A Semi-Analytical Streamline Simulation in Near-Wellbore Regions and its Applications under Constant Pressure Boundaries

by

©Xiaoyan Tang

A proposal submitted to the
School of Graduate Studies
in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2017

St. John's Newfoundland

Abstract

Streamline simulation is a powerful tool that can be used for full field forecasting, history matching, flood optimization, and displacement visualization. This research thesis presents the extension of a semi-analytical streamline simulation method and its application in the near-wellbore region in two-dimensional polar coordinate systems and three-dimensional cylindrical coordinate systems. The main objective of this research thesis is to study the effects of the permeability heterogeneity and well completion details in the near-wellbore region. These effects dictate the streamline geometries, which in turn influence well productivity. It is revealed that the semi-analytical streamline simulation method developed in this research thesis is the only known streamline method with sufficient accuracy for streamline simulation in polar/cylindrical geometries.

Previous streamline applications used a constant flow rate condition for each stream tube. However, wells in low permeability reservoirs are often produced at constant pressure. In this research thesis, streamline simulation is performed under constant pressure boundaries. This is a novel and non-trivial extension of streamline simulation.

The semi-anlytical streamline method is applied in the perforated wells. Results indicate that it is the only method that can accurately simulate the streamline path

in such wells. A new skin calculation method based on the semi-anlytical streamline simulation method is introduced and applied in perforated wells. This new skin calculation method is believed to be superior and can be used to examine the effect of the perforation parameters. It provides useful information for evaluating the well completion strategy.

In this work, the two-phase displacement process is simulated along stream tubes. Solutions are constructed by treating each stream tube as a flow unit by invoking novel analytical solutions for such geometries. Visualization experiments are direct ways to investigate the effect of the heterogeneity on flow distribution. Two-dimensional radial waterflooding visualization experiments are performed under constant pressure boundaries for homogeneous and heterogeneous porous media. The homogenous case is used to history match and determine the relative permeabilities. Using these relative permeabilities, the semi-analytical streamline simulation method is independently validated against the results from the heterogeneous visualization experiments.

Acknowledgments

I would like to express my sincere thanks to my dearest supervisors Dr. Lesley James and Dr. Thormod Johansen for their support, encouragement, and guidance throughout this study. You have always been deeply involved with my study and spent tremendous time on it. You enlightened and guided me through each step toward the accomplishment of this research thesis. Furthermore, I also would like to thank my colleagues in the research group and technical assistants in the Hibernia Enhanced Oil Recovery Lab who provided help and constructive feedback in the experiments of this research thesis.

I also would like to express my greatest gratitude to the support of the Hibernia Management and Development Company Ltd. (HMDC), Chevron Canada, the Research and Development Corporation (RDC), the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Canadian Foundation for Innovation for funding my research.

Finally, to my dear parents and husband, I would like to thank you for all your love and support throughout my life. Without your support I could not complete my study far abroad. Thank you to my brother and friends, you have also been the best of support along this study.

List of Nomenclature

Symbol	Description	Units
A	Area of cross section	m^2
e	Relative error	Dimensionless
f	Water fractional flow	Dimensionless
h	Height	m
K	Permeability	m^2
L	Length of the porous media	m
M	Mobility ratio	Dimensionless
M	Mass	kg
p	Pressure	Pa
q	Volumetric flow rate	m^3/s
r	Radius	m
S	Skin	Dimensionless
S	Saturation	Dimensionless
s	Saturation	Dimensionless
t	Time	s
u	Darcy velocity	m/s
v	Real velocity	m/s
V	Volume	m^3

Symbol	Description	Units
X	Distance in x-direction	m
Y	Distance in y-direction	m
z	Height	m

Greek	Description	Units
α	Angle	$^\circ$
λ	Mobility	$m^2/(Pa \cdot s)$
ϕ	Porosity	Dimensionless
μ	Viscosity	$Pa \cdot s$
θ	Angle	$^\circ$
ρ	Density of water	kg/m^3
τ	Time-of-flight	s

Subscripts	
BT	Breakthrough
D	Dimensionless
e	External
en	Entry point of the streamline
ex	Exit point of the streamline
i	Grid block in radial direction
j	Grid block in θ-direction
k	Grid block in z-direction
l	Lower integration limit
N	Layer number
or	Residual oil

Subscripts

r	Radial direction
s	After breakthrough
u	Upper integration limit
w	Wellbore/Water
wc	Connate water
z	z-direction
θ	Angular direction, counter-clockwise

Superscripts

$*$	Shock wave position
r	Radial direction
θ	Angular direction, counter-clockwise

Coefficients

a, b, c, d, e, f, g, h

A, B, C, D, E

Abbreviations

1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional
TOF	Time-of-flight
log-lin	Logarithmic-linear
bilin-log	Bi-linear logarithmic

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

Reservoir simulation is an essential tool for reservoir management. Reservoir simulation combines mathematical models, geological models, numerical, and computer programs in order to simulate the behavior of fluids within the reservoir over time. The finite-difference numerical method is the most commonly used numerical method in commercial reservoir simulators. In the finite-difference method, the reservoir is divided into smaller blocks. The inter-block fluxes are then calculated. The finite-difference method is efficient; however, with the incorporation of geological complexity and reservoir characterization, streamline and stream tube simulation have been proven more efficient (Thiele et al., 2010). In the near-wellbore region, heterogeneities always exist because of drilling or well completion effects. This research thesis applies a semi-analytical streamline method in the near-wellbore region to predict the flow performance.

1.1 Streamline and Stream Tube Modeling

Streamline simulation is a powerful tool that is used for reservoir management, history matching, and displacement visualization. It is especially effective in solving fluid flow problems in geologically complex and heterogeneous systems. In waterflooding, streamline simulation is an efficient method to understand the flow process of water and oil between injection wells and production wells. The distinguishing feature of streamline simulation is that fluid flow is decoupled from cell-to-cell interactions, as in conventional finite-difference methods, into one-dimensional (1D) problems along streamlines. No fluid can flow across streamlines, hence, it is not suitable for displacing processes involving cross-streamline transport mechanisms such as capillary pressure, transverse dispersion, and compressibility (Datta-Gupta and King, 1995). Table 1.1 lists the advantages and disadvantages for streamline simulation.

Table 1.1: Advantages and Disadvantages for Streamline Simulation

Streamline Simulation	**Feature**
Advantages	• Powerful in visualization of flow patterns. • Effective in geologically complex and heterogeneous systems. • Good at tracking water fronts in waterflooding. • Efficient than conventional finite-difference methods because it decoupled 3D problems into 1D problems along streamlines.
Disadvantages	• Streamlines are not update at every time step. • Streamline model ignores gravity and diffusive effects.

Streamlines are "instantaneous lines that are everywhere tangential to the velocity field" (Thiele, 2001). They describe the fluid movement through time in the reservoir. The density of the streamlines indicates the magnitude of the local flow velocity. Dense distribution indicates fast flow and sparse distribution represents slow flow. Time-of-flight (TOF) is a key concept in streamline simulation. It is the travel time of a neutral particle along a streamline. This concept was first introduced by Pollock (1988) and first used as a spatial variable by Datta-Gupta and King (1995). The TOF provides quantitative information on the connectivity between the injector and the producer (Datta-Gupta and King, 2007).

A most important feature in a streamline model is that streamlines constitute a space filling, non-intersecting family of curves. Streamlines are calculated throughout the reservoir from the instantaneous velocity field. In the uniqueness context of the Laplace equation, for any location only one velocity vector can exist. This means flow can only go in one direction at one location. Streamlines can never cross, and they show the direction in which a massless fluid element will travel at any point in time.

Streamline simulation involves the following major steps (Datta-Gupta and King, 2007):

1. Generate pressure and velocity field by numerically solving the pressure equation under given boundary conditions and applying Darcy's law;

2. Trace streamlines based on the local total fluid velocity;

3. Calculate time-of-flight along the streamlines;

4. Solve the transportation equation along the streamlines for an appropriate pe-

riod of time when it is adequate to assume fixed streamlines;

5. Update streamlines as needed based on mobility effects and changing well conditions;

6. Calculate saturation distribution based on the updated streamlines.

A stream tube is a tubular region in space bounded by a surface of streamlines as illustrated in Figure 1.1. In two dimensions, a stream tube is a region bounded by two streamlines. Since velocity is tangential to the streamlines, no convective flux can cross the boundaries of a stream tube. This shows that the calculation along a stream tube is completely decoupled from other stream tubes.

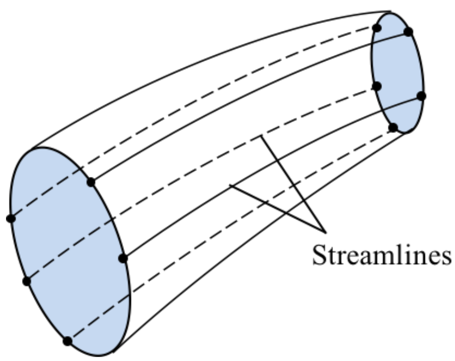


Figure 1.1: Schematic of a Stream Tube in 3D

Previous streamline/stream tube applications have been performed under constant flow rate conditions only. However, in a real field, reservoir production may be operated under constant injection pressure and constant flowing bottom hole pressure. Wells in low permeability reservoirs are often, by necessity, produced at constant pressure. Specifically, for a production well, pressure is often kept constant above the bubble point pressure. In this research thesis, streamline simulation is performed under the assumption of constant pressure boundaries, which is a novel and highly non-trivial extension of streamline simulation.

1.2 Near-wellbore Streamline Modeling

Usually, the reservoir can be modeled at two distinct scales; near-wellbore modeling and full field modeling. Streamline simulation has been mostly used in Cartesian coordinate systems. Through the research presented in this research thesis, it is also rigorously extended to cylindrical coordinate systems. Cartesian coordinate modeling is often applied to full field modeling, while cylindrical coordinate modeling is utilized for near-wellbore modeling. A streamline illustration of a full field five-spot water-flooding pattern is provided in Figure 1.2. Figure 1.3 shows the relationship between radial grids and Cartesian grids. As illustrated, radial grids are based on a much higher resolution grid in a limited region surrounding the wells. It provides more detailed information near the well, hence it has the ability to model how heterogeneities affect the flow pattern in the near-wellbore region. In this research thesis, streamline simulation is applied in the near-wellbore region, and it is performed in polar coordinate systems for two-dimensional (2D) problems and cylindrical coordinate systems for three-dimensional (3D) problems. The methodology used in this research thesis is demonstrated to be superior to previously reported methodologies for the near-well streamline simulation.

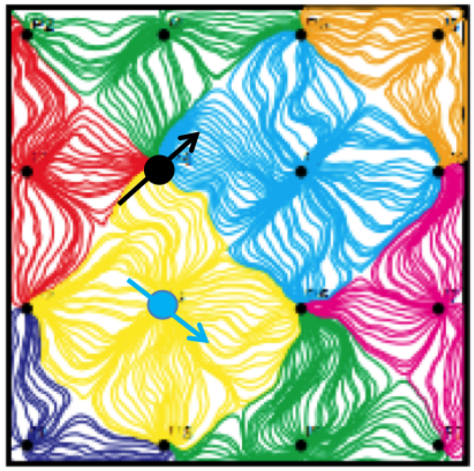


Figure 1.2: Full Field Five-Spot Waterflooding Streamline Model (Thiele, 2001)

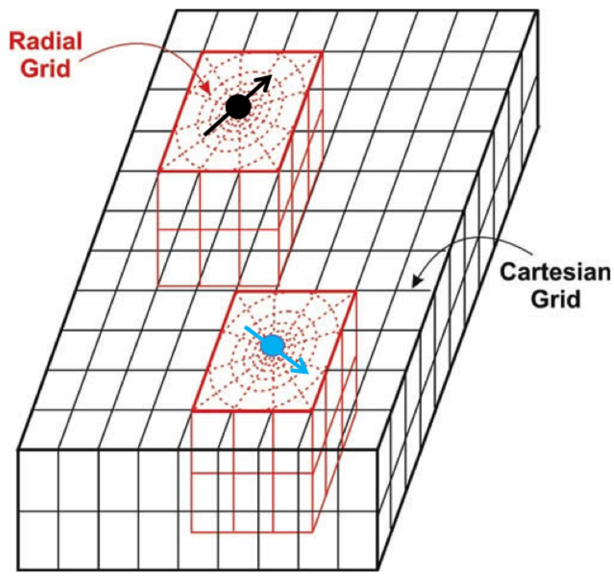


Figure 1.3: Embedding Radial Flow Numerical Model within a Cartesian Grid (British Geological Survey, 2013)

1.3 Solution of Riemann Problems in Stream Tubes

Buckley-Leverett (1942) described the one-dimensional (1D) Riemann problem for constant flow rate for a two-immiscible-phase displacement process. The classical Buckley-Leverett theory does not apply under the constant pressure boundary conditions. The associated 1D Riemann problem for constant pressure boundaries was solved in Johansen and James (2015) for multi-component systems and described in Johansen et al. (2016) for waterflooding. However, Riemann solutions for the classical Buckley-Leverett theory and Johansen et al. (2016) apply only to 1D problems. Specifically, both of these references describe displacement problems for constant cross section area porous media only (1D). For the problems considered in this research thesis, the stream tube cross section area changes along the stream tube arc length. For constant flow rate boundary conditions, the flow velocity can be obtained by dividing the flow rate by the cross section area; however, the 1D Riemann solution in Johansen and James (2015) and Johansen et al. (2016) cannot be used in stream tubes with changing cross-sectional area. Stream tubes with constant pressure boundaries require a three-dimensional (3D) Riemann solution. Johansen and Liu (2016) presented the solution of the Riemann problem in three-dimensional (3D) porous media. Applying this solution, the location of the displacement front and the flow rate at any given time, the time for water breakthrough at the outlet can be obtained. By connecting the fronts in neighboring stream tubes, the global front for the entire near-wellbore region is obtained. The flow rate at the well is the summation of flow rates over all stream tubes.

1.4 Purpose of Research

The objective of this research thesis is to explore and understand streamline simulation in the near-wellbore region.

The first goal of this research thesis is to apply a semi-analytical streamline simulation method to homogeneous and heterogeneous media in a near-wellbore region in two dimensions and three dimensions. Johansen (2010) proposed a new semi-analytical streamline simulation method that can be applied to the near-wellbore region. In his method, the pressure distribution polynomial function was assumed to a logarithmic-linear (log-lin) form in 2D and bi-linear logarithmic (bilin-log) in 3D within each grid block. Depending on the pressure function, streamlines within each grid block can be expressed as smooth curves instead of straight line segments. This method is applied in this research and it is important because streamline simulation in the near-wellbore region is a relatively new area. Little literature can be found in the near-wellbore streamline simulation. Skinner (2011), Skinner and Johansen (2012), and Hadibeik (2011) reported the streamline simulation in the near-wellbore region. Through this research thesis, a rigorous understanding of the principles of streamline tracing and transport problem solving along streamlines in the near-wellbore region is presented.

The second goal of this research thesis is to demonstrate the application of the streamline simulation method in the near-wellbore region. The transport problem for two-phase, immiscible displacement process is solved along stream tubes under constant pressure boundaries. Applying the Riemann solution along a stream tube, front location and flow rate can be solved. They are crucial information for the understanding of the displacement process.

The third important goal of this research thesis is the application of the streamline simulation model in well skin calculation. A new total mechanical skin calculation method for a perforated well based on the semi-analytical streamline simulation method is introduced in this research thesis. Skin is an important parameter for production predictions. It helps us to understand how the near-wellbore damages affect the skin factor and flow. It is a new area for application of the streamline simulation, and the methodology is demonstrated to be superior to existing methods.

The fourth goal of this research thesis is to use visualization experiments to demonstrate that the near-wellbore streamline simulator can be applied to history matching the displacement process in the radial geometry. In addition to streamline simulation, visualization experiments provide a direct way to investigate the effect of heterogeneity on fluid distributions in the near well region. Streamline methods have advantages for history matching since they allow us to visualize the sensitivity of the production response to reservoir model parameters such as permeability (Datta-Gupta and King 2007). In this research thesis, a series of waterflooding visualization experiments are performed at constant pressure boundaries using glass-bead macro-models. Homogeneous and heterogeneous radial macro-models are designed (James 2012) and fabricated to study the waterflooding mechanisms. Flow behavior in the heterogeneous near-wellbore region can be understood by laboratory visual models when matched with simulations. The displacement front is captured by the camera and corresponding parameters such as flow rate and breakthrough time are recorded. Simultaneously, the streamline simulator is applied to model the actual displacement processes. The simulated location of the water front at a specific time, the water breakthrough time, and flow rates are obtained. The accuracy of the present streamline simulator and its superiority over other near-well streamline models are demonstrated through comparisons between simulated and experimental results.

1.5 Thesis Structure

This research thesis is organized in six chapters. **Chapter 1** includes the general background for streamline simulation and the near-wellbore streamline simulation. It also outlines the purpose for this research thesis.

Chapter 2 provides some relevant literatures on streamlines, stream tube simulation, front tracking method, and some relevant researches on visualization experiments.

Chapter 3 discusses the basic information for developing the streamline model in the near-wellbore region. It also provides the methodologies for generating streamlines using the semi-analytical streamline simulation method in 2D polar coordinate systems and 3D cylindrical coordinate systems. Pressure distribution within grid block is assumed to be logarithmic-linear (log-lin) in 2D polar coordinate systems and bi-linear logarithmic (bilin-log) in 3D cylindrical coordinate systems. The determination of grid block corner pressures, which are used in the semi-analytical streamline simulation method, is also presented in this chapter. Finally, it describes the novel streamline tracing procedure.

Chapter 4 demonstrates some applications of the semi-analytical streamline model. First, it shows how to solve the two-phase immiscible problem by mapping the 3D Riemann solution along stream tubes. Then, a skin calculation method based on streamline simulation method for perforated wells is introduced. A series of case studies are also described and discussed.

Chapter 5 describes the set-up and the process for the two-dimensional (2D) waterflooding visualization experiments. The experimental procedures and properties characterization are also included. Finally, the semi-analytical streamline simulation method is used to history match the relative permeabilities of the experiments performed.

Chapter 6 gives a brief summary of the conclusion from this research thesis. It also provides the recommendations for future work.

Chapter 2

Literature Review

2.1 Introduction

Streamline and stream tube simulation have been used in the oil industry for decades. They are used to model flow in porous media for multiple purposes in both petroleum and ground water literature. This chapter reviews some published literature related to streamlines/stream tube modeling. It also includes a brief review of front tracking methods, which are similar to the streamline method presented in this work. Finally, a section reviewing visualization experiments is also included since this research runs waterfront visualization experiments which are used to compare to the simulation results. In reviewing the relevant literature, chronological order is used to highlight original contributions. In two dimensions, a stream tube is a region bounded by a pair of streamlines. For obvious reasons, there is some overlap among the streamlines and stream tube modeling.

2.2 Streamline and Stream Tube Simulation

2.2.1 Streamline and Stream Tube Simulation Origins

Table 2.1 summarizes the majority of streamline and stream tube simulation methods reported before 1980. These models apply to only simple displacement mechanism in two dimensions.

Table 2.1: Streamline and Stream Tube Simulation Methods before 1980

Author (Year)	**Details**
Lagrange (1781)	Introduced the two-dimensional stream function.
Muskat and Wyckoff (1934)	Applied the analytical streamline methods to model fluid flow.
Muskat (1937)	Used streamline modeling to model incompressible fluid flow through a two-dimensional porous media.
Fay and Pratts (1951)	Applied stream tube modeling in petroleum reservoir simulation.
Pitts and Crawford (1970)	Studied permeability heterogeneity using the stream tube modeling process.
LeBlanc and Caudle (1971)	Introduced a stream tube model with variable mobility ratios.
Martin and Wegner (1979)	Extended the fixed stream tube method by updating stream tube paths.

Streamline modeling was first introduced in Lagrange (1781). Lagrange introduced the two-dimensional stream function which defines the streamlines. As with many subjects in reservoir engineering, streamline and stream tube studies date back to the work of Muskat. Muskat and Wyckoff (1934) applied the analytical streamline methods to model fluid flow. Their theory was based on the solution of Laplace line-source and -sink equation and the superposition principle. They concluded that high permeability communication is more important than the well location. This indirectly shows the effect of the reservoir heterogeneity on fluid recovery.

Muskat (1937) used streamline modeling to model incompressible fluid flow through a two-dimensional porous media. The analytical solutions for the stream function and the potential function for the two-dimensional displacement flow problem were derived. Subsequently, the application of stream tube modeling was extended to petroleum fluid modeling.

Fay and Pratts (1951) first applied stream tube modeling to petroleum reservoir simulation. A numerical model was developed for single-phase flow to determine streamlines. A two-phase flow model in stream tubes was then used in a two-well homogeneous two-dimensional system. The model was used to predict the breakthrough time of the injected fluid. Their two-phase results contained some inaccuracies because no simple numerical method was identified capable of tracking the streamlines in their model.

Pitts and Crawford (1970) first studied permeability heterogeneity in the stream tube modeling process. In their model, heterogeneous porous media and homogeneous porous media were simulated in two dimensions. Their model shows that the fluid preferentially flows through the high permeability zone. Although this method captured the permeability effect, gravity, and capillary pressure effects were not considered.

LeBlanc and Caudle (1971) introduced a stream tube model that was able to simulate two-phase flow in a two-dimensional heterogeneous reservoir with variable mobility ratios. The flow rate was integrated along each streamline in this model to capture the variation in total velocity and was suitable for secondary oil recovery prediction. This stream tube model reduced the computational time substantially compared to conventional methods and was therefore very valuable in waterflooding of an oil reservoir.

Martin et al. (1973) extended the fixed stream tube method by updating stream tubes at given times. The new method worked well for $M < 1$ and $M > 100$, M being mobility ratio (Batycky, 1997). However, recalculating streamline paths introduces non-uniform initial conditions along new streamlines. Martin and Wegner (1979) extended their previous method to multi-well, two-phase systems to overcome this problem. A fixed steam tube numerical method was established for the changing mobility field and mapped the original saturation to the new stream tube locations. Then, the local saturation velocity and the total flow rate were used to calculate the saturation movements. Martin and Wegner showed that the largest error occurs for the isolated inverted five-spot pattern for favorable mobility ratios and that the stream tube model would help to reduce numerical diffusion.

The streamline/stream tube methods discussed so far are more computationally efficient than conventional finite-difference simulations. However, these methods are only applicable to 2D problems for simple displacement mechanisms. Next, the hybrid approaches used in streamline simulation and some more advanced streamline simulation methods will be reviewed.

2.2.2 Hybrid Approach to Three-Dimensional Streamline Modeling

Table 2.2 lists the summary of the hybrid approaches to three-dimensional streamline method developed in the 1980s.

