ISSN: 2581-4311
DOI: doi.org/10.51219/IJCRSET/Wang-Xiaoming/125

URF PUBLISHERS
connect with research world

International Journal of Current Research in Science, Engineering & Technology

https://urfpublishers.com/journal/ijcrset

Vol: 7 & Iss: 1

Research Article

Path to Build Your Own Simulator

Wang Xiaoming*

*Independent Researcher, Chengdu, China

Citation: Xiaoming W. Path to Build Your Own Simulator. *Int J Cur Res Sci Eng Tech* 2023; 7(1), 6-9. DOI: doi.org/10.51219/IJCRSET/Wang-Xiaoming/125

Received: 11 January, 2024; **Accepted:** 24 January, 2024; **Published:** 27 January, 2024

***Corresponding author:** Wang Xiaoming, Independent Researcher, Chengdu, China, Email: wangxiaoming76842@proton.me

ABSTRACT

This paper outlines the journey towards constructing a personalized reservoir simulator. Beginning with an introduction, the narrative navigates through key elements, including the reservoir model, simulation inputs, and outputs. Emphasis is placed on the intricacies of simulation execution and stability, shedding light on the critical aspects of boundary conditions and 3D visualization. The exploration culminates in a comprehensive conclusion, providing insights and reflections on the path to building a customized reservoir simulator. This endeavor encompasses a holistic understanding of reservoir dynamics, simulation intricacies, and the significance of effective visualization in the simulation process.

Keywords: Reservoir simulation; Customized simulator; 3D visualization

1. Introduction

Demystifying reservoir simulation, one of the most challenging hurdles for future petroleum engineers, is the heart of this article. We delve into the fundamental principles and steps involved, but in the interest of clarity and conciseness, some smaller intricacies have been left out. Fear not, this streamlined approach still packs a punch, providing you with the necessary grasp of this powerful tool in petroleum engineering.

In the dynamic realm of research and engineering, where tasks often demand unique solutions, the ability to create specialized tools can be a game-changer. Enter Python and VBA, two potent programming languages that have become trusted allies for authors seeking to forge their own instruments of efficiency and precision.

Python: A versatile favorite among scientists and engineers, Python's renowned readability and extensive libraries have made it a popular choice for crafting tools that tackle data analysis, visualization, numerical computations, and automation tasks. Its intuitive syntax allows for rapid development and seamless integration with other software, making it a powerful asset for streamlining workflows and unlocking new analytical insights.

VBA: While often found within the confines of Microsoft Office applications, VBA (Visual Basic for Applications) holds untapped potential for those seeking to extend the capabilities of these familiar platforms. From automating repetitive Excel tasks to creating custom functions in Word or PowerPoint, VBA empowers authors to tailor their workspaces and streamline document-driven processes, saving valuable time and effort.

Bridging the Gap: Some authors[1-3] have skillfully combined the strengths of both languages, using Python to handle complex computations and data analysis while leveraging VBA to interact with Office applications and automate reporting tasks. This synergistic approach harnesses the best of both worlds, enabling the creation of robust and adaptable tools that seamlessly bridge diverse software environments.

The Impact of Custom Tools: The benefits of crafting custom tools extend beyond personal productivity. By sharing their creations with colleagues or within online communities, authors contribute to knowledge sharing, collaborative problem-solving, and the continuous evolution of research and engineering practices. This collective effort fosters innovation and drives advancements in a wide range of fields, demonstrating the profound impact of empowered authors shaping their digital toolkits to meet their unique needs.

2. Reservoir Model, Simulation Inputs, and Outputs

The main function of a reservoir simulator is to model fluid flow through porous, permeable media, located deep in the subsurface. The spatial domain over which the simulation is performed is divided into a number of interconnected "gridblocks" of geometric dimensions Δx, Δy, and Δz in a classic Cartesian coordinate system (the more the gridblocks, the slower the simulator's speed, but higher the accuracy).

The simplest modeling approach is the finite differences method (FDM), which ultimately converts a highly-complex partial differential equation (PDE) into a system of linear algebraic equations; one for each node ($1 \leq i \leq N$), connecting it to its surrounding nodes. These algebraic equations are combined together into one matrix-vector expression and are solved simultaneously in each time-step. During every simulation time-step, various reservoir properties (**Table 1**) are assigned for each node; some inputs staying constant throughout the entire simulation, while others vary following a prescribed model or correlation, such as the Corey and Brooks (1964)[4] and van Genuchten (1980)[5] models for relative permeability and capillary pressure, respectively.

