, G. W.; THURNAU, D. H. **Reservoir simulation using an adaptive implicit method**. Society of Petroleum Engineers Journal, 23(5):759–768, 1983.

PUC-Rio - Certificação Digital Nº 1912763/CA

[19] ERTEKIN, T.; ABOU-KASSEM, J. H.; KING, G. R. **Basic Applied Reservoir Simulation**. SPE textbook series. Society of Petroleum Engineers, USA, 2001.

[20] MONCORGÉ, A.; TCHELEPI, H. A.; JENNY, P. **Sequential fully implicit formulation for compositional simulation using natural variables**. Journal of Computational Physics, 371:690–711, 2018.

[21] MONCORGÉ, A.; MØYNER, O.; TCHELEPI, H. A.; JENNY, P. **Consistent upwinding for sequential fully implicit multiscale compositional simulation**. Computational Geosciences, 24(2):533–550, 2020.

[22] PRICE, H. S.; COATS, K. H. **Direct methods in reservoir simulation**. Society of Petroleum Engineers Journal, 14:295–308, 1974.

[23] OGBUOBIRI, E. C.; TINNEY, W. F.; WALKER, J. W. **Sparsity-directed decomposition for gaussian elimination on matrices**. IEEE Transactions on Power Apparatus and Systems, 89(1):141–150, 1970.

[24] PEACEMAN, D. W.; RACHFORD, H. H. **The numerical solution of parabolic and elliptic differential equations**. Journal of the Society for Industrial and Applied Mathematics, 3(1):28–41, 1955.

[25] YOUNG, D. **Iterative methods for solving partial differential equations of elliptic types**. Transactions of the American Mathematical Society, 76(1):92–111, 1954.

[26] WEINSTEIN, H. G.; STONE, H. L.; KWAN, T. V. **Iterative procedure for solutions pf systems of parabolic and elliptic equations in three dimensions**. Industrial & Engineering Chemistry Fundamentals, 8(2):281–287, 1969.

[27] VINSOME, P. K. W. **Orthomin, an iterative method for solving sparse sets of simultaneous linear equations**. In: SPE SYMPOSIUM OF NUMERICAL SIMULATION OF RESERVOIR PERFORMANCE, Los Angeles (USA), 1976.

[28] SAAD, Y.; SCHULTZ, M. H. **GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems**. SIAM Journal of Scientific and Statistical Computing, 7(3):856–869, 1986.

[29] VAN DER VORST, H. A. **Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems**. SIAM Journal on Scientific and Statistical Computing, 13(2):631–644, 1992.

[30] BENZI, M. **Preconditioning techniques for large linear systems: A survey**. Journal of Computational Physics, 182:418–477, 2002.

[31] APPLEYARD, J. R; CHESHIRE, I. M. **Nested factorization**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 315–320, San Francisco (USA), 1983.

[32] BEHIE, H. A.; VINSOME, P. K. W. **Block iterative methods for fully implicit reservoir simulation**. Society of Petroleum Engineers Journal, 22:658–668, 1992.

[33] WALLIS, J. R.; KENDALL, R. P.; LITTLE, T. E. **Constrained Residual Acceleration of Conjugate Residual Methods**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 415–426, Dallas (USA), 1985.

[34] BROWN, G. L.; COLLINS, D. A.; CHEN, Z. **Efficient preconditioning for algebraic multigrid and red-black ordering in adaptive-implicit black-oil simulations**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 1–13, Houston (TX, USA), 2015.

[35] TORONYI, R. M.; FAROUQ ALI, S. M. **Two-phase, two dimensional simulation of a geothermal reservoir**. SPE J., 17:171–183, 1977.

[36] POPE, G. A.; NELSON, R. C. **A chemical flooding compositional simulator**. Society of Petroleum Engineers Journal, 18:339–354, 1978.

[37] STRINBEI, C.; DOSPINESCU, O.; STRAINU, R. M.; NISTOR, A. **Software architectures - present and visions**. Informatica Economica, 19(4):13–27, 2015.