Table 2.2: Summary of Hybrid Approaches to 3D Streamline Modeling

Author (Year)	**Details**
Lake et al. (1981)	Applied streamlines in three dimensionsdimensions.
Emanuel et al. (1989)	Used a hybrid finite-difference/stream tube model to CO_2 injection and waterflooding.
Mathews et al. (1989)	Applied hybrid finite-difference/stream tube method to a miscible water-alternating-gas injection.
Tang et al. (1989)	Applied hybrid finite-difference/stream tube method to capture the transition from radial flow near the wells to linear flow away from the wells.

Lake et al. (1981) first attempted to apply streamlines in three dimensions. They combined an areal stream tube model with a cross-sectional finite difference simulator to simulate a 3D reservoir under large-scale polymer flooding. In their model, they assumed that the areal flow was dominated by the well placement. The vertical flow was dominated by the geology and the displacement fluid type. In the vertical direction, a finite difference solution was applied to get the average upscaled 1D solution, which was used as the average solution in stream tubes to solve the 3D problem. This hybrid approach was a more efficient approach than using the two independent modeling methods (areal stream tube method and cross-sectional finite difference method).

Emanuel et al. (1989) applied a hybrid finite-difference/stream tube approach similar to Lake et al. (1981) to CO_2 injection projects and a mature waterflooding case. A finite-difference simulator was used to model the displacement efficiency and vertical sweep, and the stream tube model was used for the areal performance. Fractals were used to describe reservoir heterogeneity in this model. Their results agreed with field data and required less simulation time.

Mathews et al. (1989) applied a hybrid finite-difference/stream tube method to a miscible water-alternating-gas injection. In their model fractals were used to describe the reservoir heterogeneity. They compared the accuracy with the conventional finite difference method. Results indicated that a hybrid finite-difference method/stream tube method could enable the efficient use of effort and computational time.

Tang et al. (1989) also utilized the hybrid approach to a waterflooding case and a CO_2 flooding case. To capture the transition from radial flow near the wells to linear flow away from the wells, they first determined an average cross-sectional response function, then varied the width of the cross section in the finite difference simulation. Furthermore, Tang et al. generated ten different fractional flow curves to account for varying CO_2 slug sizes, which results from updating the flow rates for each stream tube as the flood progresses (Thiele, 1994).

The next section describes the Pollock's method, which represents a milestone in streamline modeling.

2.2.3 Pollock's Streamline Method

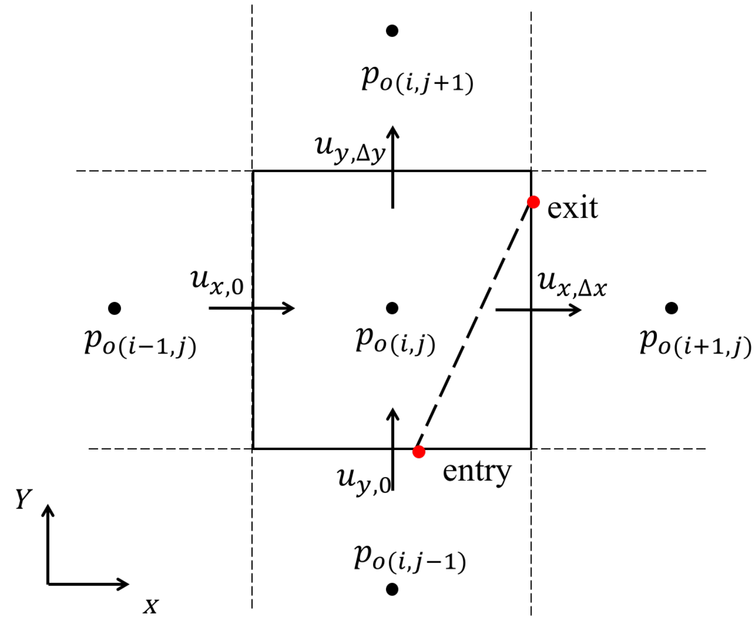


Figure 2.1: Schematic of a Streamline through a Square Grid Block in 2D

Pollock (1988) improved the three-dimensional streamline method by defining a piecewise linear interpolation of the velocity field within a grid block. His approach was a major breakthrough in streamline modeling since it was applicable directly in three-dimensional cases. Figure 2.1 demonstrates a 2D streamline in a square grid block. Pollock's method is a semi-analytical particle tracking method with velocities generated from a block centered finite-difference pressure solution. The points (i, j), $(i, j-1)$, $(i, j+1)$, $(i-1, j)$, and $(i+1, j)$ are the pressure nodes calculated from the finite-difference method. The next step in Pollock's method is to calculate velocities across the block boundaries $u_{x,0}$, $u_{x,\Delta x}$, $u_{y,0}$ and $u_{y,\Delta y}$ using Darcy's law. At the end of this step, velocities across the block boundaries are known. It is then assumed that each directional velocity component varies linearly in the component direction within each grid block as shown in Equation 2.1 to 2.3 for 3 space dimensions x, y, z.

$$u_x = u_{x,0} + a_x(x - x_0), a_x = \frac{u_{x,\Delta x} - u_{x,0}}{\Delta x}, \tag{2.1}$$

$$u_y = u_{y,0} + a_y(y - y_0), a_y = \frac{u_{y,\Delta y} - u_{y,0}}{\Delta y}, \tag{2.2}$$

$$u_z = u_{z,0} + a_z(z - z_0), a_z = \frac{u_{z,\Delta z} - u_{z,0}}{\Delta z}, \tag{2.3}$$

where u_x, u_y and u_z are the velocities in $x-, y-$and $z-$ direction respectively, a_x, a_y, and a_z are the velocity gradients in $x-, y-$, and $z-$ direction, respectively.

After defining the velocity field, travel times from an entry point to the three possible exit boundaries are calculated in each direction separately. The particle then exit the boundary with the minimum positive travel time. This minimum time is the time-of-fight (TOF) for the streamline in the block considered. Using this TOF, the exit point for this grid block is determined. This exit point is the entry point for the next grid block. One streamline is obtained by continuing this process until the particle reaches the boundary. Pollock's streamline method is, to date, the most commonly used streamline simulation method in the industry. It is noted that this method does not allow streamlines to exit from the same face as they enter. Hence, since this is physically possible, Pollock's method is expected to give large errors in situations with large grid blocks. However, the semi-analytical streamline simulation method proposed in this research allows streamlines to exit from the same face as they enter; and therefore, it is a more accurate approach.

2.2.4 Recent Streamline and Stream Tube Simulation Developments

Table 2.3 summarizes the recent development of streamline and stream tube simulation methods.

Thiele (1994) applied stream tube modeling to non-linear flow displacement processes in heterogeneous reservoirs, by mapping analytical solutions (Buckley and Leverett, 1942) along stream tubes in the displacement calculation. By this numerical dispersion is shown to reduce substantially. This method requires two to five orders of magnitude less computation time than the traditional finite difference simulation approach. It is also emphasized that, unlike the work in this research thesis, Thiele's approach used constant flow rate boundary conditions.

Datta-Gupta and King (1995) extended Pollock's particle tracking method in the reservoir engineering field to cases with arbitrary well configuration, also assuming constant flow rates. They used time-of-flight (TOF) as a spatial variable. In the TOF system, the transport equation along the streamline is solved. The TOF formulation decreases the influence of geological heterogeneity on transport calculations.

Bratvedt et al. (1996) modeled streamlines in three dimensions by extending Pollock's method and applying an operator splitting to incorporate gravity in the front tracking method. Their method is more accurate and computationally efficient compared to the traditional finite difference methods.

Batycky et al. (1997) developed a three-dimensional, two-phase streamline simulation method. In their model, heterogeneity, changing well conditions, gravity and mobility were considered in the numerical solutions by re-calculating streamlines at

Table 2.3: Recent Development of Streamline and Stream tube Modeling

Author (Year)	**Details**
Pollock (1988)	Introduced the most efficient semi-analytical streamline tracing method in three dimensions.
Thiele (1994)	Applied stream tube modeling to non-linear flow displacement processes in heterogeneous reservoirs.
Datta-Gupta and King (1995)	Extended Pollock's particle tracking method in the reservoir engineering field with an arbitrary configuration of wells.
Bratvedt et al. (1996)	Applied an operator splitting to incorporate gravity in the front tracking method.
Batycky et al. (1997)	Developed a heterogeneous three-dimensional, two-phase streamline simulation method.
Peddibhotla et al. (1997)	Used a three-dimensional mapping algorithm and a third order Total Variation Diminishing (TVD) scheme to solve the multi-phase flow equations.
Thiele (2001)	Summarized the applicability of streamline simulation.
Rodriguez et al. (2003)	Developed a full three-dimensional streamline simulator for two-phase incompressible flow that included capillary pressure.
Matringe and Gerritsen (2004)	Investigated the factors that affect the accuracy of streamline modeling.
Juanes and Matringe (2009)	Introduced a higher-degree of streamline tracing method in two-dimensional triangular or quadrilateral grid systems.

selected times. This streamline method decoupled a three-dimensional fluid displacement into two-phase multiple one-dimensional problems. Hence it has a consistent agreement with the finite difference method and is far more computationally efficient.

Front tracking along streamlines in their model also demonstrated that the streamline method is an efficient visualization tool in reservoir simulation.

Peddibhotla et al. (1997) presented two major improvements to streamline modeling. First, they used a three-dimensional mapping algorithm instead of averaging streamlines during changing well conditions. Second, to minimize numerical dispersion and prevent unphysical oscillations, they used a third order Total Variation Diminishing (TVD) scheme to solve the multi-phase flow equations.

Thiele (2001) summarized the applicability of streamline simulation for upscaling, quantifying displacement efficiency, history matching, and field optimization. Thiele also pointed out the advantage of streamline simulation in flow visualization, its capability on full field modeling, its computational speed, and its increasing ability to model more complicated physics.

Rodriguez et al. (2003) developed a full three-dimensional streamline simulator for a two-phase incompressible flow that included capillary pressure. In this model, the capillary pressure was separated from the convective part, so was the gravity term. This operator splitting was first used in Bratvedt et al. (1996).

Matringe and Gerritsen (2004) investigated the factors that affect the accuracy of the streamline modeling. They mentioned that for the homogeneous quarter five spot pattern, the analytical streamline pattern is known. The errors in streamline location, streamline arc length and time-of-flight can affect the streamline simulation results. They applied the mixed hybrid finite element method which is more accurate on flux calculations than the traditional finite difference method. They also presented two methods to improve streamline tracing within grid blocks: using an adaptive mesh refinement, or using a second order interpolation of the velocity field.

Juanes and Matringe (2009) introduced a higher-degree of streamline tracing method in two-dimensional triangular or quadrilateral grid systems. In their method, the mixed finite element method is used to solve the pressure and velocity field simultaneously. The velocity field is then used in the stream functions to trace streamlines. Compared to low-degree tracing such as Pollock's method, this high-degree tracing is more accurate and less sensitive to grid distortion. However, the velocity interpretation methods are limited to the mixed finite element framework.

2.2.5 Near-wellbore Streamline Simulation

Most streamline/stream tube methods were developed for full field scale. Very little literature exists relevant for near-wellbore streamline simulation.

Hadibeik et al. (2011) presented a streamline simulation method for near-wellbore fluid flow modeling in vertical and deviated wells. In their streamline tracing process, three coefficients used to trace the streamline path were introduced according to the divergence free flow velocity in cylindrical coordinates, *i.e.*

$$\nabla \cdot \vec{u} = \frac{1}{r}\frac{\partial}{\partial r}(ru_r) + \frac{1}{r}\frac{\partial}{\partial \theta}(u_\theta) + \frac{\partial}{\partial z}u_z = 0. \tag{2.4}$$

Rewriting Equation 2.4 results in:

$$\nabla \cdot \vec{u} = \sum_{i=1}^{3} c_i = c_r + c_\theta + c_z. \tag{2.5}$$

where c_i is a constant for each discretization cell. Hadibeik et al. (2011) proposed:

$$c_r = \frac{2(r_{i+1}u_{ri+1} - r_i u_{ri})}{r_{i+1}^2 - r_i^2}, \tag{2.6}$$

$$c_\theta = \frac{r_i(u_{\theta i+1} - u_{\theta i})}{\theta_{i+1} - \theta_i}, \tag{2.7}$$

$$c_z = \frac{(u_{zi+1} - u_{zi})}{z_{i+1} - z_i}. \tag{2.8}$$

where r, θ, and z describe cylindrical coordinates, u^r, u^θ, and u_z are the velocity in the radial direction, angular direction and vertical direction, respectively.

The time-of-flight across the cell is then given by:

$$\Delta\tau_{ri} = \frac{1}{c_r} ln\left(\frac{r_i u_{ri}}{r_0 u_{r0}}\right), \tag{2.9}$$

$$\Delta\tau_{\theta i} = \frac{1}{c_\theta} ln\left(\frac{u_{\theta i}}{u_{\theta 0}}\right), \tag{2.10}$$

$$\Delta\tau_{zi} = \frac{1}{c_z} ln\left(\frac{u_{zi}}{u_{z0}}\right). \tag{2.11}$$

In their method, the velocity relations seem to lack physical significance.

Skinner (2011) and Skinner, Johansen (2012) presented two-dimensional streamline modeling in the near-wellbore region. This simulation was based on Pollock's method and was used to evaluate the well completion strategies in general, and perforation skin in particular. They optimized the completion design by using the design of experiments methodology combined with streamline simulation. It is a useful tool for maximizing productivity for individual wells.

In their method, the reservoir is first divided into grid blocks in a polar coordinate system. Once the pressure in each grid block center p_o is found by solving the cylindrical Laplace equation, the velocities across the simulation grid block boundaries can be calculated by using Darcy's law for both radial j and angular i directions. After the face velocities are known, velocities throughout the entire reservoir are determined.

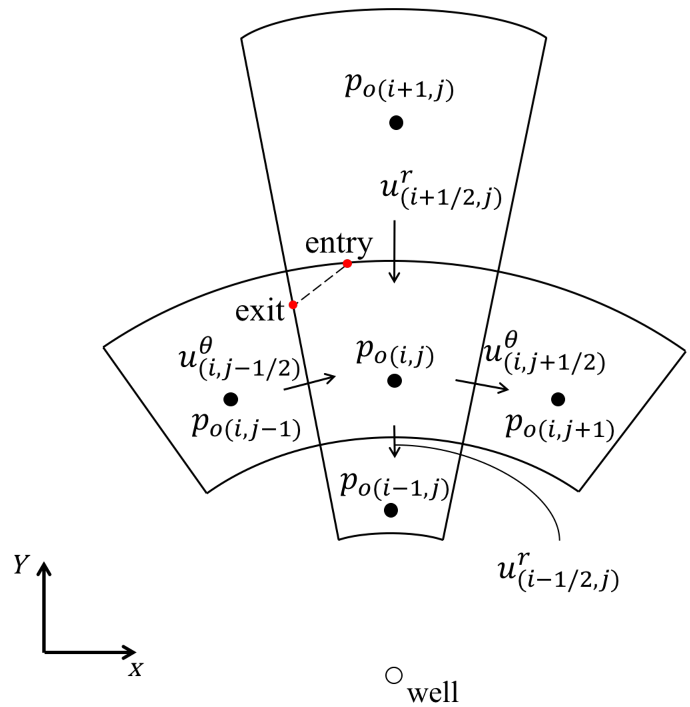


Figure 2.2: Pollock's Single Grid Block for Near-Wellbore Region in 2D

In Skinner (2011) and Skinner, Johansen (2012), the Pollock's method was employed in a log-lin fashion. For a single grid block (i, j) shown in Figure 2.2, the angular velocity u^θ varies linearly in the $\theta-$direction within each grid block. However, pressure is known to drop logarithmically in the radial direction towards the wellbore, hence the velocity in the radial direction u^r varies as the inverse of the radius. Equations 2.12 and 2.13 below are the general formulas for the velocity in $r-$ and $\theta-$directions, respectively, which is the idea of the streamline method applied in polar coordinates, *i.e.*

$$u^r_{i,j} = \frac{a^r_{i,j}}{r} + b^r_{i,j}, \tag{2.12}$$

$$u^\theta_{i,j} = a^\theta_{i,j}\theta + b^\theta_{i,j}. \tag{2.13}$$

The coefficients $a^r_{i,j}$, $b^r_{i,j}$, $a^\theta_{i,j}$ and $b^\theta_{i,j}$ in Equations 2.12 and 2.13 are determined by the boundary velocities and will differ for each grid block. They can be expressed as:

$$a_{i,j}^r = \frac{r_{i-1/2}r_{i+1/2}}{r_{i+1/2} - r_{i-1/2}}(u_{i-1/2,j}^r - u_{i+1/2,j}^r), \tag{2.14}$$

$$b_{i,j}^r = u_{i+1/2,j}^r - \frac{a_{i,j}^r}{r_{i+1/2}}, \tag{2.15}$$

$$a_{i,j}^\theta = \frac{u_{i,j-1/2}^\theta - u_{i,j+1/2}^\theta}{\theta_{j-1/2} - \theta_{j+1/2}}, \tag{2.16}$$

$$b_{i,j}^\theta = u_{i,j+1/2}^\theta - a_{i,j+1/2}^\theta \theta_{j+1/2}, \tag{2.17}$$

since the velocity distribution throughout the reservoir is known. For any entry point in a grid block, a particle can exit from three possible boundaries. In Pollock's method, the particle cannot exit from the same boundary as it entered because the streamlines are assumed to be straight lines. Clearly, this is a severe restriction. The required transit times t_r and t_θ for the particle to travel from the entry point to the possible boundaries are determined by Equations 2.18 and 2.19 below,

$$t_{i,j}^r = \int_{r_{ex}}^{r_{en}} \frac{dr}{u_r} = \int_{r_{ex}}^{r_{en}} \frac{dr}{\frac{a_{i,j}^r}{r} + b_{i,j}^r}, \tag{2.18}$$

$$t_{i,j}^\theta = \int_{\theta_{en}}^{\theta_{ex}} \frac{d\theta}{u_\theta} = \int_{\theta_{en}}^{\theta_{ex}} \frac{d\theta}{a_{i,j}^\theta \theta + b_{i,j}^\theta}. \tag{2.19}$$

In the near-wellbore region, the radius for the entry point is always larger than the exit point, due to the pressure distribution. For a homogeneous reservoir, the interval $[r_{ex}, r_{en}]$ travel time is:

$$t_{i,j}^r = \int_{r_{ex}}^{r_{en}} \frac{dr}{u_r} = \int_{r_{ex}}^{r_{en}} \frac{dr}{\frac{a_{i,j}^r}{r}} = \frac{1}{2a_{i,j}^r}(r_{en}^2 - r_{ex}^2). \tag{2.20}$$

For a heterogeneous reservoir, the particle travel time in the radial direction is:

$$t_{i,j}^r = \int_{r_{ex}}^{r_{en}} \frac{dr}{\frac{a_{i,j}^r}{r} + b_{i,j}^r} = \frac{1}{b_{i,j}^{r2}} \left[b_{i,j}^r (r_{en} - r_{ex}) - a_{i,j}^r ln \left(\frac{a_{i,j}^r + b_{i,j}^r r_{en}}{a_{i,j}^r + b_{i,j}^r r_{ex}} \right) \right]. \tag{2.21}$$

The general form for the particle travel time in angular direction is:

$$t^{\theta}_{i,j} = \int_{\theta_{en}}^{\theta_{ex}} \frac{d\theta}{a^{\theta}_{i,j}\theta + b^{\theta}_{i,j}} = \frac{1}{a^{\theta}_{i,j}} \left[ln \left(\frac{b^{\theta}_{i,j} + a^{\theta}_{i,j}\theta_{ex}}{b^{\theta}_{i,j} + a^{\theta}_{i,j}\theta_{en}} \right) \right]. \tag{2.22}$$

The time-of-flight (TOF) for each grid block is the minimum positive transit time:

$$TOF_{i,j} = min(t^{r}_{i,j}, t^{\theta}_{i,j}). \tag{2.23}$$

The actual exit point can then be determined by the TOF, together with the velocities in the radial and angular direction in each block.

Johansen (2010) proposed a new semi-analytical streamline simulation method that can be applied to the near-wellbore region. In this method, the pressure distribution polynomial function was assumed to be logarithmic-linear (log-lin) in 2D and bi-linear logarithmic (bilin-log) in 3D within each grid block. Depending on the pressure function, streamlines within each grid block can be expressed as smooth curves instead of straight line segments. This method is applied in this research, and will be described in detail in **Chapter 3**.

2.3 Front Tracking Method

The front tracking method is a method for calculating the convective motion of fronts throughout a reservoir. In the front tracking method, saturation discontinuities are calculated by conservation equations for the two-phase immiscible displacement process along streamlines. Table 2.4 reviews the front tracking method to date.

Table 2.4: Summary of Front Tracking Method

Author (Year)	**Details**
Buckley and Leverett (1942)	Introduced 1D Riemann problem for constant flow rate in two-phase flow.
Higgins and Leighton (1962)	Applied Buckley-Leverett (1942) theory along stream tubes.
Morel-Seytous (1965)	Expressed an analytical-numerical method in water flooding predictions.
Glimm et al. (1983)	Introduced hyperbolic equations to track the shock front without numerical and physical dispersion in 3D.
Bratvedt et al. (1992, 1996)	Developed a new front tracking scheme incorporating gravity in 3D.
Glimm et al. (1999)	Developed an improved algorithm for the interaction of a tracked contact discontinuity with an untracked shock wave.
Nilsen and Lie (2009)	Applied front tracking methods to streamline simulation in three-dimensional models.
Johansen and James (2015)	Introduced 1D Riemann problem for constant pressure boundaries for multi-component systems.
Johansen et al. (2016)	Introduced 1D Riemann problem for constant pressure boundaries for waterflooding.
Johansen and Liu (2016)	Introduced analytical solution of Riemann problems for two-phase flow in 3D stream tubes.

Buckley and Leverett (1942) presented the analytical solution for a fluid displacement front in an immiscible displacement process in one space dimension and constant flow rate. The Buckley-Leverett theory applies to two immiscible phases under the assumption that the flow rate is constant over time. It is reviewed in **Appendix A**.

Higgins and Leighton (1962) introduced stream tube bundles to model two-phase displacements in a complex rock geometry. Each stream tube was treated as a one-dimensional object throughout the displacement process, and the fluid saturation was calculated along the tubes by applying the Buckley-Leverett (1942) fractional flow theory for constant flow rate. This was the first time to apply the Riemann solutions in stream tubes. Although the stream tube bundles were fixed throughout the displacement process, the resistance within each tube was updated at the end of each time step instead of using a changing mobility field. Then, injection volumes into stream tubes were calculated based on tube resistances. Their model showed good agreement with experimental results. Higgins and Leighton (1962) utilized stream tubes to model three-phase displacements in porous media. The results also showed a good agreement with laboratory waterflooding data.