Table 1: Typical input properties for a reservoir simulator[6].

Reservoir Property	Symbol	Physical Definition	Oilfield Units
Porosity	φ	Pore volume per unit rock	[-]
Absolute permeability	k	Ability to flow fluids through the rock. May be anisotropic (vary with x, y, and z-directions)	mD
Relative permeability	k_r	Adaptation of permeability to multiphase flow (e.g., water and oil)	[-]
Total compressibility	c_t	Volume change with pressure variation (incorporates rock and fluids)	1/psi
Fluid density	ρ	Fluid mass per unit volume	ppg
Fluid viscosity	μ	Resistance to flow	cp
Formation volume factor (FVF)	B	Ratio of fluid volume at reservoir conditions to fluid volume at standard conditions	RB/STB
Capillary pressure	p_c	The pressure difference between the non-wetting (usually oil) and wetting (usually water) phases	psi
Elevation	z	For each gridblock from a reference datum	ft
Well flowrate	q	Positive for injection wells and negative for production wells	STB/day
Bottomhole flowing pressure	p_{wf}	Always positive—for FDM it can be approximated from the pressure in the node(s) the well is occupying	psi

In simulations involving two distinct, non-mixing fluids (water and oil), certain reservoir properties (permeability, relative permeability, viscosity, formation volume factor, and flow rate) exhibit unique values for each phase. To distinguish these properties, we employ subscripts "w" and "o" to represent water-phase and oil-phase values, respectively (e.g., kw, ko, krw, kro, μw, μo, Bw, Bo, qw, and qo). As a result, parameters derived from these properties during the simulation will also vary accordingly for each phase.

After every simulation time-step (lasting a duration of Δt), the model generates outputs encompassing pressure (p), water saturation (Sw), and oil saturation (So) within the defined spatial domain illustrated in (**Figure 1**). These values are stored in corresponding output vectors (p, Sw, and So) that undergo updates after each time-step. Prior to initiating the first simulation time-step, it's crucial to establish the initial conditions (ICs) for these three output vectors.

This image showcases the dynamic interplay of pressure, water saturation (Sw), and oil saturation (So) within the reservoir's depths. Imagine this: A horizontal well, stretching 1000 feet across, diligently taps into the reservoir's bounty, extracting 200 barrels of oil per day for a period of 100 days. Initially, the pressure within this subterranean treasure trove stood at a formidable 4,000 psi, with water claiming 20% (Sw = 0.2) of the pore space and oil reigning supreme over the remaining 80% (So = 0.8). As the simulation unfolds, (**Figure 1**) paints a vivid picture of the evolving story within the reservoir.

The pressure distribution, represented by the color map, reveals the impact of the relentless extraction. We can see how the pressure plummets around the wellbore, creating a cone-shaped depression that gradually expands with time.

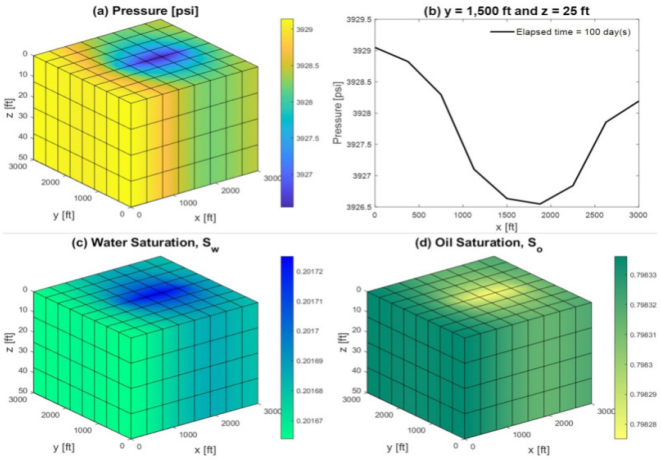

Figure 1: The essence of a simulated two-phase reservoir, courtesy of "Program for Integrated Modeling of Petroleum Systems," a 3D masterpiece crafted by the author in MATLAB[7].