[38] TRAVASSOS, G.; SHULL, F.; FREDERICKS, M.; BASILI, V. R. **Detecting defects in object-oriented designs: using reading techniques to increase software quality**. In: 14TH CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, SYSTEMS, LANGUAGES AND APPLICATIONS, p. 47–56, New York, 1999.

[39] MANSOOR, U.; KESSENTINI, M.; MAXIM, B. R.; DEB, K. **Multi-objective code-smells detection using good and bad design examples**. Software Quality Journal, 25:529–552, 2017.

[40] DEB, M. K.; REDDY, M. P. **A new generation solution adaptive reservoir simulator**. In: SPE ANNUAL TECHNICAL CONFERENCE & EXHIBITION, p. 175–189, Dallas (TX, USA), 1995.

PUC-Rio - Certificação Digital Nº 1912763/CA

[41] PARASHAR, M.; WHEELER, J. A.; WANG, K.; WANG, P. **A new generation EOS compositional reservoir simulator: Part II - framework and multiprocessing**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 31–38, Dallas (TX, USA), 1997.

[42] GREIN, E. A. **A parallel computing approach applied in petroleum reservoir simulation**. Masters thesis, Universidade Federal de Santa Catarina, Florianópolis, Brazil, 2015.

[43] CAO, H. **Development of techniques for general purpose simulators**. Phd thesis, Stanford, 2002.

[44] MOLANO, H. E. B.; SEPEHRNOORI , K. **Development of a framework for parallel reservoir simulation**. International Journal of High Performance Computing Applications, 33(4):632–650, 2019.

[45] RASMUSSEN, A.F.; SANDVE, T. H.; BAO, K.; LAUSER, A., HOVE, J.; SKAFLESTAD, B.; SAEVAREID, O.; LIE, K.A.; THUNE, A. **The open porous media flow reservoir simulator**. Computers and Mathematics with Applications, 81:159–185, 2021.

[46] DOLLEISER, M.; HASHEMI-NEZHAD, S. R. **Dumux: DUNE for multi-{phase, component, scale, physics, ...} flow and transport in porous media**. Advances in Water Resources, 34:1102–1112, 2011.

[47] BYER, T. **Preconditioned Newton methods for simulation of reservoirs with surface facilities**. Phd thesis, Stanford, 2000.

[48] JAURÉ, S; LOUBENS, R. **Reservoir simulation prototyping platform for high performance computing**. In: SPE LARGE SCALE COMPUTING AND BIG DATA CHALLENGES IN RESERVOIR SIMULATION, p. 1–19, Istanbul (Turkey), 2014.

[49] NIKNEJAD, N; ISMAIL, W.; GHANI, RI.; NAZARI, B.; BAHARI, M.; CHE HUSSIN, A. R. **Understanding service-oriented architecture (SOA): A systematic literature review and directions for further investigation**. Information Systems, 91:1–27, 2020.

[50] MUELLER, B.; VIERING, G.; LEGNER, C.; RIEMPP, G. **Understanding the economic potential of service-oriented architecture**. Journal of Management Information Systems, 26(4):145–180, 2010.

[51] DUARTE, L. S. **TopSim: A plugin-based framework for large-scale numerical analysis**. Phd thesis, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brazil, 2016.

PUC-Rio - Certificação Digital Nº 1912763/CA

[52] AMOOIE, M.A.; MOORTGAT, J. **High-order black-oil and compositional modeling of multiphase compressible flow in porous media**. International Journal of Multiphase Flow, 105:45–59, 2018.

[53] PEACEMAN, D. W. **Fundamentals of numerical reservoir simulation**. Elsevier, New York (NY, USA), 6th edition, 1977.

[54] STONE, H. L. **Estimation of three-phase relative permeability and residual oil data**. Journal of Canadian Petroleum Technology, 12(4):53–61, 1973.

[55] EL-BANBI, A.; ALZAHABI, A.; EL-MARAGHI, A. **Chapter 7 - Black Oils**. In: PVT PROPERTY CORRELATIONS, p. 147–182. Gulf Professional Publishing, 2018.

[56] SCHLUMBERGER. **Eclipse technical description**, 2015. [software, v.2015.2].