Morel-Seytoux (1965) expressed an analytical-numerical method in water flooding predictions. They discussed the impact of well pattern geometry in the two-phase displacement process. It is a simple model because some restrictive assumptions were made, for instance, fluids were assumed incompressible; mobility ratio was treated as a constant and the displacement front was assumed piston-like; gravity and capillary pressure were ignored in this model. Results showed that well pattern geometry is a major factor in predicting water-flood recovery. It also provided new ideas for changing mobility ratios.

Glimm et al. (1983) established the front tracking method which is similar to the streamline method in the sense that a local flow direction is calculated from a pressure equation. They introduced hyperbolic equations to track the shock front without numerical and physical dispersion for homogeneous and heterogeneous reservoirs. Their model can also be applied to immiscible displacements and variable mobility ratios.

Bratvedt et al. (1992) developed a new front tracking scheme incorporating gravity. They updated the hyperbolic conservation laws by a separate gravity driven computation and developed a new method to solve the saturation in the simulation process. In their computational process, the discontinuity surface was treated as an independent object. At that time, their model only focused on the front tracking of the discontinuity surface.

Bratvedt et al. (1996) improved the front tracking method along streamlines with gravity effects. Throughout the simulation, the pressure equation was solved implicitly, and the saturation equation was solved explicitly. A block-based numerical streamline method was used in the saturation calculation. In comparison, the saturation calculation in this research is based on an analytical method that can be used for an arbitrary geometry of stream tubes, which makes this research more accurate and more efficient.

Glimm et al. (1999) introduced a simplified description of the microtopology of the interface, based on interface crossings with cell block edges, and developed an improved algorithm for the interaction of a tracked contact discontinuity with an untracked shock wave.

Nilsen and Lie (2009) applied front tracking methods to streamline simulation in three-dimensional models. Their numerical results demonstrate that both streamlines and the front tracking method enable efficient simulation of compressible flow.

The Riemann problem for constant flow rate in two-phase flow was described in Buckley, Leverett (1942). The associated Riemann problem for constant pressure boundaries was recently published by Johansen and James (2015) and Johansen et al. (2016). Under the constant pressure boundaries, the flow rate varies over time. In this model, the analytical solution for flow rate as a function of time is determined. It also provides an analytical solution for the location of the displacement front at any given time, the time for frontal breakthrough at the outlet, and saturation profiles after frontal breakthrough.

Johansen and Liu (2016) introduced an analytical solution of Riemann problems for two-phase flow in 3D stream tube geometries. This method is applied to model the two-phase flow process in stream tubes in this research thesis, and will be described in detail in **Chapter 4**.

2.4 Visualization Experiments

All hydrocarbon reservoirs are heterogeneous. It is necessary to understand the physics associated with flow in these reservoirs. Visualization experiments can be used to study the two-phase displacement flow behavior in homogeneous and heterogeneous media, measure the interfacial tension and phase saturation, measure the relative permeability, and study oil recovery. In the petroleum field, visualization experiments are often performed in four main types of visual porous media: 1. Hele Shaw cells; 2. pore scale mircomodels; 3. unconsolidated porous media (glass beads packs and sand packs); 4. consolidated glass beads packs. This section reviews some of the visualization experiments to date.

2.4.1 Porous Media for Visualization Experiments

A Hele Shaw cell, a common model used to study fluid-fluid displacement, is constructed by using two parallel closely-spaced glass plates. Chuoke et al. (1959) conducted a Hele Shaw micromodel to study two-phase flow. They used two types of fluids (water-glycerine, and water with and without initial interstitial water) to displace oil through Hele Shaw cell. Paterson (1981) and Chen (1987) used circular Hele Shaw cells to observe the radial fingering phenomena. Butler and Mokry (1993) and James et al. (2008) studied the effect of Vapor Extraction Process (VAPEX) for reservoirs with free bottom water. The reasons for the popularity of the Hele Shaw cell in the visualization experiments are: 1. It can be easily constructed and set up; 2. It provides reasonable qualitative results.

Pore scale micromodels have been increasingly used to investigate the flow behavior of fluids on the pore scale. Most pore scale micromodels are made of glass plates with chemical reactions or interaction of a laser on the glass surface. Pore scale micromodels were used to relate pore structure and pore network to residual saturation (Chatzis et al., 1983). They are also used to study pore scale phenomena associated with the Vapor Extraction Process (VAPEX) for heavy oil recovery (Chatzis 2002, James and Chatzis 2004, James 2009) and capillary fingering effect (van der Marck and Glas, 1997). This type of model is relatively easy to fabricate and has the ability to choose the wetting properties depending on the material, and can reproduce the network pattern.

Unconsolidated porous media (glass beads packs and sand packs) have a structure similar to that of a Hele Shaw cell but with glass beads or sand between two glass plates. Because this type of porous media is cheap and easy to make, it is widely used to investigate two-dimensional two-phase flow problems. Chatenever and Calhoun (1952) used an unconsolidated porous media with a single layer of glass beads covered by color film to study the two-phase (brine and crude oil) immiscible displacement fluid behavior. Front patterns and the residual oil and water were observed and discussed. They found that in a system at steady state, two immiscible fluids that flow simultaneously in a porous medium will establish their own pathways. Since Chatenever and Calhoun (1952) the application of unconsolidated porous media has increased because it is convenient to fabricate. Some other unconsolidated porous media visualization experiments with heterogeneity are reviewed in **2.4.2**.

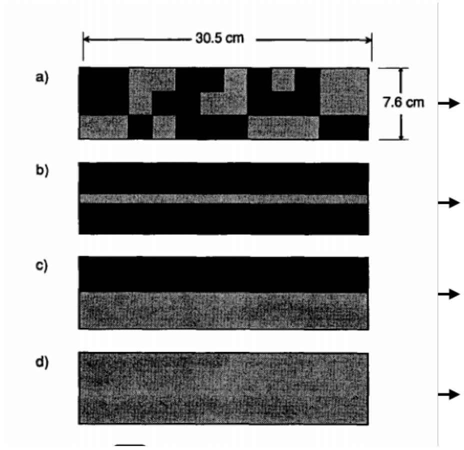
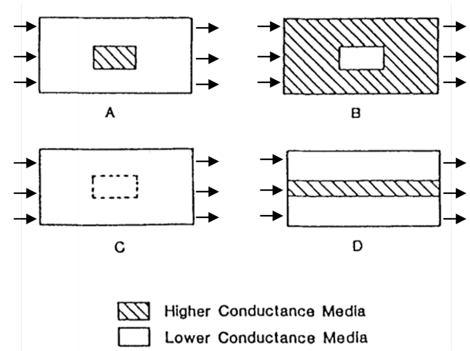
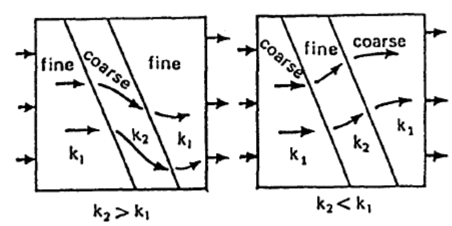
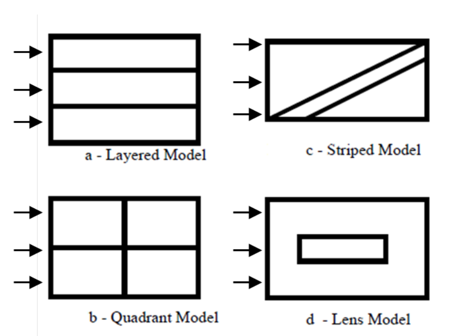
Consolidated porous media are usually created by melting glass beads and glass plates into one piece. James (2003) investigated the effect of the model height and dip angle on live oil production from consolidated glass beads saturated with bitumen using butane as solvent. However, consolidated porous media are relative complicated to fabricate.

2.4.2 Unconsolidated Visualization Experiments with Heterogeneity

Flow experiments performed in this research thesis use two-dimensional visual models packed with unconsolidated glass beads. Table 2.5 summarizes unconsolidated glass beads visualization experiments reviewed in this section and describes the experimental goals and the porous media patterns. In this table, the figures in column 3 are references form the literature list in column 1.

Brock and Orr Jr. (1991) performed flow visualization experiments and numerical simulations to study the combined effects of viscous fingering and permeability heterogeneity. There were four different glass bead packs used. In each model, experiments were performed at three different flow rates ($3, 6$, and $9\ ml/min$) and mobility ratios ($M = 1$, 40, and 80). The initial fluids used were two grades of mineral oil, Soltrol 10 (a refined isoparaffin), and toluene. Injected fluids were dyed with Automate Red B

Table 2.5: Summary of Unconsolidated Visualization Experiments with Heterogenity

Author (Year)	Experiment Goals	Porous Media Patterns
Brock and Orr Jr. (1991)	Performed flow visualization experiments and numerical simulations to study the combined effects of viscous fingering and permeability heterogeneity.	
Dawe et al. (1992)	Studied the effects of well-defined heterogeneous porous media on immiscible flooding.	
Roti and Dawe (1993)	Studied the effects of layer thickness, permeability contrast, angle of layer to flow direction, mobility ratio, and flood rate.	
Silva and Dawe (2003)	Used different geologic models to study the effect of permeability and wettability heterogeneities.	

dye at a concentration of approximately 0.005 grams of dye per gram of fluid. They found that in the homogeneous model, fingering patterns were sensitive to the mobility ratio but not to flow rate. In the layered and heterogeneous model, flow was largely determined by the patterns of heterogeneity. Their experiment results and simulation results were similar.

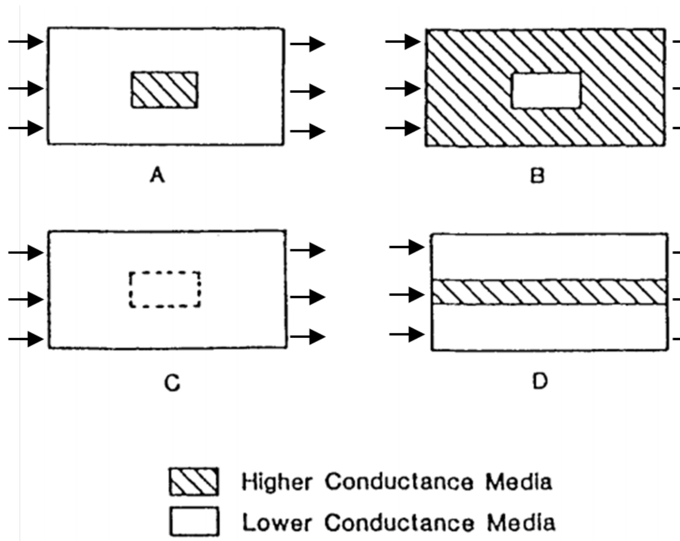


Figure 2.3: Packing Patterns used in Dawe et al. (1992)

Dawe et al. (1992) studied the effects of well-defined heterogeneous porous media on immiscible flooding by using the glass beads pack. In their model, the heterogeneities were layers and lenses, with some of the lenses exhibiting a wettability contrast. As shown in Figure 2.3, the lens patterns A and B had a conductance contrast of 2.55; pattern C consisted of a single-glass matrix but with a lens having hydrophobic properties (the lens beads were coated with the water repellent chemical dimethyldichlorosilane). The lenses were 3 cm wide and 10 cm long. The layered pattern D had a conductance contrast of 2.5 (center layer has the higher conductance) and a layer width of 2 cm. The effect of flooding rate ($6.67 \times 10^{-2}, 1 \times 10^{-2}, 1.67 \times 10^{-2}$, and 0.5×10^{-2} cc/sec), initial fluid saturations, and wettability on drainage and imbibition were discussed. The primary conclusions that Dawe et al. obtained were that capillary forces become more important and can even dominate the flow, the balance between capillary and viscous forces is rate dependent, and the effects of capillary forces become larger as the flow rate decreases.

Roti and Dawe (1993) performed experiments on glass beads packs and numerical simulations to study flow displacements, effluent profiles and streamline patterns for layered systems with flow not parallel to the layers. They studied the effects of layer thickness (0 to 0.4 times the width of the model), permeability contrast (0.1 to 25), the angle of the layer to flow direction (0 to 90°), mobility ratio, and flow rate. They found that for miscible displacements, the breakthrough recovery decreases with an increase in permeability contrast.

Silva and Dawe (2003) performed two-dimensional visualization immiscible displacement experiments in unconsolidated glass beads models. Different geologic models were used to study the effect of permeability and wettability heterogeneities. In their waterflooding experiments, high permeability regions were bypassed due to capillary pressure differences. They pointed out that their results could also be used to study the reservoir production performance in immiscible displacement.

The unconsolidated visualization experiments with heterogeneity described so far are in 2D porous media. However, they are all simple 1D fluid displacements from one side of the porous media to the opposite side. Next, radial visualization experiment (real 2D flow) will be reviewed.

2.4.3 Radial Visualization Experiment

Paterson (1981) used a Hele Shaw cell with inward and outward flow to observe the radial fingering phenomena. The Hele Shaw cell used in the experiments consisted of two 13 mm thick glass disks, 600 mm in diameter, spaced a few millimetres apart. In the experiments, the fluid was injected or withdrawn in the center of the porous media at a constant rate. The width of fingers was examined, and they provided an approximate equation for the growth of the fingers.

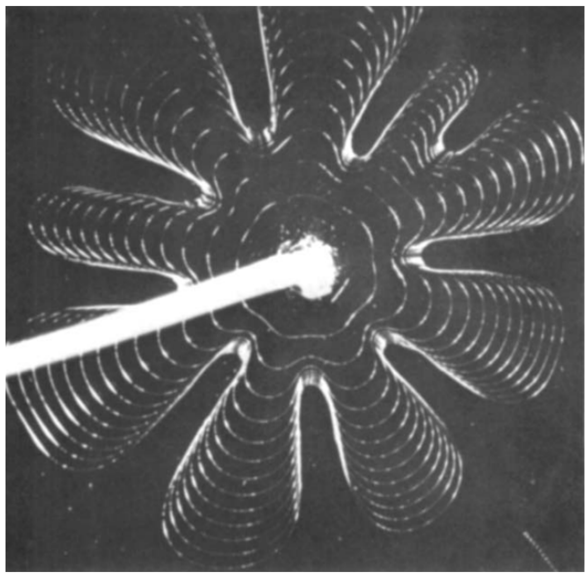


Figure 2.4: Fluid Patterns in Paterson (1981)

Chen (1987) performed viscous fingering experiments in Hele Shaw cells by injecting a liquid to radially displace a much more viscous liquid. The Hele-Shaw cell was made by using two glass plates of 0.55 $cm \times 10$ $cm \times 10$ cm with four spacers of $75+2$ $microns$ in thickness clamped in between. The top plate has a small hole (of 0.17 cm in diameter) drilled in the center for injecting the fluids. Both smooth and etched plates were used to study the influence of plate roughness on the fingering mechanism. The fingering patterns were strongly affected by the geometry of the network etched on the glass plate surface. Chen used different flow rates ($1.4 \times 10^{-4}, 2.0 \times 10^{-4}$, and 5.6×10^{-4} ml/s) to study the influence of the flow rate in the miscible case. The results showed that fingering patterns were strongly affected by the flow rate in the immiscible case. Figure 2.5 shows the radial viscous fingering patterns in Chen (1987).

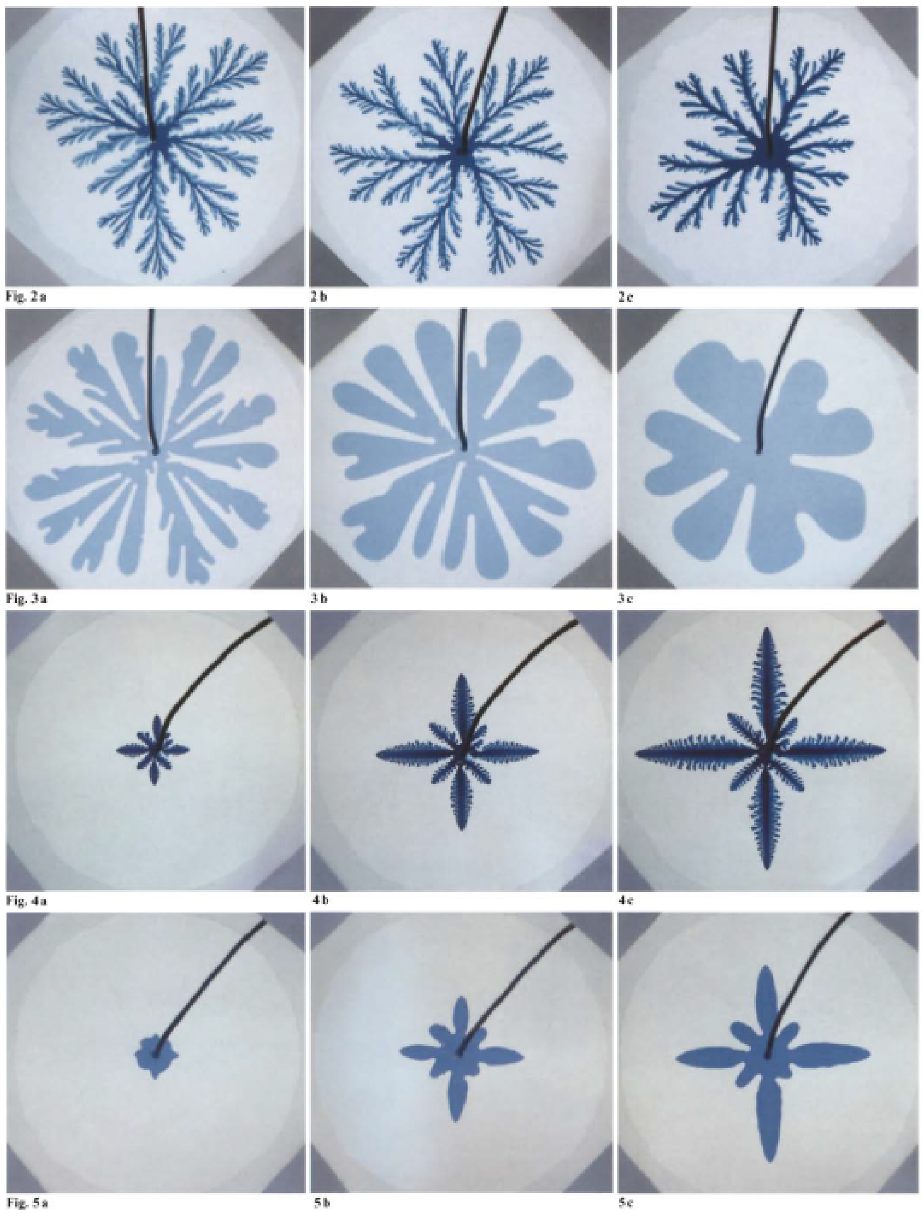
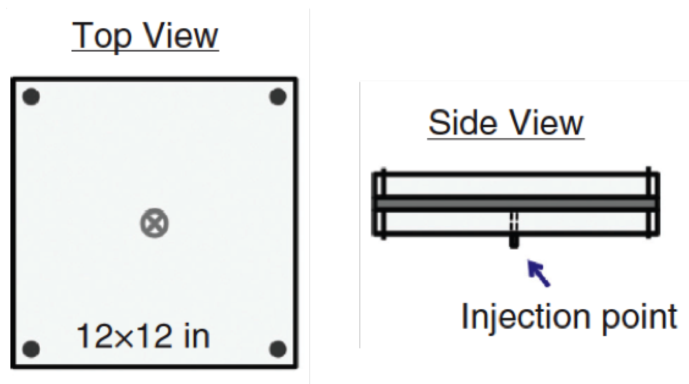
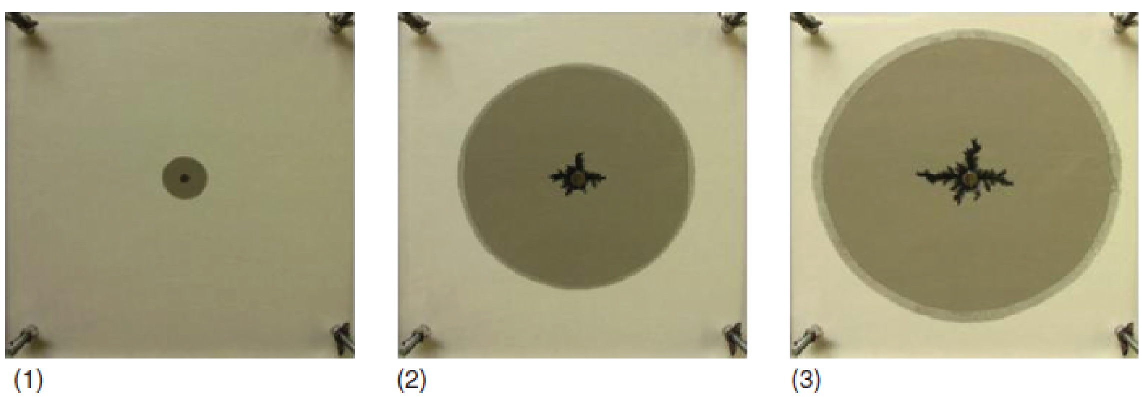


Figure 2.5: Radial Viscous Fingering Patterns in Chen (1987)

Huang et al. (2012) conducted a series of injection experiments in a Hele-Shaw cell-like radial-flow device which was filled with fine sand. The objective of these experiments was to investigate the flow mechanisms when porous media were invaded and displaced by aqueous glycerin solution or polyacrylamide solution. They found that with the same porous media properties, as the injection velocity and the fluid viscosity increase, the porous media transited from solid-like to fluid-like. They also mentioned that the Hele-Shaw cell is a useful tool for understanding the flow mechanisms in the injection processes.



(a) Experimental Setup



(b) Fluid Pattern

Figure 2.6: Experimental Setup and Fluid Patterns in Huang et al. (2012)

The radial visualization experiments, which are made of transparent materials, enable the visual observation in two dimensions. However, they have limitations for 3D visualization. For 3D visualization a CT scanner and NMR (nuclear magnetic resonance) are needed.

Valsecch et al. (2012) developed a new approach to visualize the geometrical features and flow patterns for various well completion types in the near-wellbore region. They used the acid-stimulated carbonate rock as the porous media. CT scanning technology was applied for the image processing. As can be seen from Figure 2.7, water was continuously injected through a pipe. The non-metallic hoses distribute fluid to the outer shell of the experiment. Then, the fluid flows through the porous media toward the inner pipe. The fluid flows out from the inner pipe was recycled to pump. Streamlines and 3D surface reconstructions were visualized using magnetic resonance imaging (MRI) acquisitions with the Paraview software to better understand the structures and the fluid flow movement near the wellbore. Their visualization experiment was based on a CT scanner. In this research thesis, the water front will be directly visualized and recorded by a camera.