The water saturation plot (blue tones) portrays the movement of water as it displaces oil (yellow tones) within the reservoir. This intricate dance between the two fluids is governed by complex physical principles encoded within the simulation program. By studying these visuals, we gain valuable insights into the behavior of the reservoir under production. We can assess the effectiveness of the well placement, predict future production trends, and optimize recovery strategies. (**Figure 1**), therefore, serves as a powerful window into the hidden world of subsurface resources, offering invaluable knowledge for reservoir engineers and geoscientists alike.

3. Simulation Execution and Stability

The most straightforward approach for executing a reservoir simulator is IMPES (IMplicit Pressure, Explicit Saturation). This method minimizes computational workload and simplifies implementation by generating smaller systems of equations, with one for each node ($1 \leq i \leq N$). (**Table 2**) presents the matrices and vectors utilized in an IMPES scheme, along with the resulting pressure and saturation vectors.

Table 2: Matrices and vectors[7].

Matrix/vector name	Symbol	i^{th} element*	Two-phase extension	Dimension
Transmissibility matrix**	$\boldsymbol{T}$	$\left(\frac{k_{rw}k_w\Delta y\Delta z}{\mu_w B_w \Delta x}\right)_i$	$\boldsymbol{T} = \boldsymbol{T_w} + \frac{B_o}{B_w}\boldsymbol{T_o}$	$N \times N$; tridiagonal (for 1D), pentadiagonal (for 2D), or heptadiagonal (for 3D)
Accumulation rate matrix	$\boldsymbol{D}$	$\frac{1}{\Delta t}\left(\frac{\varphi\Delta x\Delta y\Delta z}{B_w}\right)_i$	$\boldsymbol{D} = \boldsymbol{D_w} + \frac{B_o}{B_w}\boldsymbol{D_o}$	$N \times N$; monodiagonal
Productivity index (PI) matrix	$\boldsymbol{J}$	$\frac{(q_w)_i}{(p)_i - p_{wf}}$	$\boldsymbol{J} = \boldsymbol{J_w} + \frac{B_o}{B_w}\boldsymbol{J_o}$	$N \times N$; monodiagonal
Source vector (from wells and boundary conditions)	$\boldsymbol{Q}$	$(J)_i p_{wf} + (q_w)_i$	$\boldsymbol{Q} = \boldsymbol{Q_w} + \frac{B_o}{B_w}\boldsymbol{Q_o}$	$N \times 1$
Gravity and capillary pressure vector	$\boldsymbol{G}$	Use volumetric average between water and oil densities	$\boldsymbol{G} = \rho g \boldsymbol{T}\boldsymbol{z} + \boldsymbol{T}\boldsymbol{p_c}$	$N \times 1$

* $1 \leq i \leq N$; for simplicity, only the elementary definition for water-phase along the x-direction is displayed.
** Conversion factor from inputs in oilfield units: 0.00633 $cp \cdot ft^2/(psi \cdot mD \cdot day)$.

Within the time-loop (**Figure 2**) the p vector-containing the pressure value at each node-is first evaluated for the new time-step, n+1, using its own values from the previous time-step, n:

$$\begin{bmatrix} p_1^{n+1} \\ \vdots \\ p_N^{n+1} \end{bmatrix} = \left(\begin{bmatrix} T_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & T_N \end{bmatrix} + \begin{bmatrix} D_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & D_N \end{bmatrix} + \begin{bmatrix} J_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & J_N \end{bmatrix} \right)^{-1} \left(\begin{bmatrix} D_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & D_N \end{bmatrix} \begin{bmatrix} p_1^n \\ \vdots \\ p_N^n \end{bmatrix} + \begin{bmatrix} Q_1 \\ \vdots \\ Q_N \end{bmatrix} + \begin{bmatrix} G_1 \\ \vdots \\ G_N \end{bmatrix} \right) \quad (1)$$

Next, the Sw vector is evaluated also for time-step, n+1, using the newly-calculated P vector from Eq. (1) by:

$$\begin{bmatrix} S_{w,1}^{n+1} \\ \vdots \\ S_{w,N}^{n+1} \end{bmatrix} \cong \begin{bmatrix} S_{w,1}^{n} \\ \vdots \\ S_{w,N}^{n} \end{bmatrix} + \begin{bmatrix} D_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & D_N \end{bmatrix}^{-1} \left(\begin{bmatrix} Q_{w,1} \\ \vdots \\ Q_{w,N} \end{bmatrix} + \begin{bmatrix} G_1 \\ \vdots \\ G_N \end{bmatrix} - \begin{bmatrix} T_{w,1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & T_{w,N} \end{bmatrix} \begin{bmatrix} p_1^{n+1} \\ \vdots \\ p_N^{n+1} \end{bmatrix} \right) \quad (2)$$