[57] COMPUTER MODELING GROUP. **IMEX technical manual**, 2020. [software, v.2020.10].

[58] ABOU-KASSEM, J. H.; RAFIQUL ISLAM, M.; FAROUQ ALI, S. M. **Petroleum Reservoir Simulation: the engineering approach**. Gulf Professional Publishing, Cambridge (USA), 2nd edition, 2020.

[59] WANG, X.; TCHELEPI, H. A. **Trust-region based solver for nonlinear transport in heterogeneous porous media**. Journal of Computational Physics, 253(C):114–137, 2013.

[60] LEE, S. H.; EFENDIEV, Y. **C1-Continuous relative permeability and hybrid upwind discretization of three phase flow in porous media**. Advances in Water Resources, 96:209–224, 2016.

[61] LI, B.; TCHELEPI, H. A. **Nonlinear analysis of multiphase transport in porous media in the presence of viscous, buoyancy, and capillary forces**. Journal of Computational Physics, 297(C):104–131, 2015.

[62] LEE, S. H.; EFENDIEV, Y.; TCHELEPI, H. A. **Hybrid upwind discretization of nonlinear two-phase flow with gravity**. Advances in water resources, 82:27–38, 2015.

[63] LEE, S. H.; EFENDIEV, Y. **Hybrid discretization of multi-phase flow in porous media in the presence of viscous, gravitational, and capillary forces**. Computational Geosciences, 22:1403–1421, 2018.

PUC-Rio - Certificação Digital Nº 1912763/CA

[64] JIANG, J.; YOUNIS, R. M. **Efficient C1-continuous phase-potential upwind (C1-PPU) schemes for coupled multiphase flow and transport with gravity**. Advances in Water Resources, 108:184–204, 2017.

[65] PEACEMAN, D. W. **Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability**. SPE Journal, 23(3):531–543, 1983.

[66] SCHIOZER, D. J. **Simultaneous simulation of reservoir and surface facilities**. Phd thesis, Stanford, 1994.

[67] CHEN, Z. **Reservoir Simulation: Mathematical Techniques in Oil Recovery (CBMS-NSF Regional Conference Series in Applied Mathematics)**. Society for Industrial and Applied Mathematics, USA, 2007.

[68] JIANG, Y. **Techniques for modeling complex reservoirs and advanced wells**. Phd thesis, Stanford, 2007.

[69] FORSYTH, P. A.; SAMMON, P. H. **Practical considerations for adaptive implicit methods in reservoir simulation**. Journal of Computational Physics, 62(2):265–281, 1986.

[70] COATS, K. H. **IMPES stability: The CFL limit**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 1–6, Houston (TX, USA), 2001.

[71] COATS, K. H. **IMPES stability: The stable step**. In: SPE RESERVOIR SIMULATION SYMPOSIUM, p. 1–10, Houston (TX, USA), 2001.

[72] FUNG, L. S.; COLLINS, D. A.; NGHLEM, L. **An adaptive-implicit switching criterion based on numerical stability analysis**. SPE Reservoir Engineering, 4(1):45–51, 1989.

[73] AN, H.; MO, Z.; LIU, X. **A choice of forcing terms in inexact newton method**. Journal of Computational and Applied Mathematics, 200(1):47–60, 2007.

[74] WANG, K.; LIU, H.; CHEN, Z. **A scalable parallel black oil simulator on distributed memory parallel computers**. Journal of Computational Physics, 301:19–34, 2015.

[75] SHETH. S.; MONCORGÉ, A. **Inexact newton method for general purpose reservoir simulation**. CoRR, abs/1912.06568, 2019.

PUC-Rio - Certificação Digital Nº 1912763/CA

[76] DEMBO, R. S.; EISENTAT, S. C.; STEIHAUG, T. **Inexact newton methods**. SIAM Journal on Numerical Analysis, 19(2):400–408, 1982.

[77] EISENSTAT, S. C.; WALKER, H. F. **Choosing the forcing terms in an inexact newton method**. SIAM Journal on Scientific Computing, 17(1):16–32, 1996.