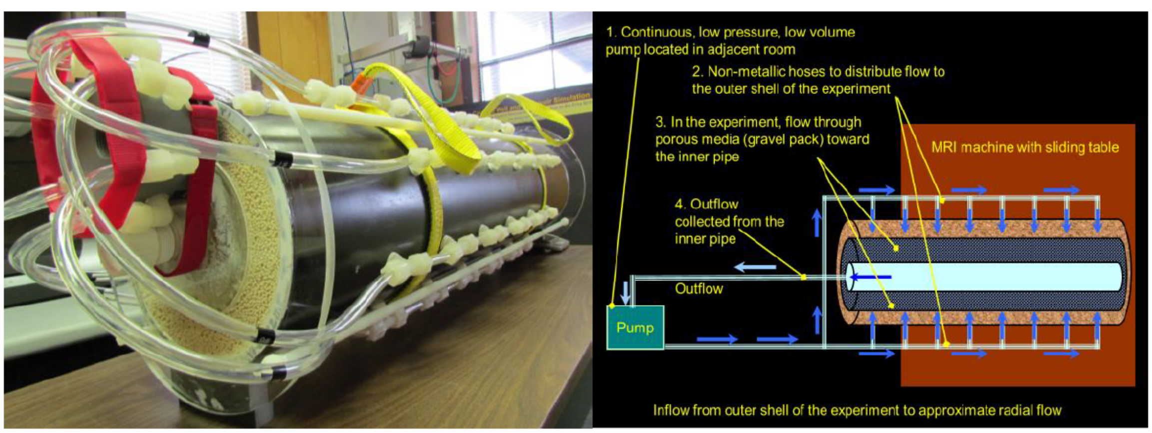


Figure 2.7: Experiment Vessel and Schematic of Experiment in Valsecch et al. (2012)

2.5 Conclusion

As in the literature, streamline simulation was mainly applied in the Cartesian coordinate to simulate the full field model. Very little literature exists relevant for the near-wellbore streamline simulation. Hadibeik et al. (2011) introduced a streamline tracing method according to the divergence-free flow velocity in the cylindrical coordinate. However, in their method, the velocity relations seem to lack physical significance. Skinner (2011) and Skinner, Johansen (2012) presented 2D streamline modeling based on Pollock's method in the near-wellbore region. In their model, streamlines excessively avoid the heterogeneous areas. Their model may require large grid refinement to provide accurate results. The streamline simulation method applied in this research depends on a logarithmic-linear pressure function inside each grid block. It is believed that this is the first time a strict logarithmic interpolation is used in the pressure assumption in the radial direction.

The visualization experiments performed in the previous literature were performed in square or rectangular porous media to study the two-phase displacement flow behavior, measure the interfacial tension and phase saturation, measure the relative permeability, and study oil recovery. A few experiments were used to study the flow patterns in the near-wellbore region. Valsecch et al. (2012) performed their near-wellbore visualization experiment based on a CT scanner. In this research thesis, a radial visualization porous media is used. The waterfronts in the near-wellbore region are directly visualized and recorded by a camera during the experiment. Pressures at the outlet and inlet of the porous media are kept constant, which is different from most visualization experiments in the literature.

Chapter 3

Streamline Simulation Methodology in Near-Wellbore Regions

In reservoir simulation, the properties of reservoir fluids and porous media are used to predict changes in reservoir pressure and fluid saturation with high precision. Streamline models represent an efficient reservoir simulation method. The semi-analytical streamline simulation method proposed in this research thesis first solves the pressure distribution on the static grid block system using the conventional finite difference method. Then, it uses smooth curves in the grid block to represent the streamlines in each block by assuming that the pressure changes along the block boundaries are linear in axial and angular directions and logarithmic in the radial direction.

In this chapter, the basic information for developing the streamline model in the near-wellbore region is first discussed. Next, the methodologies for generating streamlines using the semi-analytical streamline simulation method in 2D polar coordinate systems and 3D cylindrical coordinate systems based on a log-lin and bilin-log pressure assumption are described, respectively. The streamline generation methodology presented in this research thesis is an expanded version of that in Johansen (2010). A finite difference method calculates pressure in the center of grid blocks.

The semi-analytical streamline simulation method is based on the corner pressures for each grid block. The determination of these corner pressures from pressures in grid block centers is also presented in this chapter. Next, the pressure analyses are described. Finally, the streamline tracing procedure is described.

3.1 Near-Wellbore Model Geometry and Assumptions

The 2D and 3D streamline models generated in this research thesis are performed in the polar coordinate system and the cylindrical system, respectively. Figure 3.1 depicts a well located in the center $(0, 0)$ of the the polar grid with polar coordinates of (r, θ). The well has a radius of r_w with a corresponding constant pressure of p_w. The exterior boundary of the near-wellbore region is r_e with a constant external pressure of p_e. The scenario is realistic for large reservoirs in the early stages of production, as well as reservoirs with maintained pressure support by water injection (Skinner, 2011).

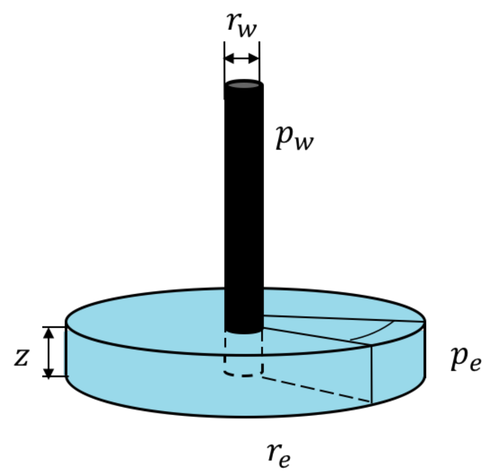


Figure 3.1: Near-Wellbore Region Sketch in 3D (Skinner, 2011)

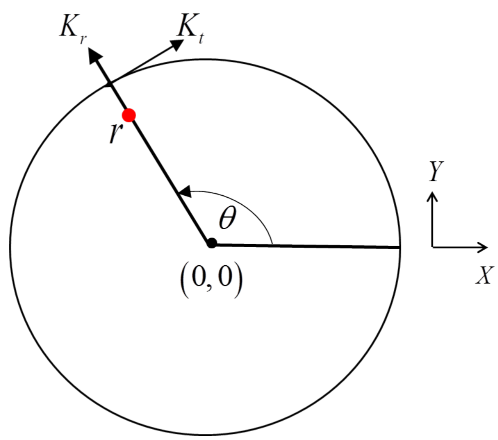


Figure 3.2: Relationship between Cartesian Coordinate System and the 2D Polar Coordinate System

The relationships between the Cartesian coordinate system (x, y) and the polar coordinate system (r, θ) are shown in Figure 3.2. As illustrated in Figure 3.2, the radius r for any point is measured from point $(0, 0)$ in the polar systems and the angle θ is measured counter-clockwise for the positive $x-$axis. The relationship can also be written as:

$$x = rcos\theta, \tag{3.1}$$

$$y = r \sin \theta, \tag{3.2}$$

$$r = \sqrt{x^2 + y^2}, \tag{3.3}$$

$$\theta = \arctan\left(\frac{y}{x}\right). \tag{3.4}$$

3.1.1 Near-Wellbore Model Geometry

In the polar coordinate system, the reservoir is segmented into grid blocks. The number of grid blocks in the radial and angular direction are N and M, respectively; i represents the radial block index and j represents the angular block index; $i \pm 1/2$ represents boundary in the radial direction and $j \pm 1/2$ represents boundary in the angular direction. The grid block construction is shown in Figure 3.3.

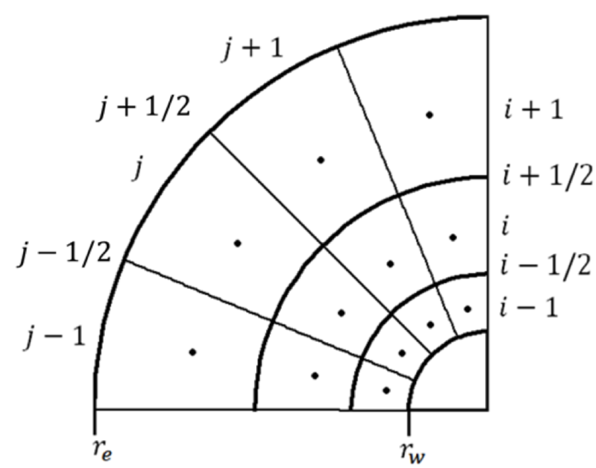


Figure 3.3: Generalized Grid Block in the 2D Polar Coordinate System

The grid blocks are not of equal size in the radial direction. The distance between pressure nodes in the radial direction is logarithmical to represent a typical pressure distribution in the near-well region. By doing this, the pressure drops between adjacent radial nodes are equal in a homogeneous isotropic medium. Equation 3.5 shows the relationship between two pressure nodes in the radial direction:

$$r_i = ar_{i-1} = r_w a^{i-1}, \tag{3.5}$$

where $i = 1, 2, \cdots, N$, where a is a constant.

Using Equation 3.5 for wellbore radius and the reservoir radius:

$$r_e = r_N = r_w a^{N-1}. \tag{3.6}$$

The constant a can then be expressed as:

$$a = \left(\frac{r_e}{r_w}\right)^{\frac{1}{N-1}}. \tag{3.7}$$

For any given wellbore radius and reservoir radius, the radius for any pressure node then can be written as:

$$r_i = r_w \left(\frac{r_e}{r_w}\right)^{\frac{i-1}{N-1}}. \tag{3.8}$$

Once the node radii are determined, the block boundaries for the interlock flow calculation are defined by (Aziz and Settari, 1979):

$$r_{i+1/2} = \frac{r_{i+1} - r_i}{ln(r_{i+1}/r_i)}. \tag{3.9}$$

3.1.2 Near-Wellbore Model Assumptions

As mentioned in **Chapter 1**, previous streamline simulations for two-phase flow were based on the assumption of constant flow rate. For most laboratory and real field water injection operations, fixed injection and production pressures are applied. In this research thesis, streamline simulation is performed under the assumption of constant pressure boundaries. Water front calculations are also performed under the same boundary conditions.

Streamline simulation is well suited for problems dominated by convection; however, in diffusive problems such as gas expansion and capillary pressure, streamlines are not well defined. Several researchers have incorporate these effects, e.g. Bratvedt et al. (1996) for the inclusion of gravity and Pasarai and Arihara (2005) incorporated effects of diffusion. The streamline simulation applied in this research thesis is based on the following assumptions:

- Incompressible fluid and rock
- Diffusive effects are negligible
- Gravity is ignored

This will yield an elliptic pressure equation, for which streamlines are well defined. These assumptions are valid for vertical wells in an oil reservoir. In an oil reservoir, oil and water can be assumed incompressible and diffusion free. For a long vertical well, the near-wellbore region radius is not long compared to the vertical thickness of the reservoir. This means that gravity plays a minority role in the displacement process. It is emphasized that these restrictions can be removed by invoking the methods in Bratvedt et al. (1996) and Pasarai and Arihara (2005).

3.2 Equations in Cylindrical Coordinate Systems

3.2.1 The Gradient Operator in Cylindrical Coordinate Systems

The gradient represents the maximum rate of change (vector) of a property. In Cartesian coordinates the gradient operator can be written as:

$$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right). \tag{3.10}$$

Applying the chain rule for differentiation in the $x-$direction and $y-$direction,

$$\frac{\partial}{\partial x} = \frac{\partial}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial}{\partial \theta}\frac{\partial \theta}{\partial x},f \tag{3.11}$$

$$\frac{\partial}{\partial y} = \frac{\partial}{\partial r}\frac{\partial r}{\partial y} + \frac{\partial}{\partial \theta}\frac{\partial \theta}{\partial y}. \tag{3.12}$$

According to Equations 3.1 to 3.4:

$$\frac{\partial r}{\partial x} = \frac{2x}{2\sqrt{x^2+y^2}} = \frac{x}{r} = \cos\theta, \tag{3.13}$$

$$\frac{\partial r}{\partial y} = \frac{2y}{2\sqrt{x^2+y^2}} = \frac{y}{r} = \sin\theta, \tag{3.14}$$

$$\frac{\partial \theta}{\partial x} = \frac{\partial}{\partial x}(arctan\left(\frac{y}{x}\right)) = -\frac{y}{r^2} = -\frac{1}{r}\sin\theta, \tag{3.15}$$

$$\frac{\partial \theta}{\partial y} = \frac{\partial}{\partial y}(arctan\left(\frac{y}{x}\right)) = \frac{x}{r^2} = \frac{1}{r}\cos\theta. \tag{3.16}$$

Therefore, the gradient in the cylindrical coordinate system is:

$$\nabla = \begin{bmatrix} \cos\theta & -\frac{\sin\theta}{r} & 0 \\ \sin\theta & \frac{\cos\theta}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial r} \\ \frac{\partial}{\partial \theta} \\ \frac{\partial}{\partial z} \end{bmatrix}. \tag{3.17}$$

3.2.2 The Pressure Gradient in Cylindrical Coordinate Systems

In the cylindrical coordinate system, the unit vectors $\vec{e_r}$, $\vec{e_\theta}$, and $\vec{e_z}$ are:

$$\vec{e_r} = [\cos\theta, \sin\theta, 0], \tag{3.18}$$

$$\vec{e_\theta} = [-\sin\theta, \cos\theta, 0], \tag{3.19}$$

$$\vec{e_z} = [0, 0, 1]. \tag{3.20}$$

Therefore the gradient operator is:

$$\nabla_r = \vec{e_r} \cdot \nabla = \cos^2\theta \frac{\partial}{\partial r} + \sin^2\theta \frac{\partial}{\partial r} = \frac{\partial}{\partial r}, \tag{3.21}$$

$$\nabla_\theta = \vec{e_\theta} \cdot \nabla = \frac{1}{r}\sin^2\theta \frac{\partial}{\partial \theta} + \frac{1}{r}\cos^2\theta \frac{\partial}{\partial \theta} = \frac{1}{r}\frac{\partial}{\partial \theta}, \tag{3.22}$$

$$\nabla_z = \vec{e_z} \cdot \nabla = \frac{\partial}{\partial z}. \tag{3.23}$$

The pressure gradient in the cylindrical coordinate system, therefore, is:

$$\nabla_{r,\theta,z} p = \left[\frac{\partial p}{\partial r}, \frac{1}{r}\frac{\partial p}{\partial \theta}, \frac{\partial p}{\partial z}\right]. \tag{3.24}$$

3.2.3 Darcy's Law in the Near-Wellbore Region

In the cylindrical coordinate system, the volumetric flux can be written by Darcy's Law:

$$\vec{u} = -\frac{1}{\mu}\nabla_{r,\theta,z} p \cdot \bar{K} = -\frac{1}{\mu}\left[\frac{\partial p}{\partial r}, \frac{1}{r}\frac{\partial p}{\partial \theta}, \frac{\partial p}{\partial z}\right] \cdot \begin{bmatrix} K_r & K_\theta & 0 \\ K_\theta & K_t & 0 \\ 0 & 0 & K_z \end{bmatrix} \tag{3.25}$$

where K_r, K_t, and K_z are the diagonal elements of the permeability tensor in the polar coordinates given by Equations 3.26 and 3.27 below and K_θ is the off-diagonal element defined by Equation 3.28. These permeabilities can be calculated from the principal permeabilities in the $x-$ and $y-$directions:

$$K_r = K_x \cos^2\theta + K_y \sin^2\theta, \tag{3.26}$$

$$K_t = K_x \sin^2\theta + K_y \cos^2\theta, \tag{3.27}$$

$$K_\theta = (K_y - K_x)\sin\theta\cos\theta. \tag{3.28}$$

Darcy's law (Equation 3.25) for the components of the flux u in $r-$, $\theta-$, and $z-$directions are therefore:

$$u_r = -\frac{1}{\mu}\left(K_r\frac{\partial p}{\partial r} + \frac{1}{r}K_\theta\frac{\partial p}{\partial \theta}\right), \tag{3.29}$$

$$u_\theta = -\frac{1}{\mu}\left(K_\theta\frac{\partial p}{\partial r} + \frac{1}{r}K_t\frac{\partial p}{\partial \theta}\right). \tag{3.30}$$

$$u_z = -\frac{1}{\mu}\left(K_z\frac{\partial p}{\partial z}\right). \tag{3.31}$$

The Laplace Equation is a second order partial differential equation. In the cylindrical coordinate system, the Laplace equation is:

$$\nabla_{r,\theta,z} \cdot \vec{u} = 0. \tag{3.32}$$

Expressing mass conservation of an incompressible fluid with fixed boundary conditions, the volumetric flux in Equation 3.25 produces:

$$\nabla_{r,\theta,z} \cdot \vec{u} = \left[\frac{\partial}{\partial r}, \frac{1}{r}\frac{\partial}{\partial \theta}, \frac{\partial}{\partial z}\right] \cdot (u_r, u_\theta, u_z) = 0, \tag{3.33}$$

i.e.,

$$\nabla_{r,\theta,z} \cdot \vec{u} = \frac{1}{r}\frac{\partial}{\partial r}(r u_r) + \frac{1}{r}\frac{\partial}{\partial \theta}(u_\theta) + \frac{\partial}{\partial z}(u_z) = 0, \tag{3.34}$$

This yields the general Laplacian in a cylindrical coordinate system:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(rK_r\frac{\partial p}{\partial r}\right) + \frac{1}{r}\frac{\partial}{\partial r}\left(K_\theta\frac{\partial p}{\partial \theta}\right) + \frac{1}{r}\frac{\partial}{\partial \theta}\left(K_\theta\frac{\partial p}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K_t\frac{\partial p}{\partial \theta}\right) + \frac{\partial}{\partial z}\left(K_z\frac{\partial p}{\partial z}\right) = 0. \tag{3.35}$$

In an isotropic medium, $K_r = K_t = K$ and $K_\theta = 0$. Because the media can still be heterogeneous, K does not cancel. The Laplacian Equation 3.34 in isotropic media then simplifies to:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(rK\frac{\partial p}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K\frac{\partial p}{\partial \theta}\right) + \frac{\partial}{\partial z}\left(K_z\frac{\partial p}{\partial z}\right) = 0. \tag{3.36}$$

3.3 Solution of the Pressure Distribution

In the polar coordinate system, the reservoir is subdivided into grid blocks. Figure 3.4 is the discretized near-wellbore region. The discretized pressures in both radial and angular directions are given in Equations 3.37 and 3.38 below; $i = 1, \cdots, N$; $j = 1, \cdots, M$:

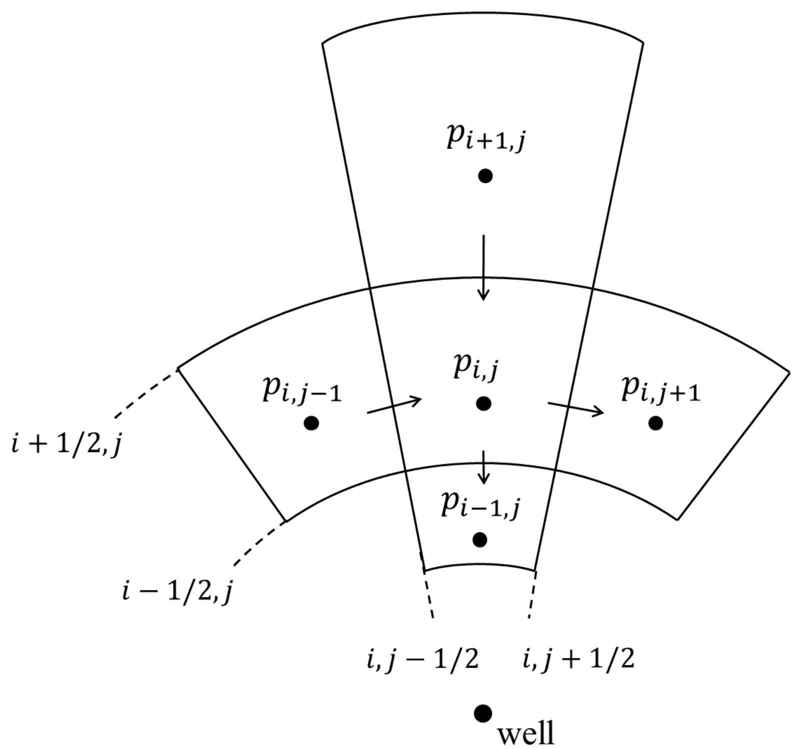


Figure 3.4: Discretization of Near-Wellbore Grid Blocks in 2D

$$\frac{1}{r}\frac{\partial}{\partial r}\left(rK\frac{\partial p}{\partial r}\right) \cong \frac{r_{i+1/2,j}K_{i+1/2,j}(p_{i+1,j}-p_{i,j})}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i+1,j}-r_{i,j})}$$
$$-\frac{r_{i-1/2,j}K_{i-1/2,j}(p_{i,j}-p_{i-1,j})}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i,j}-r_{i-1,j})}, \tag{3.37}$$

$$\frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K\frac{\partial p}{\partial \theta}\right) \cong \frac{K_{i,j+1/2}(p_{i,j+1}-p_{i,j})}{r_{i,j}^2(\theta_{i,j+1/2}-\theta_{i,j-1/2})(\theta_{i,j+1}-\theta_{i,j})}$$
$$-\frac{K_{i,j-1/2}(p_{i,j}-p_{i,j-1})}{r_{i,j}^2(\theta_{i,j+1/2}-\theta_{i,j-1/2})(\theta_{i,j}-\theta_{i,j-1})}, \tag{3.38}$$

where $K_{i\pm1/2,j}$ and $K_{i,j\pm1/2}$ are the upscaled permeability for the adjacent grid blocks in the radial and angular direction, respectively. They are determined by K_r and K_t:

$$K_{i+1/2,j} = \frac{ln(\frac{r_{i+1,j}}{r_{i,j}})}{\frac{1}{K_{i,j}^r}ln(\frac{r_{i+1/2,j}}{r_{i,j}}) + \frac{1}{K_{i+1,j}^r}ln(\frac{r_{i+1,j}}{r_{i+1/2,j}})}, \tag{3.39}$$

$$K_{i,j+1/2} = \frac{2K_{i,j}^t K_{i,j+1}^t}{K_{i,j}^t + K_{i,j+1}^t}. \tag{3.40}$$

If the discretization in angular direction is uniform, Equation 3.38 is simplified to:

$$\frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K\frac{\partial p}{\partial \theta}\right) \cong \frac{K_{i,j+1/2}(p_{i,j+1}-p_{i,j})}{r_{i,j}^2\Delta\theta^2} - \frac{K_{i,j-1/2}(p_{i,j}-p_{i,j-1})}{r_{i,j}^2\Delta\theta^2}, \tag{3.41}$$

Using this to discretize Equation 3.36 and rearranging results in:

$$\left(\frac{r_{i+1/2,j}K_{i+1/2,j}}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i+1,j}-r_i,j)}\right)p_{i+1,j} + \left(\frac{K_{i,j+1/2}}{r_{i,j}^2\Delta\theta^2}\right)p_{i,j+1} +$$
$$\left(\frac{r_{i-1/2,j}K_{i-1/2,j}}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i,j}-r_{i-1,j})}\right)p_{i-1,j} + \left(\frac{K_{i,j-1/2}}{r_{i,j}^2\Delta\theta^2}\right)p_{i,j-1} -$$
$$\left[\left(\frac{r_{i+1/2,j}K_{i+1/2,j}}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i+1,j}-r_{i,j})}\right) + \left(\frac{r_{i-1/2,j}K_{i-1/2,j}}{r_{i,j}(r_{i+1/2,j}-r_{i-1/2,j})(r_{i,j}-r_{i-1,j})}\right)\right.$$
$$\left. + \left(\frac{K_{i,j+1/2}+K_{i,j-1/2}}{r_{i,j}^2\Delta\theta^2}\right)\right]p_{i,j} = 0. \tag{3.42}$$

The general expression of the discretized Laplacian (Equation 3.42) for an isotropic reservoir is:

$$a_{i,j}p_{i,j} + b_{i,j}p_{i,j+1} + c_{i,j}p_{i,j-1} + d_{i,j}p_{i-1,j} + e_{i,j}p_{i+1,j} = 0. \tag{3.43}$$

Comparing Equation 3.42 and 3.43 we obtain the expression for the coefficients:

$$\begin{aligned} a_{i,j} = -\Bigg[&\left(\frac{r_{i+1/2,j}K_{i+1/2,j}}{r_{i,j}(r_{i+1/2,j} - r_{i-1/2,j})(r_{i+1,j} - r_{i,j})} \right) \\ + &\left(\frac{r_{i-1/2,j}K_{i-1/2,j}}{r_{i,j}(r_{i+1/2,j} - r_{i-1/2,j})(r_{i,j} - r_{i-1,j})} \right) + \left(\frac{K_{i,j+1/2} + K_{i,j-1/2}}{r_{i,j}^2 \Delta\theta^2} \right) \Bigg], \end{aligned} \tag{3.44}$$

$$b_{i,j} = \left(\frac{K_{i,j+1/2}}{r_{i,j}^2 \Delta\theta^2} \right), \tag{3.45}$$

$$c_{i,j} = \left(\frac{K_{i,j-1/2}}{r_{i,j}^2 \Delta\theta^2} \right), \tag{3.46}$$

$$d_{i,j} = \left(\frac{r_{i-1/2,j}K_{i-1/2,j}}{r_{i,j}(r_{i+1/2,j} - r_{i-1/2,j})(r_{i,j} - r_{i-1,j})} \right), \tag{3.47}$$

$$e_{i,j} = \left(\frac{r_{i+1/2,j}K_{i+1/2,j}}{r_{i,j}(r_{i+1/2,j} - r_{i-1/2,j})(r_{i+1,j} - r_{i,j})} \right). \tag{3.48}$$

The finite-difference formulation of the Laplace Equation (3.42) is a system of linear equations of the form:

$$Ap = D \tag{3.49}$$

where p is a vector of unknown grid block pressures. The matrix A is the coefficient matrix, representing the inter-block permeabilities, and D is the vector containing the boundary conditions.