Subsequently, using material balance, the So vector is obtained by:

$$\begin{bmatrix} S_{o,1}^{n+1} \\ \vdots \\ S_{o,N}^{n+1} \end{bmatrix} = 1 - \begin{bmatrix} S_{w,1}^{n+1} \\ \vdots \\ S_{w,N}^{n+1} \end{bmatrix} \quad (3)$$

Eqs. (1) and (2) are in the form "Ax=b," solvable by various numerical methods, such as lower-upper (LU) decomposition. MATLAB's backslash function is a powerful tool for efficiently solving such systems (i.e. via typing "x=A\b").

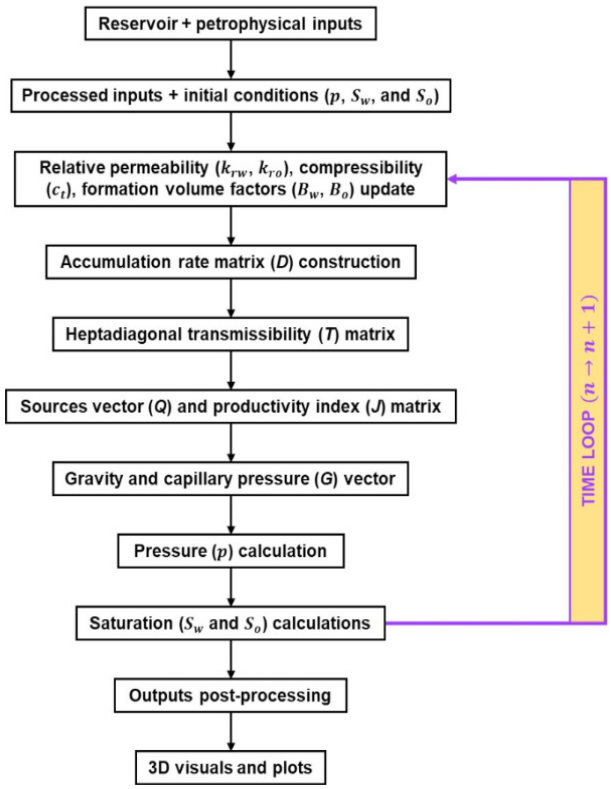

Figure 2: Flowchart depicting the fundamental processes within a 3D, two-phase reservoir simulator[7].

While the IMPES scheme offers advantages, its stability is constrained by a time-step limitation due to its explicit component. To ensure stable simulations, the Courant-Friedrichs-Lewy (CFL) criterion must be met under the specific conditions[8]. The simulator can be readily configured to automatically employ the largest permissible time step (Δt) throughout the simulation process.

4. Boundary Conditions and 3D Visualization

Beyond initial conditions (ICs), reservoir simulations also rely on boundary conditions (BCs) to define how the system behaves at its edges. The two most common types of BCs are:

Constant flow rate (Neumann): This BC specifies a fixed flow rate of fluid entering or exiting the reservoir at the boundary. Think of it like a constantly pouring well or a drain with a specific flow rate.

Constant pressure (Dirichlet): This BC maintains a constant pressure at the boundary, regardless of the flow rate. Imagine a large, ever-replenishing reservoir maintaining a steady pressure against the edge of your simulated system. To visualize the behavior of a vertical well producing from a square reservoir with a constant pressure boundary, we can use a trick. Instead of directly simulating the square reservoir with one Dirichlet boundary, we can consider a closed rectangular reservoir with no-flow (Neumann) boundaries on all sides (**Figure 3**). This essentially simulates two identical square reservoirs joined together at the center. A well in the center of the left reservoir continuously produces at a constant flow rate, while another well injects at the same rate in the center of the right reservoir. This injection creates a constant pressure boundary in the middle of the combined reservoir (at x = 2,000 ft).