[78] CHEN, T.; GEWECKE, N.; LI, Z.; RUBIANO, A. ; SHUTTLEWORTH, R.; YANG, B.; ZHONG, X. **Fast computational methods for reservoir flow models**. University of Minnesota. Institute for Mathematics and Its Applications, 2009.

[79] TODD, M. R.; O'DELL, P. M.; HIRASAKI, G. J. **Methods for increased accuracy in numerical reservoir simulators**. SPE Journal, 12(6):515–530, 1972.

[80] SAMMON, P. H.; RUBIN, B. **Practical control of timestep selection in thermal simulation**. SPE Reservoir Engineering, 1(2):163–170, 1986.

[81] AZIZ, K.; SETTARI, A. **Petroleum Reservoir Simulation**. Applied Science Publishers, London (UK), 1979.

[82] AKAKPO, D.; GILDIN, E. **A control perspective to adaptive time-stepping in reservoir simulation**. In: SPE RESERVOIR SIMULATION CONFERENCE, p. 1–20, Montgomery (TX, USA), 2017.

[83] SÖDERLIND, G. **Digital filters in adaptive time-stepping**. ACM Transactions on Mathematical Software, 29(1):1–26, 2003.

[84] ABOU-KASSEM, J. H.; OSMAN, M. E.; ZAID, A. M. **Architecture of a multipurpose simulator**. Journal of Petroleum Science and Engineering, 16:221–235, 1996.

[85] CELES, W.; PAULINO, G. H.; ESPINHA, R. **A compact adjacency-based topological data structure for finite element mesh representation**. International Journal of Numerical Methods in Engineering, 64(11):1529–1556, 2005.

[86] IERUSALIMSCHY, R.; FIGUEIREDO, L. H.; CELES, W. **Lua 5.1 Reference Manual**. lua.org, 2006.

[87] ENERGISTICS. **RESCUE standards**, 2015.

[88] WILSON, P. R. **Some issues and strategies in heap management and memory hierarchies**. ACM SIGPlan Notices, 26(3):45–52, 1991.

PUC-Rio - Certificação Digital Nº 1912763/CA

[89] WANG, B.; WU, S.; HAN, D.; HUAN, G.; LI, Q.; LI, X.; LI, H.; ZHOU, J. **Block compressed storage and computation in the large-scale reservoir simulation**. Petroleum Exploration and Development, 40(4):495–500, 2013.

[90] SCHENK, O.; GÄRTNER, K. **Solving unsymmetric sparse systems of linear equations with PARDISO**. Future Generation Computer Systems, 20(3):475–487, 2004.

[91] PIAZZA, R. E. **Performance assessment of linear solvers for fully implicit reservoir simulation**. Masters thesis, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brazil, 2019.

[92] ODEH, A. S. **Comparison of solutions to a three-dimensional black-oil reservoir simulation problem**. Journal of Petroleum Technology, 33:13–25, 1981.

[93] KILLOUGH, J. E. **Ninth SPE comparative solution project: A reexamination of black-oil simulation**. In: 13TH SPE SYMPOSIUM ON RESERVOIR SIMULATION, p. 135–147, San Antonio (TX, USA), 1995.

[94] WATTENBARGER, R. A. **Practical aspects of compositional simulation**. In: SPE SYMPOSIUM ON NUMERICAL SIMULATION, p. 1–12, Dallas (TX, USA), 1970.

[95] SÖDERLIND, G. **Automatic control and adaptive time-stepping**. Numerical Algorithms, 31:281–310, 2002.

[96] DEUFLHARD, P. **Newton methods for nonlinear problems: affine invariance and adaptive algorithms**, volume 35. Springer Science & Business Media, 2011.

[97] WANG, X.; TCHELEPI, H. A. **Trust-region based solver for nonlinear transport in heterogeneous porous media**. Journal of Computational Physics, 253:114–137, 2013.

[98] MØYNER, O.. **Nonlinear solver for three-phase transport problems based on approximate trust regions**. Computational Geosciences, 21(5):999–1021, 2017.

PUC-Rio - Certificação Digital Nº 1912763/CA