3.3.1 Wellbore and External Boundary Conditions

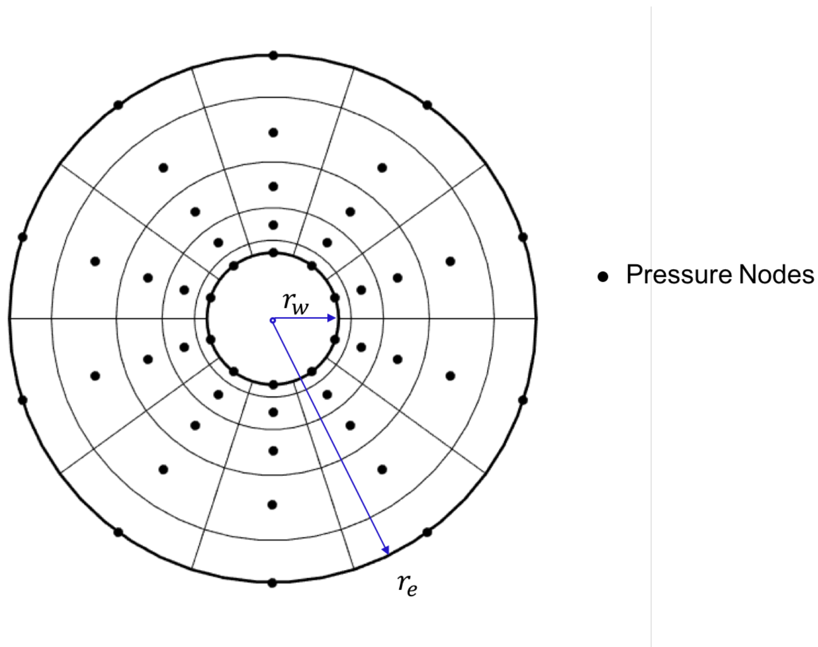


Figure 3.5: Pressure Nodes for Near-Wellbore Grid Blocks in 2D

As can be seen from Figure 3.5, node points for the internal and external boundaries are located on the boundaries of the grid blocks. Hence, the internal boundary radius $r_{1,j}$ is equal to be the wellbore radius r_w. Specifically, the first node radius is the same as the wellbore radius. This ensures the first node pressure $p_{1,j}$ is the wellbore pressure $p_{w,j}$, *i.e.*

$$p_{1,j} = p_{w,j} = p_w, \tag{3.50}$$

$$r_{1,j} = r_w. \tag{3.51}$$

Similar to the internal boundary conditions, the external node radius $r_{N,j}$ is the reservoir radius r_e:

$$r_{N,j} = r_e. \tag{3.52}$$

The corresponding node pressure at the external boundary $p_{N,j}$ is equal to the reservoir pressure $p_{e,j}$:

$$p_{N,j} = p_{e,j}. \tag{3.53}$$

3.3.2 Angular Boundary Conditions

Because of the geometry of the near-wellbore reservoir, another boundary condition must be applied. In the angular direction, the first grid block $j = 1$ is connected with the last block $j = M$ in each layer. This means, the angular grid block j reaches $j = M$, and $j + 1$ coincides with $j = 1$. Similarly, the angular grid block j reaches $j = 1$, and $j - 1$ coincides with $j = M$.

3.3.3 Solution of the System of Linear Equations

The pressure equations for the 2D case will define a system of linear equations (Equation 3.49). Using the example of a 3×4 grid, the linear equations will be the following Equation 3.55 below.

The system of equations were implemented in MATLAB ® (Memorial University license) to solved the Laplacian. The pressure for each grid block in the Polar coordinate system is then known. The inverse matrix solution is:

$$p = A^{-1} D, \tag{3.54}$$

where p is a vector of unknown grid block pressures. The matrix A is the coefficient matrix, representing the inter-block permeabilities, and D is the vector containing the boundary condition.

$$
\begin{pmatrix} p_{1,1} \\ p_{1,2} \\ p_{1,3} \\ p_{1,4} \\ p_{2,1} \\ p_{2,2} \\ p_{2,3} \\ p_{2,4} \\ p_{3,1} \\ p_{3,2} \\ p_{3,3} \\ p_{3,4} \end{pmatrix}
=
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
d_{2,1} & 0 & 0 & 0 & a_{2,1} & b_{2,1} & 0 & c_{2,1} & e_{2,1} & 0 & 0 & 0 \\
0 & d_{2,2} & 0 & 0 & c_{2,2} & a_{2,2} & b_{2,2} & 0 & 0 & e_{2,2} & 0 & 0 \\
0 & 0 & d_{2,3} & 0 & 0 & c_{2,3} & a_{2,3} & b_{2,3} & 0 & 0 & e_{2,3} & 0 \\
0 & 0 & 0 & d_{2,4} & b_{2,4} & 0 & c_{2,4} & a_{2,4} & 0 & 0 & 0 & e_{2,4} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}^{-1}
\begin{pmatrix} p_w \\ p_w \\ p_w \\ p_w \\ 0 \\ 0 \\ 0 \\ 0 \\ p_e \\ p_e \\ p_e \\ p_e \end{pmatrix}
\tag{3.55}
$$

3.4 Semi-Analytical Streamline Generation in 2D Polar Coordinate Systems

We will next describe the generation of streamlines in two situations: homogeneous reservoir in **3.4.1** and heterogeneous reservoir in **3.4.2**, both in anisotropic reservoirs. The determination of the corner pressures from pressures in grid block centers is presented in **3.4.3**. The pressure analyses are then described in **3.4.4**.

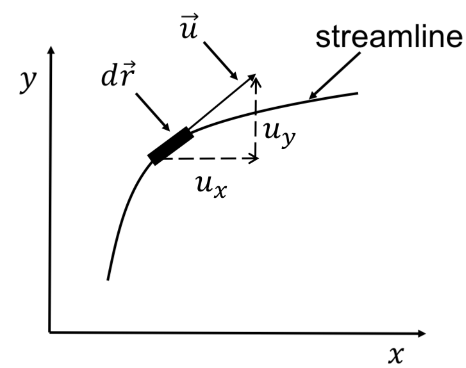


Figure 3.6: Relationship between Streamline and Velocity in Planar Flow

Streamlines are curves that are instantaneously tangent to the velocity vector of the flow, *i.e.* streamlines are integral curves that are locally tangential to a given velocity field at a given instant in time. The streamline construction is illustrated in two dimensions in Figure 3.6. The vector $\vec{u}$ is the velocity vector, u_x and u_y are the directional components of the flow velocity vector, $d\vec{r}$ is the infinitesimal arc length of the streamline.

According to its definition, the slope of the streamline at any point is given by the ratio of the components of the velocity at a given instant of time. In two-dimensional

Cartesian coordinates, a streamline can be defined by:

$$\vec{y\prime}(x) = \frac{[1, y'(x)]}{\sqrt{1 + y'(x)^2}} = \frac{[u_x, u_y]}{\sqrt{u_x^2 + u_y^2}}, \tag{3.56}$$

As for the Cartesian case, a streamline in the polar coordinate system must satisfy:

$$\vec{r\prime}(\theta) = \frac{[1, lnr'_D(\theta)]}{\sqrt{1 + lnr'_D(\theta)^2}} = \frac{[u_\theta, u_r]}{\sqrt{u_\theta^2 + u_r^2}}. \tag{3.57}$$

Streamlines can also be written in the parametric form:

$$dt = \frac{dr}{u_r(r, \theta, z)} = \frac{d\theta}{u_\theta(r, \theta, z)}. \tag{3.58}$$

In a polar coordinate system, assume that pressures at the four corners of a grid block are determined by an accurate algorithm after solving the Laplacian equation. The method to obtain the four corner pressures will be described in the **3.4.3**. These corner pressures (p_1, p_2, p_3, and p_4) are shown in Figure 3.7.

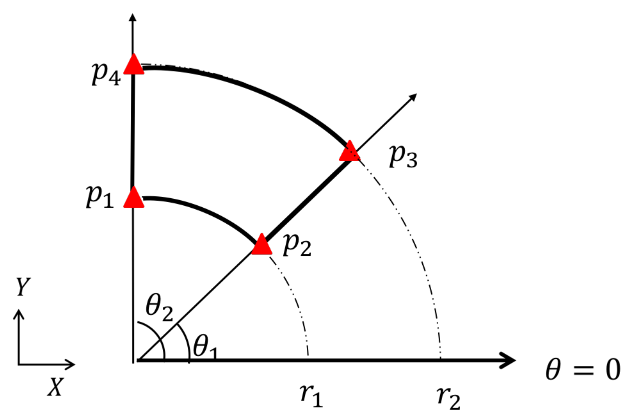


Figure 3.7: Grid Block with Corner Pressure in 2D

Assume the pressure changes linearly with lnr_D in radial direction and linearly in the θ direction. Consider a log-lin pressure distribution within a grid block, *i.e.*:

$$p(r_D, \theta) = a\theta lnr_D + b\theta + clnr_D + d, \tag{3.59}$$

where $r_D = r/r_w$, r is the radius for any location within this grid block and r_w is the wellbore radius. The function p in Equation 3.59 satisfies the Laplacian 3.36. From classical theory of PDE, this solution is unique. For any given grid block, the coefficients a, b, d, and d in Equation 3.59 are determined from a linear 4×4 system given by the pressures at grid block vertexes:

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} \theta_2 lnr_{D1} & \theta_2 & lnr_{D1} & 1 \\ \theta_1 lnr_{D1} & \theta_1 & lnr_{D1} & 1 \\ \theta_1 lnr_{D2} & \theta_1 & lnr_{D2} & 1 \\ \theta_2 lnr_{D2} & \theta_2 & lnr_{D2} & 1 \end{pmatrix}^{-1} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix}. \tag{3.60}$$

As previously mentioned, the slope of a streamline at any point is defined by the ratio of the components of the velocity. From Darcy's Law and Equation 3.59, velocities can be expressed as:

$$u_r = -\frac{1}{\mu}\left(K_r \frac{\partial p}{\partial r}\right) = -\frac{K_r}{\mu}\frac{\partial}{\partial r}[a\theta lnr_D + b\theta + clnr_D + d] = -\frac{K_r}{\mu r}(a\theta + c), \tag{3.61}$$

$$u_\theta = -\frac{1}{\mu}\left(\frac{1}{r} K_t \frac{\partial p}{\partial \theta}\right) = -\frac{K_t}{\mu r}\frac{\partial}{\partial \theta}[a\theta lnr_D + b\theta + clnr_D + d] = -\frac{K_t}{\mu r}(alnr_D + b). \tag{3.62}$$

Here, u_r and u_θ are Darcy velocities in the radial and angular directions, respectively; μ is the fluid viscosity, K_r and K_t are the permeabilities in the radial and angular directions, respectively. They can be calculated from Equation 3.26 and 3.27.

The time-of-flight depends on the real velocity rather than the Darcy velocity. The relationship between the real velocity and the Darcy velocity is:

$$v = \frac{u}{\phi}, \tag{3.63}$$

where u is the Darcy velocity and ϕ is the porosity.

3.4.1 Streamline Generation Method for Homogeneous Reservoirs

For a homogeneous reservoir, pressure decreases in a logarithmic fashion in the radial direction towards the wellbore and is constant as a function of θ. Hence, in Equation 3.59, $alnr_D + b = 0$. In this case, streamlines are straight lines towards the wellbore which can be expressed as:

$$\theta = \theta_{en} \qquad r \in (r_{en}, r_{ex}), \tag{3.64}$$

where θ_{en} is the angle for the entry point; r_{en} and r_{ex} are the radius for the entry point and the exit point, respectively.

A homogeneous reservoir is mathematically a special case of a heterogeneous reservoir in the near-wellbore region. The coefficients a and b are equal to 0 for a homogeneous reservoir, hence, from Equations 3.61 and 3.62, the velocities become:

$$u_r = -\frac{K_r c}{\mu r}, \tag{3.65}$$

$$u_\theta = 0. \tag{3.66}$$

Then, the real velocity in radial direction is:

$$v_r = \frac{u_r}{\phi} = \frac{K_r c}{\phi \mu r}. \tag{3.67}$$

The time-of-flight for this streamline is defined by the travel distance in the radial direction and the real velocity in the radial direction:

$$TOF = \int_{r_{en}}^{r_{ex}} \frac{dr}{|v_r|} = \phi\mu \int_{r_{ex}}^{r_{en}} \frac{rdr}{K_r c} = \frac{\phi\mu(r_{en}^2 - r_{ex}^2)}{2K_r c}. \tag{3.68}$$

3.4.2 Streamline Generation Method for Heterogeneous Reservoirs

For a heterogeneous reservoir, pressure changes in the angular direction. Hence we assume that $alnr_D + b \neq 0$ in Equation 3.59. If $alnr_D + b = 0$, the homogeneous reservoir streamline generation method in **3.4.1** is applied. Similarly to the Cartesian case, we find that any streamline must satisfy:

$$\frac{dr}{d\theta} = \frac{1}{r}\frac{u_r}{u_\theta} = \frac{1}{r}\frac{K_r(a\theta + c)}{K_t(alnr_D + b)}, \tag{3.69}$$

which, when integrated becomes:

$$\int rK_t(alnr_D + b)dr = \int K_r(a\theta + c)d\theta, \tag{3.70}$$

$$(\tilde{a}lnr_D + \tilde{b})^2 - (\tilde{c}\theta + \tilde{d})^2 = C, \tag{3.71}$$

where $\tilde{a} = a\sqrt{K_t}; \tilde{b} = b\sqrt{K_t}; \tilde{c} = a\sqrt{K_r}$ and $\tilde{d} = c\sqrt{K_r}$. The value of C is constant for a streamline inside a grid block. Every point on the streamline must satisfy Equation 3.71. We can calculate the constant C from the entry point in this grid block:

$$C = (\tilde{a}lnr_{Den} + \tilde{b})^2 - (\tilde{c}\theta_{en} + \tilde{d})^2. \tag{3.72}$$

According to Equations 3.61 and 3.62, we get:

$$a\theta + c = -\frac{u_r \mu r}{K_r}, \tag{3.73}$$

$$alnr_D + b = -\frac{u_\theta \mu r}{K_t}. \tag{3.74}$$

In order to express velocity in the radial direction by radial coordinate only, we substitute $\tilde{c} = a\sqrt{K_r}$ and $\tilde{d} = c\sqrt{K_r}$ into Equation 3.71 and get:

$$(\tilde{a}lnr_D + \tilde{b})^2 - (a\theta + c)^2 K_r = C. \tag{3.75}$$

In order to express velocity in the angular direction by angular coordinate only, we substitute $\tilde{a} = a\sqrt{K_t}$and $\tilde{b} = b\sqrt{K_t}$ into Equation 3.71:

$$(alnr_D + b)^2 K_t - (\tilde{c}\theta + \tilde{d})^2 = C. \tag{3.76}$$

Substituting Equation 3.73 and 3.74 into Equation 3.75 and 3.76 above gives:

$$(\tilde{a}lnr_D + \tilde{b})^2 - \frac{u_r^2(\mu r)^2}{K_r} = C, \tag{3.77}$$

$$\frac{u_\theta^2(\mu r)^2}{K_t} - (\tilde{c}\theta + \tilde{d})^2 = C. \tag{3.78}$$

Hence, the absolute value for Darcy velocities in the radial and angular directions can be written as:

$$|u_r| = \frac{\sqrt{K_r\left[(\tilde{a}lnr_D + \tilde{b})^2 - C\right]}}{\mu r}, \tag{3.79}$$

$$|u_\theta| = \frac{\sqrt{K_r\left[(\tilde{c}\theta + \tilde{d})^2 + C\right]}}{\mu r}. \tag{3.80}$$

Then, the real velocities are:

$$|v_r| = \frac{u_r}{\phi} = \frac{\sqrt{K_r\left[(\tilde{a}lnr_D + \tilde{b})^2 - C\right]}}{\phi\mu r}, \tag{3.81}$$

$$|v_\theta| = \frac{u_\theta}{\phi} = \frac{\sqrt{K_r\left[(\tilde{c}\theta + \tilde{d})^2 + C\right]}}{\phi\mu r}. \tag{3.82}$$

If $C < 0$, the streamline is a hyperbola in the $(\theta, lnr_D)-$space. The explicit formula for the streamline is:

$$\theta = -\frac{\tilde{d}}{\tilde{c}} + \frac{n}{\tilde{c}}\sqrt{(\tilde{a}lnr_D + \tilde{b})^2 - C}, \tag{3.83}$$

where $n = \pm 1$ and is determined from $\tilde{a}lnr_{Den} + \tilde{b} = n\sqrt{C + (\tilde{c}\theta_{en} + \tilde{d})^2}$. The time-of-flight is given by:

$$TOF = \int_{r_{ex}}^{r_{en}} \frac{dr}{|v_r|} = \phi\mu \int_{r_{ex}}^{r_{en}} \frac{rdr}{\sqrt{K_r\left[(\tilde{a}ln_D + \tilde{b})^2 - C\right]}}. \tag{3.84}$$

If $C > 0$, the streamline is a hyperbola in the (lnr_D, θ)-space. The explicit expression for the streamline is:

$$lnr_D = -\frac{\tilde{b}}{\tilde{a}} + \frac{n}{\tilde{a}}\sqrt{K_t\left[(\tilde{c}\theta + \tilde{d})^2 + C\right]}, \tag{3.85}$$

with $n = \pm 1$ and the time-of-flight for this streamline is:

$$TOF = \int_{\theta_{en}}^{\theta_{ex}} \frac{d\theta}{|v_\theta|} = \phi\mu \int_{\theta_{en}}^{\theta_{ex}} \frac{r(\theta)d\theta}{\sqrt{K_t\left[(\tilde{c}\theta + \tilde{d})^2 + C\right]}}. \tag{3.86}$$

We can also use numerical method (Runge-Kutta fourth order method) to determine the streamline path by using Equations 3.61 and 3.62:

$$\frac{u_r}{u_\theta} = \frac{K_r(a\theta + c)}{K_t(alnr_D + b)}. \tag{3.87}$$

3.4.3 Determination of Corner Pressures

Pressure nodes obtained by solving the discrete Laplace equation are located in the logarithmic center of each grid block; however, the semi-analytical streamline generation method relies on knowing the pressure in the corners of the grid block. In Figure 3.8, the •-nodes are the original pressure nodes $p_o(i,j)$ we obtained by using the finite difference method to solve the Laplacian; the ▲-nodes are the pressure points (p_1, p_2, p_3, and p_4) needed for the center grid block (blue grid block) in the semi-analytical method. In this section, the method to calculate the pressure in the corners of a grid block is described.