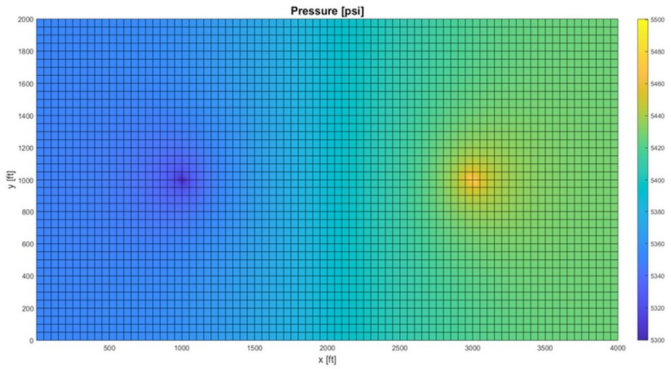

Figure 3: Top view of a model.

(**Figure 3**) presents a top view of a model featuring a well extracting at a consistent flow rate from the center of a square reservoir. The reservoir is enclosed by "Neumann" boundaries with no flow at the top, bottom, and left sides, while the right side has a constant pressure "Dirichlet" boundary. Simulating the latter involves appending a second, "imaginary" reservoir to the right side of the actual reservoir, complete with a well at its center injecting at the identical flow rate as the production well.

To emulate Neumann conditions, one can simulate them by defining a flowrate along a reservoir edge. Similarly, implementing no-flow conditions can replicate the effect of the commencement of an impermeable zone, such as a shale acting as a seal. To approximate a Dirichlet boundary, it is optimal to introduce "imaginary" gridblocks beyond the reservoir edge with designated pressures. This ensures that the arithmetic average between an imaginary grid block and its adjacent (real) grid block equals the desired constant pressure.

Incorporating Neumann or Dirichlet boundary conditions necessitates adjustments to the elements within the vector Q in Eqs. (1) and (2) corresponding to the nodes along the specific boundary. For Dirichlet boundary conditions, additional modifications involve altering the diagonal elements within the matrix T that correspond to the nodes along the boundary.

5. Conclusion

In conclusion, this paper has charted a comprehensive path toward constructing a personalized reservoir simulator. The journey began with an introduction to the essential components, followed by a detailed exploration of the reservoir model, simulation inputs, and outputs. The nuances of simulation execution and stability were scrutinized, highlighting key considerations in achieving robust and reliable results.

The discussion then delved into the critical aspects of boundary conditions, elucidating their impact on simulation accuracy. Additionally, the incorporation of 3D visualization was emphasized as a crucial tool for enhancing the understanding of reservoir dynamics. As we reflect on this endeavor, it becomes

evident that building a customized simulator demands a holistic approach that combines theoretical insights with practical implementation. The interplay between reservoir complexities, boundary conditions, and visualization tools underscores the multifaceted nature of reservoir simulation.

This journey not only contributes to the evolving field of reservoir engineering but also underscores the importance of tailoring simulation tools to specific contexts. As advancements continue, the insights gained from this exploration will serve as a valuable foundation for future developments in reservoir simulation methodologies. Ultimately, the pursuit of constructing a personalized reservoir simulator is a dynamic process, continuously shaped by technological innovations, theoretical advancements, and a deepening understanding of subsurface reservoir behavior.

6. References

1. Alagoz E, Mengen AE, Bensenouci F, Dundar EC. Development of a Computational Tool for Wellbore Stability Analysis and Mud Weight Optimization. 21th Inter Petro Natur Gas 2023;27-29.
2. Alagoz E. Development and Analysis of a Program for Phase-Equilibrium CalcuFlations Using the Peng-Robinson Equation of State. Interl J Earth Sci Knowled App 2023;5(1):51-61.
3. Alagoz E, Ergul MS. Surface Diffusion in Nanopores and Its Effects on Total Mass Transport in Shale Gas Reservoirs, Inter J Ener Envi Sci 2023;8(4):73-78.
4. Brooks R, Corey T. Hydraulic Properties of Porous Media. 1994;24-37.
5. Van Genuchten. A Closed-Form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. Soil Sci Soci of Amer J, 1980;44(5): 892-898.
6. Michael A. TWA Managing Editor's Column: How to Build Your Own Reservoir Simulator. 2021
7. Michael A. PIMPS3D2P: A Practical, Efficient, Three-Dimensional, Two-Phase Reservoir Simulator Written in MATLAB. Europ Associ of Geosci Eng. 2021;1-5
8. Coats KH. IMPES stability: the CFL limit. SPE J. 2023;8(3):291-297.