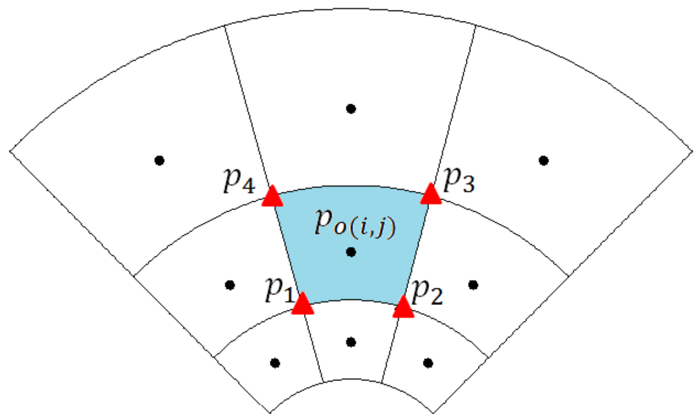
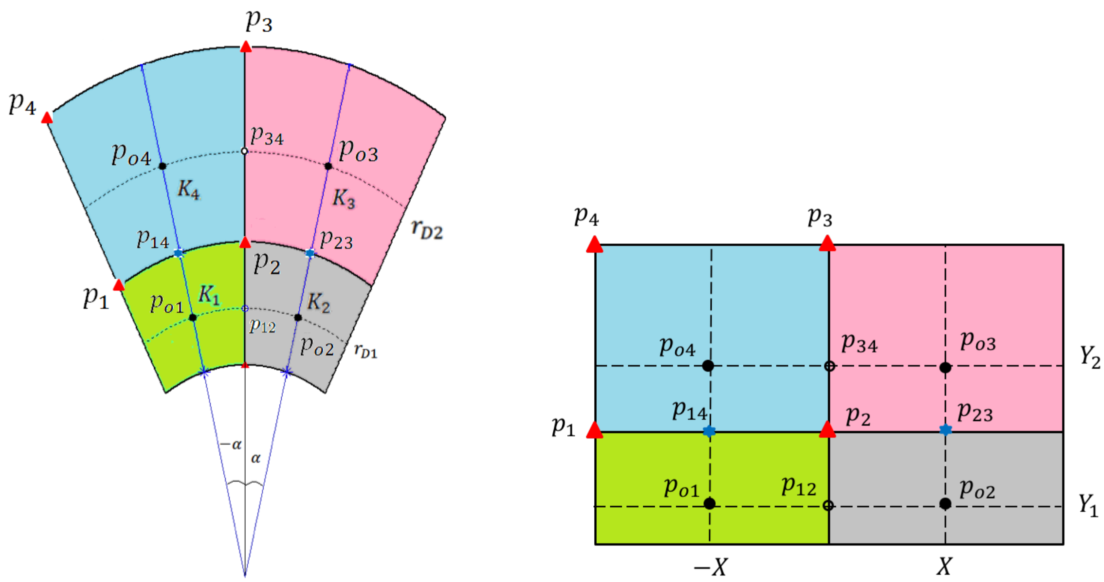


Figure 3.8: Grid Shifting of Pressure Distribution in 2D

The grid blocks used in the semi-analytical method are the same grid blocks used to solve the Laplace equation. Instead of using the pressure nodes in the center of each grid block, pressure points at the four corners are utilized in the streamline simulation. In order to generate the streamlines by the present method, corner pressures need to be determined. To achieve this, we impose three principles in an incompressible and source free system where gravity and capillary effects are negligible:

1. Flux continuity across each grid block boundary;
2. Pressure continuity across each grid block boundary;
3. Mass conservation over the control volume bounded by the pressure nodes (p_{o1}, p_{o2}, p_{o3}, and p_{o4}).

A 4×4 grid system is used to demonstrate how to apply these three principles to obtained the corner pressure, see Figure 3.9.



(a) Radial Grid Blocks

(b) Transformed Grid Blocks

Figure 3.9: Grid blocks for Pressure Distribution Calculation in 2D

A transform is used to perform the pressure calculation:

$$lnr_D = y, \quad \theta = x. \tag{3.88}$$

Then, the pressure $p_i(r_D, \theta)$ in Equation 3.59 is:

$$p_i = a_i xy + b_i x + c_i y + d_i. \tag{3.89}$$

In Figure 3.9, four pressure nodes p_{oi}, $i = 1, 2, 3, 4$, are calculated from the Laplacian for grid block 1, 2, 3, and 4. The radius r and angle θ are known, therefore, the transform can be written as:

$$lnr_{D(1,2)} = lnr_{D1} = Y_1, \quad lnr_{D(3,4)} = lnr_{D2} = Y_2, \tag{3.90}$$

$$\theta_{(1,4)} = -\alpha = -X, \quad \theta_{(2,3)} = \alpha = X. \tag{3.91}$$

where X, Y_1 and Y_2 refer to the transformed nodes coordinates shown in Figure 3.9.

The pressure at the interface between grid block 1 and 2 can be calculated from the flux continuity equation (Principle 1 above):

$$\int_{y_l}^{y_u} K_{t1}\frac{\partial p}{\partial x}dy = \int_{y_l}^{y_u} K_{t2}\frac{\partial p}{\partial x}dy, \tag{3.92}$$

where K_{t1} and K_{t2} are the permeability in the angular direction of grid blocks 1 and 2, respectively. Here, u and l are the upper and lower integration limit, respectively. By developing this expression, we find:

$$p_{12} = \frac{K_{t1}p_{o1} + K_{t2}p_{o2}}{K_{t1} + K_{t2}}, \tag{3.93}$$

where p_{12} is the pressure for the half distance point between the pressure node 1 and 2.

For the interface between grid block 1 and 4:

$$\int_{x_l}^{x_u} K_{r1}\frac{\partial p}{\partial y}dx = \int_{x_l}^{x_u} K_{r2}\frac{\partial p}{\partial y}dx, \tag{3.94}$$

i.e.

$$p_{14} = \frac{K_{r4}p_{o4}Y_2 + K_{r1}p_{o1}Y_1}{K_{r4}Y_2 + K_{r1}Y_1}, \tag{3.95}$$

where p_{14} is the pressure for the logarithmic center between the pressure node 1 and 4; K_{r1} and K_{r4} are the permeability in the radial direction of grid blocks 1 and 4, respectively.

Similarly, we can calculate the value for p_{23} and p_{34}, *i.e.*

$$p_{34} = \frac{K_{t3}p_{o3} + K_{t4}p_{o4}}{K_{t3} + K_{t4}}, \tag{3.96}$$

$$p_{23} = \frac{K_{r2}p_{o2}Y_1 + K_{r3}p_{o3}Y_2}{K_{r2}Y_1 + K_{r3}Y_2}. \tag{3.97}$$

In order to solve the parameters in the linear pressure assumption, 13 equations are needed. At this point, we can write 12 equations (Equation 3.98 to 3.109 below) for the 8 known pressure points. For the 4 finite difference method pressure nodes, 4 equations can be obtained:

$$p_{o1} = -a_1XY_1 - b_1X + c_1Y_1 + d, \tag{3.98}$$

$$p_{o2} = a_2XY_1 - b_2X + c_2Y_1 + d, \tag{3.99}$$

$$p_{o3} = a_3XY_2 + b_3X + c_3Y_2 + d, \tag{3.100}$$

$$p_{o4} = -a_4XY_2 + b_4X + c_4Y_2 + d. \tag{3.101}$$

Pressures on the interfaces satisfy the pressure assumptions in both blocks. Hence, for each point two equations can be obtained (Principle 2 above):

$$p_{12} = c_1Y_1 + d, \tag{3.102}$$

$$p_{12} = c_2Y_1 + d, \tag{3.103}$$

$$p_{23} = b_2X + d, \tag{3.104}$$

$$p_{23} = b_3X + d, \tag{3.105}$$

$$p_{34} = c_3Y_2 + d, \tag{3.106}$$

$$p_{34} = c_4Y_2 + d, \tag{3.107}$$

$$p_{14} = -b_1X + d, \tag{3.108}$$

$$p_{14} = -b_4X + d. \tag{3.109}$$

We need one more equation. This last equation is the material balance equation for steady state flow over the control volume defined by the grid block bounded by the four known pressures $p_{o1}, p_{o2}, p_{o3}, p_{o4}$ (Principle 3 above), *i.e.*

$$\sum_{i=1}^{8} q_i = 0, \tag{3.110}$$

where q_i is the flow rate in the radial and angular direction at the boundaries, *i.e.*

$$q_1 = K_{r1} \int_{-X}^{0} \frac{\partial p_1}{\partial y} dx = K_{r1} \left(-\frac{a_1X^2}{2} + c_1X \right), \tag{3.111}$$

$$q_2 = K_{r2} \int_0^X \frac{\partial p_2}{\partial y} dx = K_{r2} \left(\frac{a_2 X^2}{2} + c_2 X \right), \tag{3.112}$$

$$q_3 = -K_{r3} \int_0^X \frac{\partial p_3}{\partial y} dx = -K_{r3} \left(\frac{a_3 X^2}{2} + c_3 X \right), \tag{3.113}$$

$$q_4 = -K_{r4} \int_{-X}^0 \frac{\partial p_4}{\partial y} dx = -K_{r4} \left(-\frac{a_4 X^2}{2} + c_4 X \right), \tag{3.114}$$

$$q_5 = -K_{t1} \int_{Y_1}^0 \frac{\partial p_1}{\partial x} dy = -K_{t1} \left(\frac{-a_1 Y_1^2}{2} - b_1 Y_1 \right), \tag{3.115}$$

$$q_6 = K_{t2} \int_{Y_1}^0 \frac{\partial p_2}{\partial x} dy = K_{t2} \left(\frac{-a_2 Y_1^2}{2} - b_2 Y_1 \right), \tag{3.116}$$

$$q_7 = K_{t3} \int_{Y_2}^0 \frac{\partial p_3}{\partial x} dy = K_{t3} \left(\frac{a_3 Y_2^2}{2} + b_3 Y_2 \right), \tag{3.117}$$

$$q_8 = -K_{t4} \int_{Y_2}^0 \frac{\partial p_4}{\partial x} dy = -K_{t4} \left(\frac{a_4 Y_2^2}{2} + b_4 Y_2 \right). \tag{3.118}$$

After numerous calculations we found that the 13 equations (Equation 3.98-3.109 and Equation 3.110) are linearly independent. This is because the principles are physically independent. Parameters a_i, b_i, c_i, and d for these four grid blocks are determined from a linear 13×13 system described above (Equation 3.119). The pressure for any point within these four grid blocks (x, y) can then be calculated by Equation 3.59, and the pressure profiles for these four grid blocks obtained. Extended to the entire reservoir, the pressure distribution needed for the present semi-analytical method is therefore known and streamlines can be generated.

$$
\begin{pmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \\ c_3 \\ a_4 \\ b_4 \\ c_4 \\ d \end{pmatrix}
=
\begin{pmatrix}
-XY_1 & -X & Y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & XY_1 & -X & Y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & XY_2 & X & Y_2 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -XY_2 & X & Y_2 & 1 \\
0 & 0 & Y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & Y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & X & 0 & 0 & 0 & 1 & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Y_2 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Y_2 & 1 \\
0 & -X & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -X & 0 & 1 & \\
A_1 & B_1 & C_1 & A_2 & B_2 & C_2 & A_3 & B_3 & C_3 & A_4 & B_4 & C_4 & 0
\end{pmatrix}^{-1}
\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_{12} \\ p_{12} \\ p_{23} \\ p_{23} \\ p_{34} \\ p_{34} \\ p_{14} \\ p_{14} \\ 0 \end{pmatrix} . \tag{3.119}
$$

In Equation 3.119 the coefficients are:

$$A_1 = -\frac{K_{r1}X^2}{2} - \frac{K_{t1}Y_1^2}{2}, \tag{3.120}$$

$$B_1 = -K_{t1}Y_1, \tag{3.121}$$

$$C_1 = K_{r1}X, \tag{3.122}$$

$$A_2 = \frac{K_{r2}X^2}{2} + \frac{K_{t2}Y_1^2}{2}, \tag{3.123}$$

$$B_2 = K_{t2}Y_1, \tag{3.124}$$

$$C_2 = K_{r2}X, \tag{3.125}$$

$$A_3 = -\frac{K_{r3}X^2}{2} - \frac{K_{t3}Y_2^2}{2}, \tag{3.126}$$

$$B_3 = -K_{t3}Y_2, \tag{3.127}$$

$$C_3 = -K_{r3}X, \tag{3.128}$$

$$A_4 = \frac{K_{r4}X^2}{2} + \frac{K_{t4}Y_2^2}{2}, B_4 = K_{t4}Y_2, \tag{3.129}$$

$$C_4 = -K_{r4}X. \tag{3.130}$$

3.4.4 Pressure Analysis for the 2D Streamline Simulation

The pressure assumption discussed in this section is the pressure distribution within each grid block i, j given by Equation 3.59. We demand three principles for the pressure distribution:

1. Flux continuity across each grid block boundary;

2. Pressure continuity across each grid block boundary;

3. Satisfy Laplace equation at each point inside grid blocks.

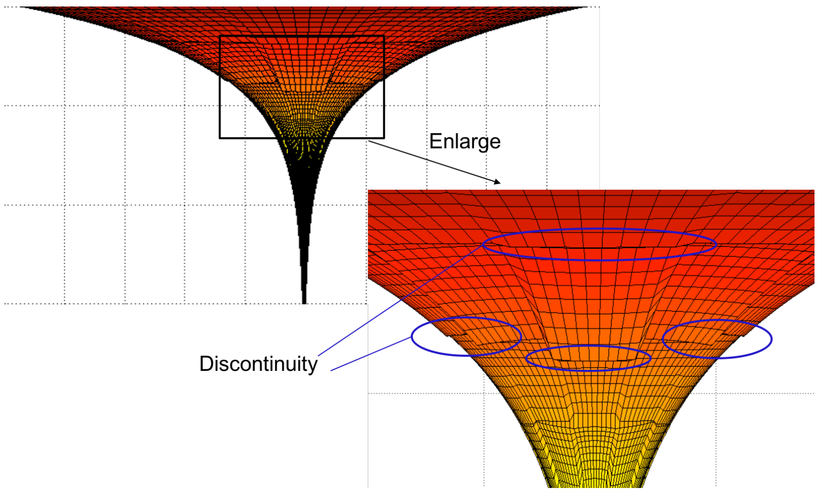
There are some high degree pressure polynomials that satisfy Laplace equation but they are not practical in the near-wellbore region. It is complicated and computationally expensive to find the pressure solution for high degree pressure polynomials. For these reasons, in this research thesis, we ignore such high degree pressure assumptions and focus on the classical Pollock's pressure assumption (piecewise constant), and our piecewise log-lin pressure assumption.

First, we will discuss the feasibility of the Principle 1 above.

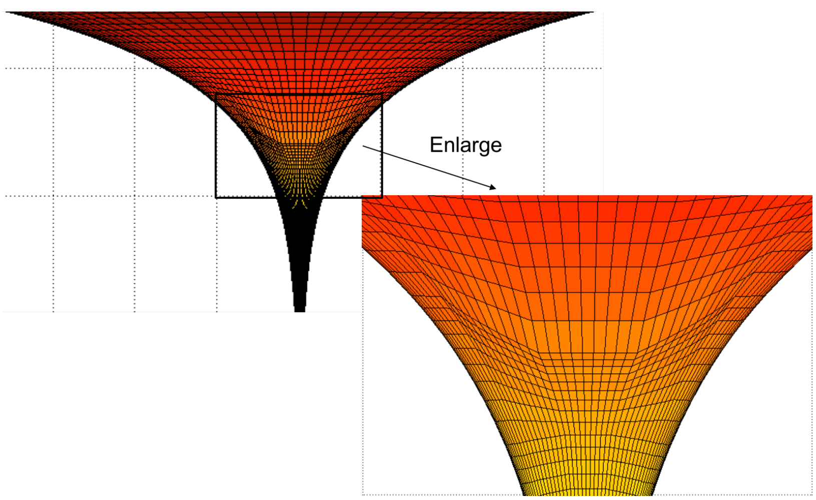
In Pollock's method, the velocities across the simulation grid block boundaries are calculated by using Darcy's law. This satisfies the flux continuity across each grid block interface. For the piecewise log-lin pressure assumption, as described in **3.4.3**, the principle of the flux continuity across each grid block interface is also satisfied.

Next, we will discuss the Principle 2 above.

For homogeneous cases, both Pollock's pressure assumption and the piecewise log-lin pressure assumption are continuous across the grid block boundaries, *i.e.* satisfies Principle 2. For heterogeneity cases, Pollock's pressure assumption cannot satisfy the pressure continuity principle. The velocity field calculated from the finite difference method is used in combination with Equations 3.133 and 3.132 to calculate the velocities in the angular and radial directions. The pressure distribution for each grid block can be determined by Darcy's velocity (Equation 3.131). Pressure distributions for the present method and Pollock's method are shown in Figure 3.10. The ratio of permeability between the heterogeneity and the bulk reservoir is 1/10. As we can see from Figure 3.10, there are some pressure discontinuities across the grid boundaries for Pollock's pressure distribution. In contrast to Pollock's method and as demonstrated pressure distribution for the present method is smooth and continuous. As described in **3.4.3**, the determination of the corner pressure ensures the global pressure is continuous over grid block boundaries for the piecewise log-lin pressure assumption.



(a) Pollock's Pressure Distribution



(b) Pressure Distribution for the Present Streamline Method

Figure 3.10: Enlarged Pressure Field for the Present Method and Pollock's Method

Therefore, Pollock's method cannot be used in coarse grids without potentially introducing large errors.

Finally, we will discuss the Principle 3 above.

As described in **Chapter 2**, in the Pollock's streamline method, the angular velocity u^θ is assumed to vary linearly in $\theta-$direction within each grid block and the radial velocity u^r is assumed to increase as the inverse of the radius, see Equations 2.12 and 2.13. According to Darcy's law, velocity is proportional to the pressure gradient. The velocity assumption used in Pollock's method is therefore equivalent to assuming a pressure function given in Equation 3.131 within each grid block:

$$p(r,\theta) = A(lnr) + B\theta^2 + Cr + D\theta + E. \tag{3.131}$$

This is incompatible with having a constant pressure within each grid block as assumed by the boundary velocity calculation in Pollock's method. Then, the velocities are given by:

$$u_r = -\frac{K_r}{\mu}\frac{\partial p}{\partial r} = -\frac{K_r}{\mu}\left(\frac{A}{r} + C\right), \tag{3.132}$$

$$u_\theta = -\frac{K_t}{\mu r}\frac{\partial p}{\partial \theta} = -\frac{K_t}{\mu r}(2B\theta + D). \tag{3.133}$$

For an isotropic reservoir, $K_r = K_t = K$. As in Equation 3.131:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(rK\frac{\partial p}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K\frac{\partial p}{\partial \theta}\right) = \frac{KC}{r} + \frac{2KB}{r^2}. \tag{3.134}$$

Therefore, Pollock's pressure assumption cannot satisfy the Laplacian for 2D isotropic porous media (Equation 3.135).

The second partial derivatives of the log-lin pressure is identically equal to 0, which means the piecewise log-lin pressure assumption satisfies the Laplacian for 2D isotropic porous media,

$$\frac{1}{r}\frac{\partial}{\partial r}\left(rK\frac{\partial p}{\partial r}\right)+\frac{1}{r^2}\frac{\partial}{\partial \theta}\left(K\frac{\partial p}{\partial \theta}\right)=0. \tag{3.135}$$

Hence, we can conclude that the piecewise log-lin pressure assumption satisfies Principle 3 listed above. The Pollock's pressure assumption cannot satisfy Principle 3 listed above.

Table 3.1 summarizes the imposed principles for the two pressure distributions discussed. As we have shown, the log-lin pressure assumption is more accurate and can also be applied in the 3D case.

Table 3.1: Summary of Demanded Principles

Pressure Assumption	**Flux Continuity Across Grid Blocks Boundaries (1)**	**Pressure Continuity Across Grid Blocks Boundaries (2)**	**Satisfy Laplace Equation (3)**
Pollock's Pressure Assumption	Yes	No	No
Bi-linear Pressure Assumption	Yes	Yes	Yes

3.5 Semi-Analytical Streamline Generation in 3D Cylindrical Coordinate Systems

3.5.1 Streamline Generation Method

For a 3D cylindrical grid block shown in Figure 3.11, the pressure is assumed to change linearly on each edge of the box in the $z-$ and $\theta-$direction, while logarithmically in the $r-$direction. Following Johansen (2010) the bi-linear logarithmic (bilin-log) pressure function is:

$$p(r_D, \theta, z) = a\theta lnr_D z + b\theta lnr_D + c\theta z + dlnr_D z + e\theta + flnr_D + gz + h. \tag{3.136}$$

This bilin-log pressure assumption satisfies the general Laplacian in 3D (Equation 3.35).

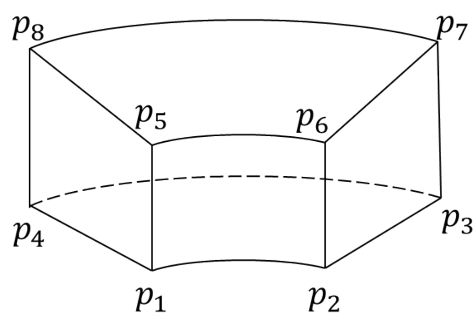


Figure 3.11: A Single Grid Block in 3D

The coefficients in Equation 3.136 are given by the 8 corner pressures. By taking derivatives according to Darcy's Law, the velocities are derived.

$$u_r = -\frac{1}{\mu}\left(K_r \frac{\partial p}{\partial r}\right) = -\frac{K_r}{\mu r}(a\theta z + b\theta + dz + f), \tag{3.137}$$

$$u_\theta = -\frac{1}{r\mu}\left(K_t \frac{\partial p}{\partial \theta}\right) = -\frac{K_t}{\mu r}(alnr_D z + blnr_D + cz + e), \tag{3.138}$$

$$u_z = -\frac{1}{\mu}\left(K_z \frac{\partial p}{\partial z}\right) = -\frac{K_z}{\mu}(a\theta lnr_D + c\theta + dlnr_D + g), \tag{3.139}$$

where K_r and K_t are defined in Equation 3.26 and 3.27, respectively; and K_z is the permeability in $z-$direction.

The streamlines are determined by a system of two ODEs as follows:

If $a\theta z + b\theta + dz + f \neq 0$, lnr_D is used as a parameterization for streamlines. The two ODEs can be then written as:

$$\frac{u_\theta}{u_r} = \frac{K_t(azlnr_D + blnr_D + cz + e)}{K_r(a\theta z + b\theta + dz + f)}, \tag{3.140}$$

$$\frac{u_z}{u_r} = \frac{K_z r(a\theta lnr_D + c\theta + dlnr_D + g)}{K_r(a\theta z + b\theta + dz + f)}. \tag{3.141}$$

If $azlnr_D + blnr_D + cz + e \neq 0$, θ is used as a parameterization for streamlines. The streamlines are then determined by a system of two ODEs:

$$\frac{u_r}{u_\theta} = \frac{K_r(a\theta z + b\theta + dz + f)}{K_t(azlnr_D + blnr_D + cz + e)}, \tag{3.142}$$

$$\frac{u_z}{u_\theta} = \frac{K_z r(a\theta lnr_D + c\theta + dlnr_D + g)}{K_t(azlnr_D + blnr_D + cz + e)}. \tag{3.143}$$

If $a\theta lnr_D + c\theta + dlnr_D + g \neq 0$, z is used as the parameterization. The two ODEs then are:

$$\frac{u_r}{u_z} = \frac{K_r(a\theta z + b\theta + dz + f)}{K_z r(a\theta lnr_D + c\theta + dlnr_D + g)}, \tag{3.144}$$

$$\frac{u_\theta}{u_z} = \frac{K_t(azlnr_D + blnr_D + cz + e)}{K_z r(a\theta lnr_D + c\theta + dlnr_D + g)}. \tag{3.145}$$

At least one of these situations will provide the streamline in each grid block since we do not consider a stagnation curve as a streamline. We may have to change parameterization when tracing the streamlines but the parameterization will not change within a grid block for a streamline. If more than one situation is true, we calculate the possible travel times for all situations and the time-of-flight is the minimum among them. Then the time-of-flight is used to determine the streamline path. The TOF for a streamline is given by:

$$TOF = min(t_r,\ t_\theta,\ t_z), \tag{3.146}$$

where

$$t_r = \phi \int_{r_{en}}^{r_{ex}} \frac{dr}{|u_r|}, \tag{3.147}$$

$$t_t = \phi \int_{\theta_{en}}^{\theta_{ex}} \frac{d\theta}{|u_\theta|}, \tag{3.148}$$

$$t_z = \phi \int_{z_{en}}^{z_{ex}} \frac{dz}{|u_z|}. \tag{3.149}$$

These integrals can be solved by numerical integration methods.

3.5.2 Determination of Corner Pressures

As with the two-dimensional case, pressure nodes calculated from the finite difference method are located in the logarithmic center of each grid block in the radial direction and in the half distance center of each grid block in the angular and the $z-$direction. In order to obtain the pressure in the $z-$direction, the point distributed grid structure is used in the $z-$direction. The grid blocks used in 3D are shown in Figure 3.12.

If we assume that the discretization in the angular and vertical directions is uniform, the general expression of the discretized Laplacian 3.35 in 3D is:

$$\begin{aligned} &a_{i,j,k}p_{i,j,k} + b_{i,j,k}p_{i,j+1,k} + c_{i,j,k}p_{i,j-1,k} + d_{i,j,k}p_{i-1,j,k} \\ &+e_{i,j,k}p_{i+1,j,k} + f_{i,j,k}p_{i,j,k-1} + g_{i,j,k}p_{i,j,k+1} = 0, \end{aligned} \tag{3.150}$$

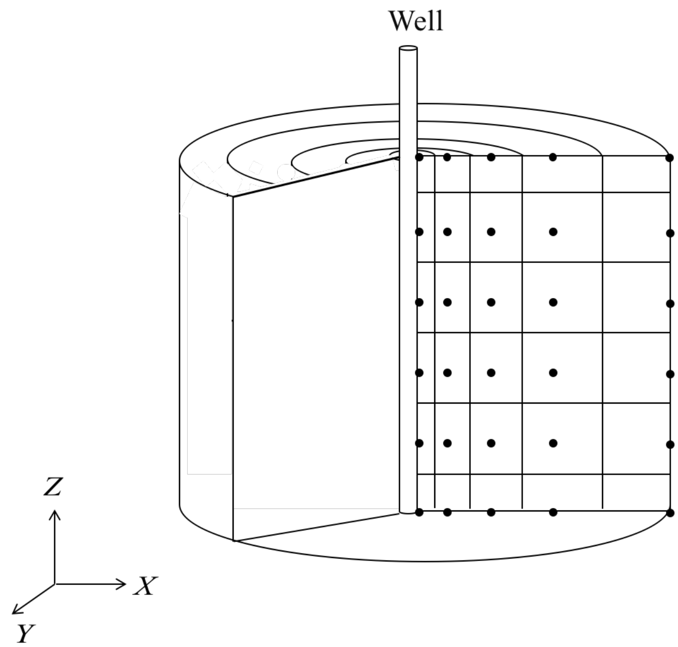


Figure 3.12: Pressure Nodes for 3D Grid Blocks

where the coefficients are :

$$a_{i,j,k} = -\Bigg[\left(\frac{r_{i+1/2,j,k} K_{i+1/2,j,k}}{r_{i,j,k}(r_{i+1/2,j,k} - r_{i-1/2,j,k})(r_{i+1,j,k} - r_{i,j,k})} \right)$$
$$+ \left(\frac{r_{i-1/2,j,k} K_{i-1/2,j,k}}{r_{i,j,k}(r_{i+1/2,j,k} - r_{i-1/2,j,k})(r_{i,j,k} - r_{i-1,j,k})} \right)$$
$$+ \left(\frac{K_{i,j+1/2,k} + K_{i,j-1/2,k}}{r_{i,j,k}^2 \Delta\theta^2} \right) + \left(\frac{K_{i,j,k+1/2} + K_{i,j,k-1/2}}{\Delta z^2} \right) \Bigg], \quad (3.151)$$

$$b_{i,j,k} = \left(\frac{K_{i,j+1/2,k}}{r_{i,j,k}^2 \Delta\theta^2} \right), \quad (3.152)$$

$$c_{i,j,k} = \left(\frac{K_{i,j-1/2,k}}{r_{i,j,k}^2 \Delta\theta^2} \right), \quad (3.153)$$

$$d_{i,j,k} = \left(\frac{r_{i-1/2,j,k} K_{i-1/2,j,k}}{r_{i,j,k}(r_{i+1/2,j,k} - r_{i-1/2,j,k})(r_{i,j,k} - r_{i-1,j,k})} \right), \quad (3.154)$$

$$e_{i,j,k} = \left(\frac{r_{i+1/2,j,k} K_{i+1/2,j,k}}{r_{i,j,k}(r_{i+1/2,j,k} - r_{i-1/2,j,k})(r_{i+1,j,k} - r_{i,j,k})} \right), \quad (3.155)$$

$$f_{i,j,k} = \left(\frac{K_{i,j,k-1/2}}{\Delta z^2} \right), \tag{3.156}$$

$$g_{i,j,k} = \left(\frac{K_{i,j,k+1/2}}{\Delta z^2} \right). \tag{3.157}$$

The pressure equation written for each grid point and the resulting linear equations can be expressed in matrix form as:

$$Ap = D. \tag{3.158}$$

The pressure matrix can be solved by the inverse of the system matrix:

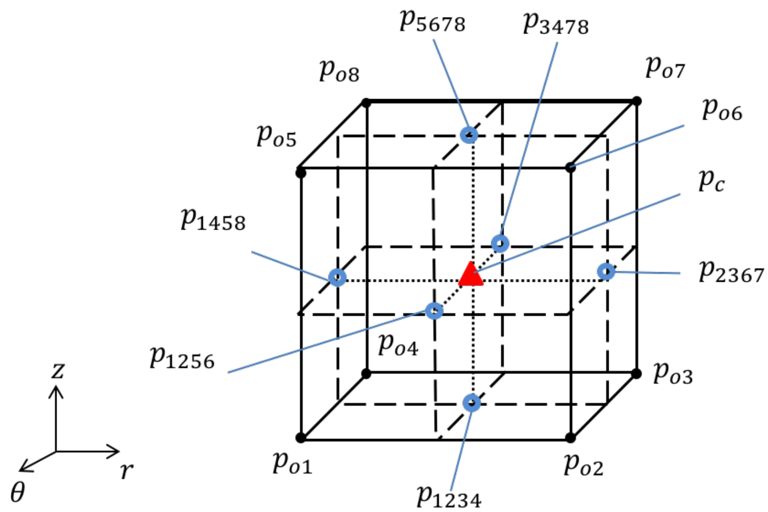
$$p = A^{-1}D, \tag{3.159}$$

where p is a vector of unknown grid block pressures. The matrix A is the coefficient matrix, representing the inter-block permeabilities, and D is the vector containing the boundary condition. The inverse matrix MATLAB ® code is used to obtain the solution of Equation 3.158.

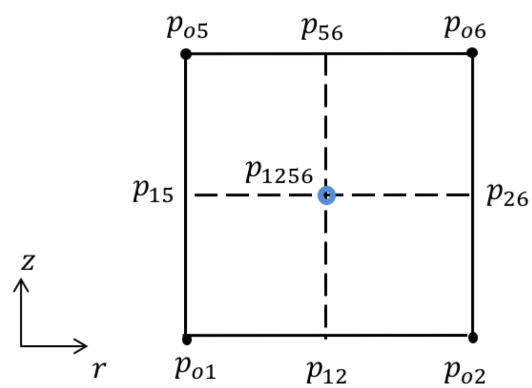
After obtaining the finite difference pressure nodes, the same transform in 2D (Equation 3.88) is used in the calculation to obtain the corner pressure nodes in 3D. Since the transform is used in the calculation, the grid block geometry is transformed into cubes. As illustrated in Figure 3.13, eight node pressures ($p_{oi}, i = 1, 2, 3, 4, 5, 6, 7, 8$) for 8 grid blocks are calculated from the finite difference method. Pressures for the logarithmic centers in the radial direction and half distance centers in the angular and $z-$direction such as $p_{12},\ p_{15},\ p_{56}$, and p_{26} shown in Figure 3.13 are calculated by the flux continuity principle (Principle 1 in **3.4.3**). *i.e.*

$$\int_{z_l}^{z_u} K_{z1} \frac{\partial p}{\partial r} dz = \int_{z_l}^{z_u} K_{z5} \frac{\partial p}{\partial r} dz, \tag{3.160}$$

where K_{z1} and K_{z5} are the permeability in the $z-$direction of grid blocks 1 and 5, respectively. Here, u and l are the upper and lower integration limit, respectively.



(a) 3D View



(b) Face View

Figure 3.13: Transformed 3D Grid Blocks

For an incompressible system, pressures for the points located in the center of the grid block faces created by any four node points ($p_{1234}, p_{2367}, p_{5678}, p_{1458}, p_{1256}$, and p_{3478}) and the corner point c (located in center of the cube in Figure 3.13) are then calculated by mass conservation over the control volume around the primal grid block (Principle 3 in **3.4.3**) as:

$$\sum q = 0. \tag{3.161}$$

For any system like this, $8\times7+1 = 57$ unknowns shown in Equation 3.162 ($a_i, b_i, ..g_i, i = 1, 2, ...8$, and h) are introduced:

$$p_i(r_D, \theta, z) = a_i\theta lnr_Dz + b_i\theta lnr_D + c_i\theta z + d_ilnr_Dz + e_i\theta + f_ilnr_D + g_iz + h, \tag{3.162}$$

where h is the pressure value p_c for corner point formed by these 8 grid blocks. We need 57 equations to compute p_c. Table 3.2 summarizes the equations that are used.

Table 3.2: Equations for Corner Pressure Calculation

Description	**Number of Equations**
Pressure equations for 8 node points	8
Pressure equations for 12 midpoints between two node points (Principles 2 in **3.4.3**: Each midpoint satisfies the pressure equation for 2 grid blocks)	24
Pressure equations for 6 points located in the center of the grid block faces created by any four node points (Principles 2 in **3.4.3**: Each point satisfies the pressure equation for 4 grid blocks)	24
Material balance equation for steady state flow over the control volume bounded by eight node points (Principles 3 in **3.4.3**)	1

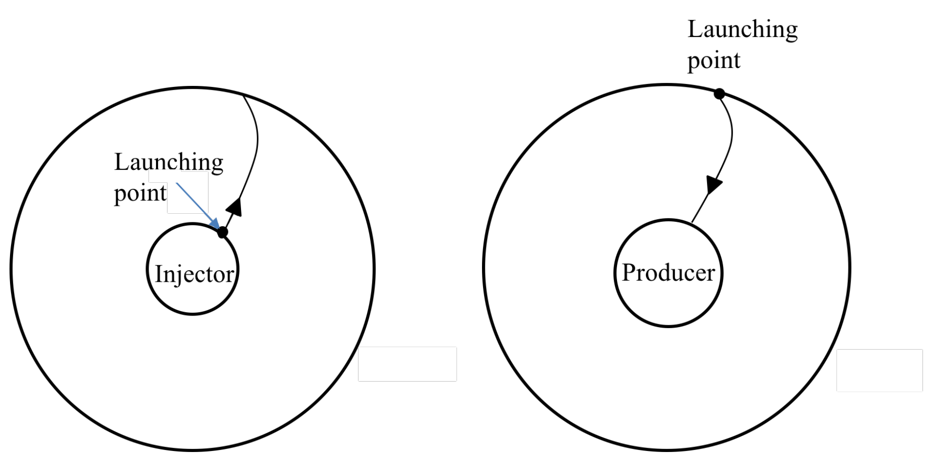
These 57 equations determine a linear 57×57 system, the solution of which is the corner pressures. Once the corner pressures are known, the pressure for any location within the grid blocks can be calculated by Equation 3.162. Extended to the entire porous media, the globally continuous pressure distribution at any location within the medium can be generated.

The principle discussed above is rigorous, however, for most cases we can use the averaging method to determine the corner pressures for simplification.

3.6 Streamline Tracing Procedure

Using the present semi-analytical streamline method described above (**Section 3.4**), we can trace a single streamline from injector to producer in the near-wellbore region as shown in the flow chart in Figure 3.15 and descried below:

1. Give a particle launching point. The launching point defines the initial space location of the particle. In the near-wellbore region, the launching point is located at the inner boundary for an injection well and located at the outer boundary for a production well (Figure 3.14).



(a) Injector Located in Center (b) Producer Well Located in Center

Figure 3.14: Schematic of Injection Wells and Production Wells

2. Consider the velocity for the given launching point. If the velocity equals to zero, stop tracing; and consider the next launching point.

3. Determine the grid block coordinates which the given launching (or entry) point belongs to.

4. Calculate the potential exit points within the grid block in 3.

5. Calculate the time-of-flight of the streamline as minimum of the travel time to the potential exit points in 4.

6. Determine the actual exit point by considering the $r-, \theta-, z-$directions independently, defined by the TOF calculated in 5.

7. Use this exit point as the entry point of the next grid block and calculate the coordinates for this new grid block.

8. Go back to step 2 for a new tracing process until the fluid particle reaches the other boundary.

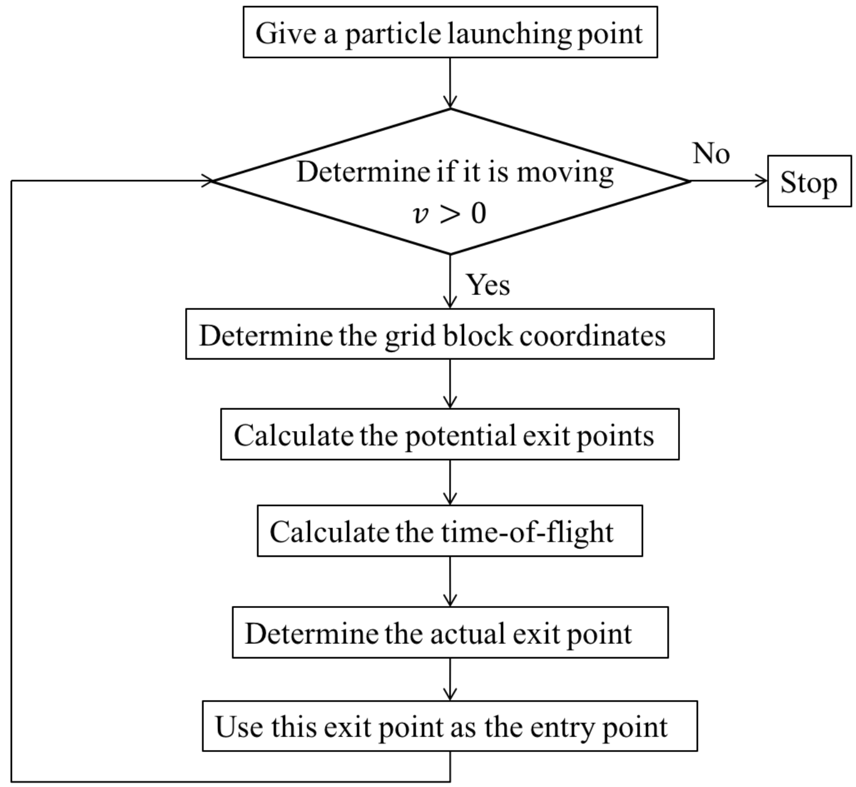
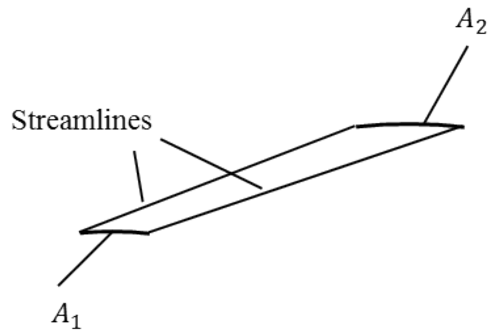
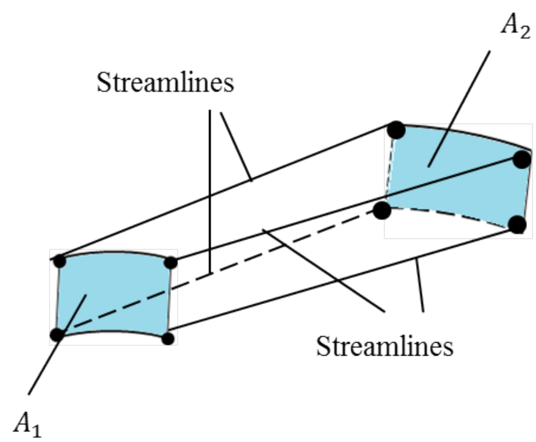


Figure 3.15: Flow Chart of Stream Tracing Procedure

As shown in Figure 3.16, in this research thesis, a stream tube is a tubular region in space bounded by two streamlines in 2D. In 3D, a stream tube is defined by four streamlines. In this research thesis, the starting points for these four streamlines are located at numerical layer interfaces. The streamline coordinates, cross section area and the streamline arc length are stored to be used when solving two-phase transport problems in **Chapter 4**.



(a) A 2D Stream Tube Defined by Two Streamlines



(b) A 3D Stream Tube Defined by Four Streamlines

Figure 3.16: Schematic of Stream Tubes Structure

Chapter 4

Applications and Case Studies in Near-Wellbore Regions

In the previous chapter, the method for determining the streamline paths in the near-wellbore region was presented. The application of streamline simulation is becoming standard for reservoir flow visualization, dynamic reservoir characterization, and optimal flood management. This chapter will discuss the utility of streamline simulation in the near-wellbore region with three main aspects: 1. Water flooding prediction; 2. Streamline modeling for open hole well completions; 3. Skin calculation for perforated wells.

4.1 Modeling Two-Phase Flow in Stream Tubes

Streamline models provide fast and accurate solutions to displacements even for strongly heterogeneous systems. The computational efficiency is due to the fact that the streamline simulation method decouples the full 3D problem into a set of multiple 1D problems along streamlines. Fluids move along the natural streamline grid rather than between discrete grid blocks as in conventional methods (Batycky, 1997). The fluid movement can be calculated by using Riemann solutions based on the fractional flow function.

Previous streamline simulation methods used the analytical 1D Riemann approach (Buckley and Leverett, 1942) to describe constant flow rate cases. A Riemann solution for waterflooding consists of a propagation of a smooth rarefaction wave trailing a shock front as in Figure 4.1. The propagation velocity is monotonically increasing from the injector to producer. The theory is briefly reviewed in **Appendix A**.

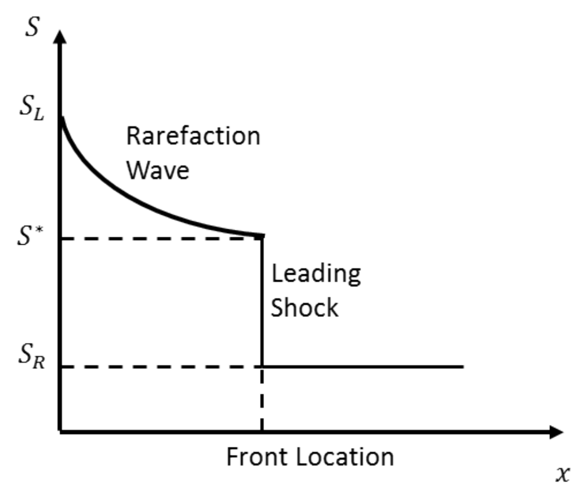


Figure 4.1: Analytical 1D Riemann Solution

The classical fractional-flow theory was under the assumption of constant flow rate. In a stream tube, the flow velocity in the tube is the flow rate divided by the cross section area of the stream tube, and for a given constant flow rate this can be used directly in the Riemann solution. However, if the boundary condition instead is specified as constant pressure it is no longer true that the flow rate is constant or even known a priori. Such constant pressure boundaries occur when the reservoir is operated under constant injection pressure and constant production pressure or as in laboratory experiments, under constant differential pressure. For the case of constant pressure boundaries, the flow rate is a function of time. Johansen, James (2015) and Johansen et al. (2016) determined the 1D Riemann solution for constant pressure boundaries. In a stream tube, the area is changing along the stream tube and the problem is not 1D. For a constant pressure boundary stream tube, it requires a 3D Riemann solution as determined in Johansen and Liu (2016) both before and after breakthrough of the front, and briefly described in **4.1.1**.

The application of streamline simulation to model two-phase flow involves five major steps. A flow chart for simulating two-phase flow using the Riemann approach along stream tubes is shown in Figure 4.2.

1. Input the geological and fluids information such as reservoir dimensions, permeability, porosity and viscosities.

2. Solve the Laplace equation by using the finite difference method and then determine the corner pressures as described in **3.4.3** to obtain the continuous profile for the entire reservoir.

3. Generate the streamlines for single-phase flow as described in **3.4.1** or **3.4.2**.

4. Bundle the neighboring streamlines into stream tubes and capture the stream tube information such as cross section area and stream tube length.

5. Map the 3D Riemann solution along stream tubes to simulate the fluid movement.

In this research thesis, streamlines are assumed constant. This means, streamlines are not updated with time. With an unfavorable mobility ratio or the change of well conditions, streamlines need to be updated frequently.

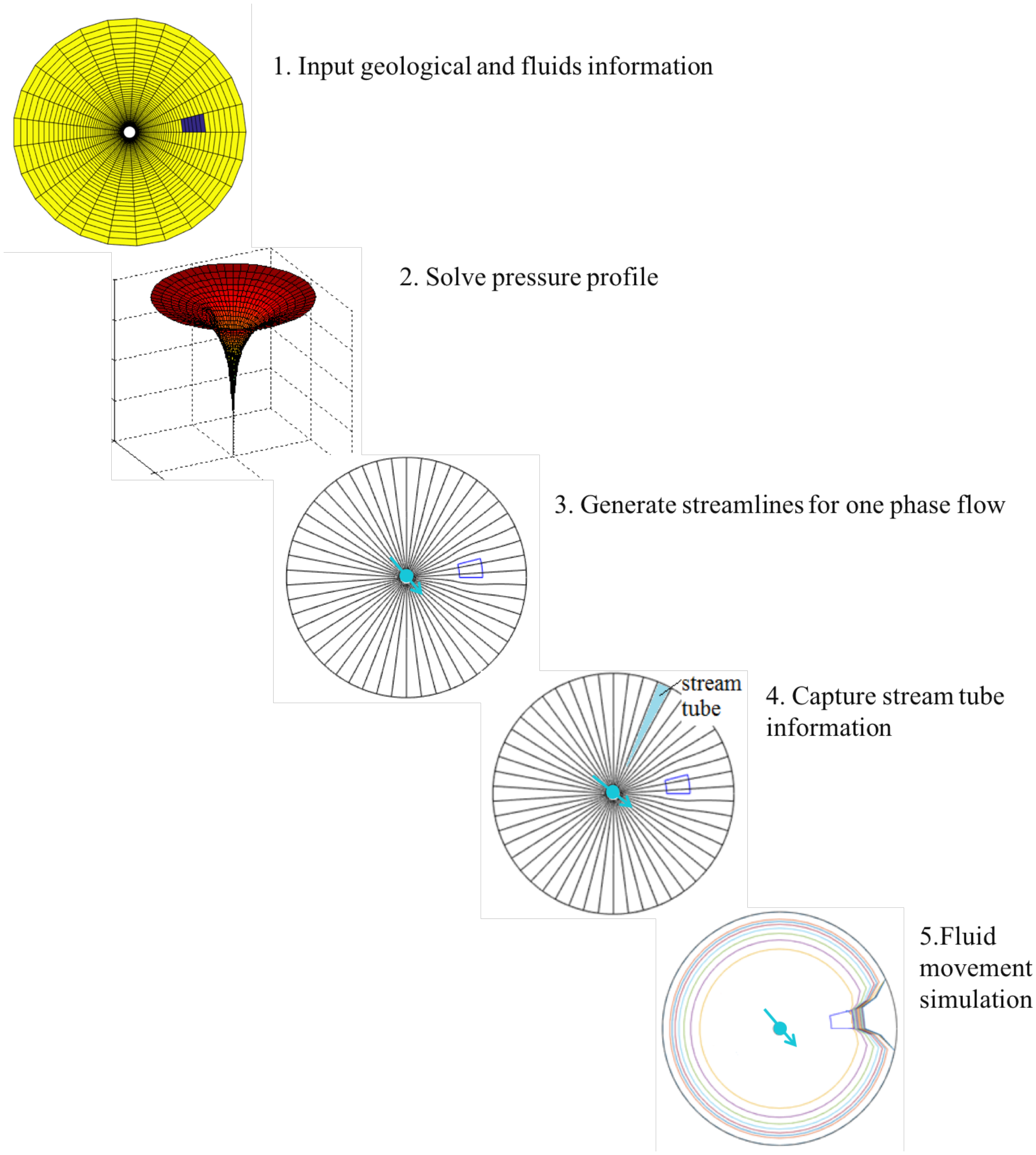


Figure 4.2: Flow Chart of Riemann Approach along Stream Tubes

Next, we will present a semi-analytical Riemann approach for constant pressure boundary used to move fluid numerically along stream tubes, which is described in Johansen and Liu (2016).

4.1.1 Solution of the Riemann Problems in Stream Tubes under Constant Pressure Boundaries

In a waterflooding process, in a near-wellbore region study, the injection well is located in the center of a cylindrical reservoir. Fluids are produced at the outer boundary. The injection pressure p_w and the production pressure p_e are kept constant by assumption. After stream tubes are generated, for each stream tube, the pressures on the inlet and outlet boundaries are constants (Figure 4.3). This solution is also applicable to the case when differential pressure between the injection well and production well is constant during the flow process. The pressure boundary conditions for the problem are:

$$p(0,t) = p_{in} = p_w, \tag{4.1}$$

$$p(L,t) = p_{out} = p_e, \tag{4.2}$$

where p_{in} and p_{out} are the inlet pressure and outlet pressure, respectively, and L is the length of the stream tube.

In accordance with classical Fractional Flow theory, we assume initial saturations for the reservoir and injected saturations are constant. The saturation boundary conditions are:

$$S_L = S(0,t) = 1 - S_{or}, t \geq 0, \tag{4.3}$$

$$S_R = S(x,0) = S_{wc}, x \in [0,L], \tag{4.4}$$

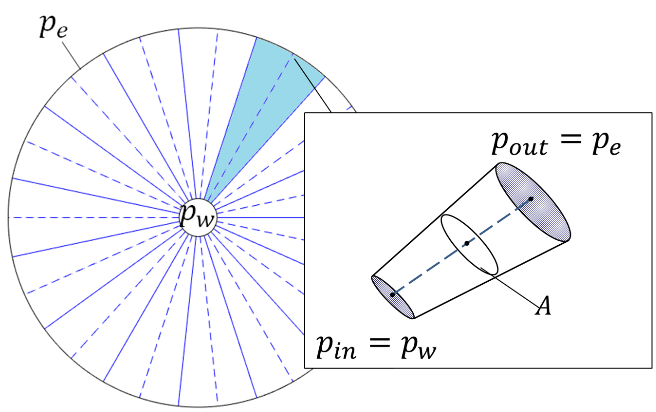


Figure 4.3: Near-Wellbore Stream Tube Sketch

where S_L and S_R are the water saturation at the inlet and outlet of the stream tube, respectively, and S_{or} is the residual saturation and S_{wc} is the connate water saturation.

A Riemann solution for this problem is described by a propagation of two waves. Specifically, the saturation jump is the leading shock front at saturation (S^*), see Figure 4.1 and **Appendix A**. The velocities participating in an overall global solution increase in the direction from the injection side to the production side. Equation 4.5 is used to calculate the fluid movement.

Before water breakthrough, the shock front flow rate q at a given time t is given by:

$$q(t) = \frac{\Delta p}{-\frac{V(x(S^*,t))}{f'(S^*)}\mathcal{J}(S^*) + \frac{1}{\lambda_R}\int_{x(S^*,t)}^{L}\frac{dx}{A(x)}}, \tag{4.5}$$

where

$$\mathcal{J}(S^*) = \int_{S^*}^{S_L} \frac{f''(S)dS}{A^2\left[V^{-1}[V((x(S^*,t))\frac{\phi f'(S)}{f'(S^*)}\right]\lambda(S)}. \tag{4.6}$$

Here, x represents arc length along the stream tube, Δp is the constant pressure difference between the inlet and the outlet, $x(S^*,t)$ is travel distance for front saturation S^* from the injection point at time t, $V(x)$ is the volume of the stream tube from injection to x, $A(x)$ is the cross section area for the stream tube at x, $f(S)$ is the fractional flow function, $f'(S)$, $f''(S)$ are the first and second derivative of water fractional flow with respect to water saturation S, respectively, λ is the total fluid mobility $\lambda = \lambda_o + \lambda_w$, and λ_R is the total mobility at the S_R, $\lambda_R = \lambda_{S_R}$.

The shock front flow rate is used to determine the travel distance from the classical Buckley and Leverett solution. Since S^* is known, for any $x^* \in [0, L]$, we can calculate $V(x(S^*, t))$. According to mass conservation, this volume during an infinitesimal time dt is the same as the injection volume during the same time:

$$dx(S^*, t) = \frac{q(t)f'(S^*)}{\phi A(x(S^*, t))} dt. \tag{4.7}$$

After breakthrough, the time for an arbitrary water saturation larger than shock front saturation $S \in [S^*, S_L]$ to reach the outlet $x = L$ is given by:

$$t_s = \frac{1}{2}[V^2(x(S, t^*)) - V^2(L)] \frac{\phi \mathcal{J}(S)}{\Delta p f'^2(S)} + t^*, \tag{4.8}$$

where

$$\mathcal{J}(S) = \int_S^{S_L} \frac{f''(s)ds}{A^2 \left[V^{-1}[V(x(S, t)) \frac{\phi f'(s)}{f'(S)}\right] \lambda(s)}, \tag{4.9}$$

where t^* is the breakthrough time of the front, ϕ is porosity, s is the saturation between S and S_L, $V(L)$ is the volume for the entire stream tube described before. The flow rate after breakthrough at time t_s can then be calculated by:

$$q(t_s) = \frac{[V^2(L) - V^2(x(S, t^*))]\phi}{2V^2(L)f'^2(S)(t_s - t^*)}. \tag{4.10}$$

Here, t^* is determined by integration of Equation 4.7 between $x = 0$ and $x = L$ using Equation 4.5 for $q(t)$.

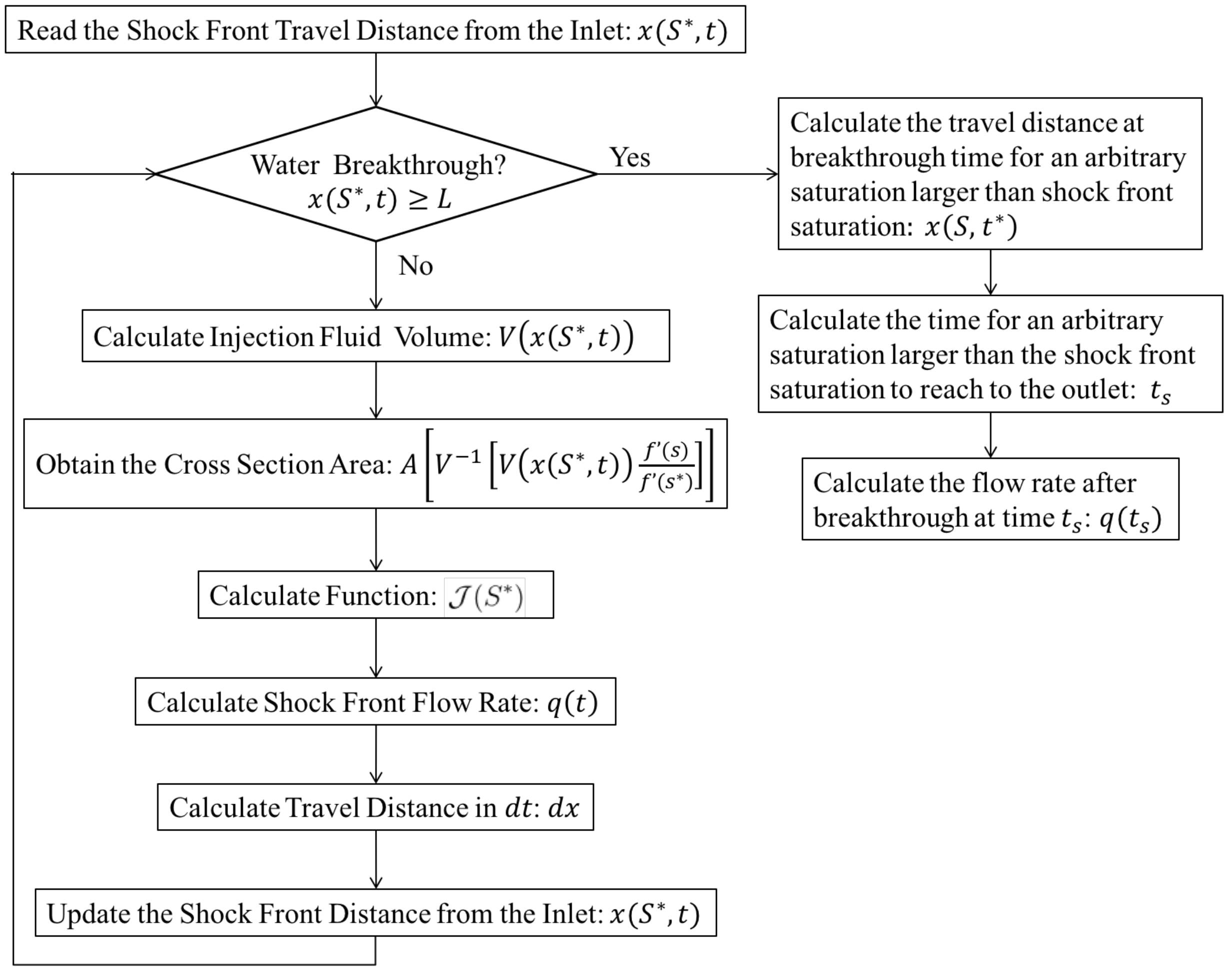


Figure 4.4: Flow Chart of Riemann Approach along Stream Tube

Figure 4.4 is the flow chart for simulating two-phase flow, using the semi-analytical Riemann approach along stream tubes.

1. Choose a stream tube.

2. Specify the initial shock front travel distance from the inlet, $x(S^*,t)$.

3. Determine if it is before breakthrough. If it is after breakthrough (the travel distance is larger than the stream tube length), label the stream tube as post

breakthrough and follow steps 9 through 11 below for post breakthrough calculation. Go back to 1 (next stream tube).

4. With the known travel distance $x(S^*, t)$ determine the volume of the stream tube $V(x(S^*, t))$ from the injection to x.

5. Calculate the value of $V(x(S^*, t))\frac{\phi f'(S)}{f'(S^*)}$, then use this value to calculate the value of $A^2\left[V^{-1}[V(x(S^*, t))\frac{\phi f'(S)}{f'(S^*)}\right]$. Then, obtain $\mathcal{J}(S^*)$ defined by Equation 4.6. The integration is obtained by the numerical method which is described in **Appendix C**.

6. Calculate flow rate $q(t)$ at time t by applying Equation 4.5.

7. Calculate the incremental travel distance dx in time interval dt by using Equation 4.7.

8. Update the shock front travel distance $x(S^*, t) = x(S^*, t) + dx$ and return to 3.

9. Calculate the travel distance $x(S, t^*)$ at breakthrough time t^* for an arbitrary saturation larger than shock front saturation $S \in [S^*, S_L]$ for any post breakthrough stream tube.

10. Calculate the time t_s for an arbitrary saturation larger than shock front saturation $S \in [S^*, S_L]$ to reach the outlet $x = L$.

11. Calculate the flow rate $q(t_s)$ for an arbitrary saturation larger than shock front saturation at time t_s.

Following this Riemann approach, the flow rate, and waterfront at different times can be determined.

4.1.2 Calculation of Stream tubes Areal Geometry

The streamlines are generated under the assumption of single-phase flow. In 2D, the space between two streamlines is a stream tube. The Riemann solution provides the analytical solution for a homogeneous stream tube. For a heterogeneous case, the length of each streamline and the stream tube boundaries are obtained numerically during the streamline tracing process. As shown in Figure 4.5, the stream tube boundaries are represented by the solid lines; streamlines in the middle of the stream tubes are represented by dotted lines. The cross section area is calculated as the equation in Figure 4.5. It is defined by the 4 points with the same radius.

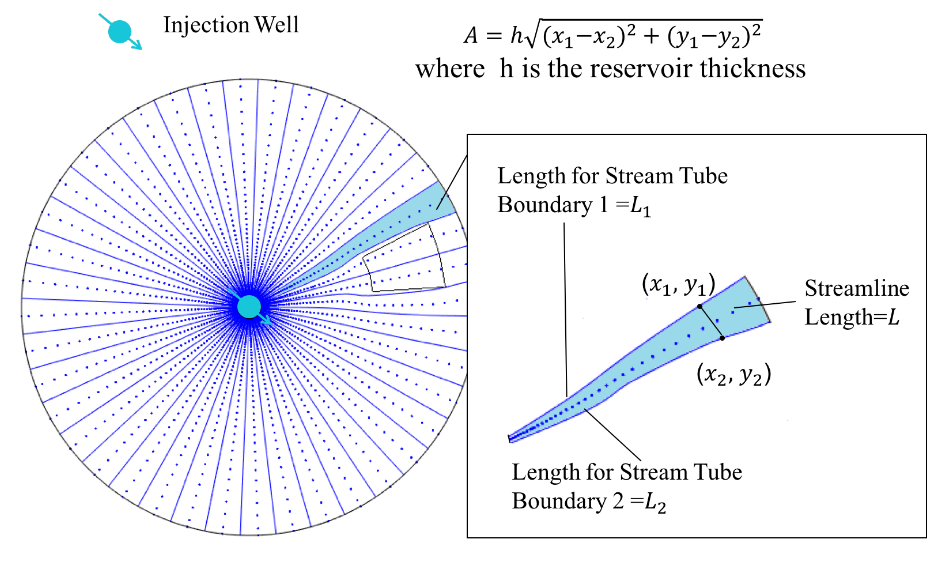


Figure 4.5: Stream Tube Area Calculation in 2D

For each stream tube, L_1 is the length of stream tube boundary 1, and L_2 for boundary 2. Both length can be calculated. The middle streamline length L is also known. The length of the center streamline is used to represent the length of the stream tube since the coordinates for this streamline may be needed for the heterogeneous stream tubes. As illustrated in Figure 4.5, stream tube boundaries and the center streamline are divided into the same number of segments. The stream tube area is a unique function of stream tube length since the stream tubes are fixed. Figure 4.6 is an example on the areal geometry versus stream tube length for different stream tubes. The permeability contrast between the block and the bulk in this particular example is 3/4. The stream tube length is the distance from the injector.

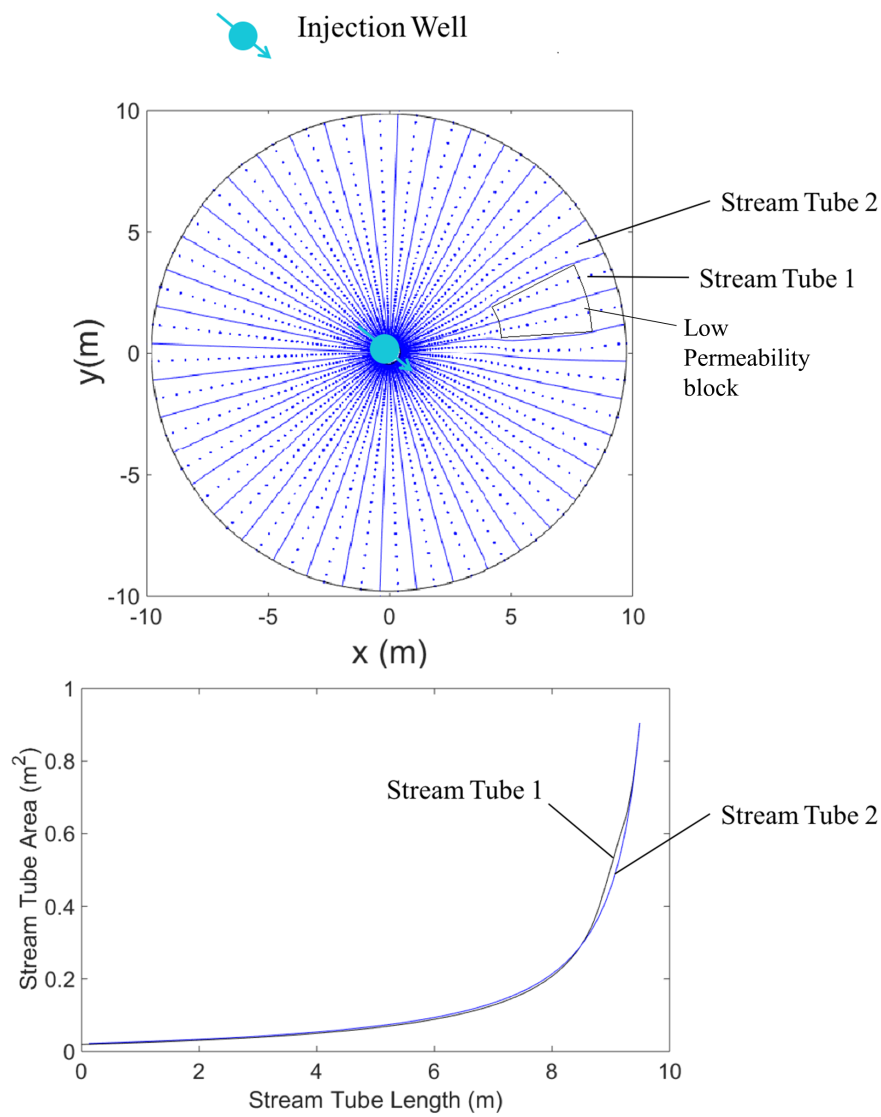


Figure 4.6: Area and Length Relationship for Two Selected Stream Tubes

4.1.3 Treatment of the Stream tubes with a Heterogeneity

The 3D Riemann solution described in **4.1.1** provides the analytical solution for a homogeneous stream tube. In the case of heterogeneous stream tube, it requires a special procedure. Figure 4.7 shows a stream tube with one heterogeneity. The permeability of the heterogeneity is denoted by K_H; permeability for the rest of the stream tube is K. The 3D Riemann solution can only be applied in stream tubes with unique permeability, any stream tube with heterogeneity inside is split into three homogeneous stream tubes. While generating the streamlines we can capture the intersection points between the center streamline and the heterogeneity. Once the coordinates of these points are known, the pressure for these two points p_1, p_2 can be calculated by using Equation 3.59. This stream tube is then split into three homogeneous stream tubes with known inlet and outlet pressures. The 3D Riemann approach can then be applied to each of them. In each stream tube, under the constant pressure boundaries, the flow rate varies with time, however, for a fixed time, it is constant as a function of stream tube arc length. The flow rate depends on the permeability, hence we first use pressure boundaries p_w, p_1 and the permeability K in Stream Tube 1 to calculate the water front movement and flow rate. Once the waterfront reaches the the intersection point 1, parameters (pressure boundaries p_1, p_2 and permeability K_H) in Stream tube 2 are used in the calculation to obtain the waterfront and flow rate. As soon as the front reaches the Intersection point 2, parameters in Stream tube 3 are used to calculate the water front movement and flow rate. The treatment of the stream tubes with heterogeneity requires only the use of the pre-breakthrough equations. We can alternatively use the upscaled permeability to do the movement calculation. Using the upscaled permeability only provides the same breakthrough time as the method we applied in this research thesis but it introduces errors for the frontal movement (smears out the water front along this stream tube).

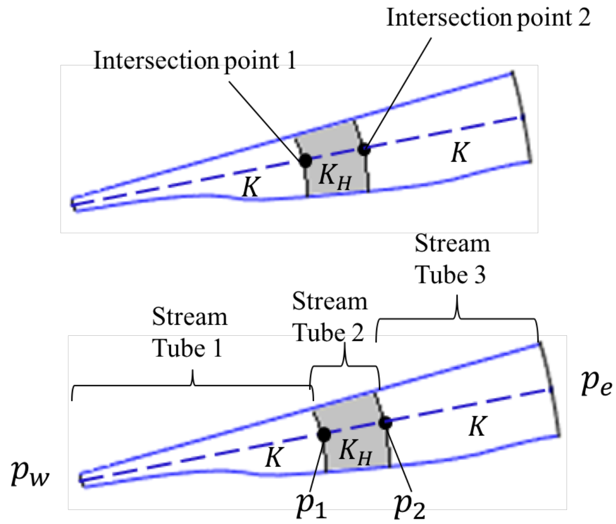


Figure 4.7: Stream Tube with a Heterogeneity

4.1.4 Riemann Solution for Homogeneous Radial Reservoirs

For a homogeneous reservoir, streamlines are straight radial lines from the injection well surface to the production ring. The $\theta-$coordinate along one of these streamlines is constant. The geometry for a stream tube is shown in Figure 4.8.

Then, the analytical solution for the flow rate is determined in Johansen and Liu (2016) as a special case of the general solution and is given by:

$$q(t) = \frac{2\Delta p\alpha h}{-[(x+r_w)^2 - r_w^2]\mathcal{J}(S^*) + \frac{1}{\lambda_R} ln\left(\frac{r_e}{x+r_w}\right)}, \tag{4.11}$$

where

$$\mathcal{J}(S^*) = \int_{S^*}^{S_L} \frac{f''(S)dS}{[r_w^2 f'(S^*) + [(x+r_w)^2 - r_w^2]f'(S)\phi]\lambda(S)}. \tag{4.12}$$

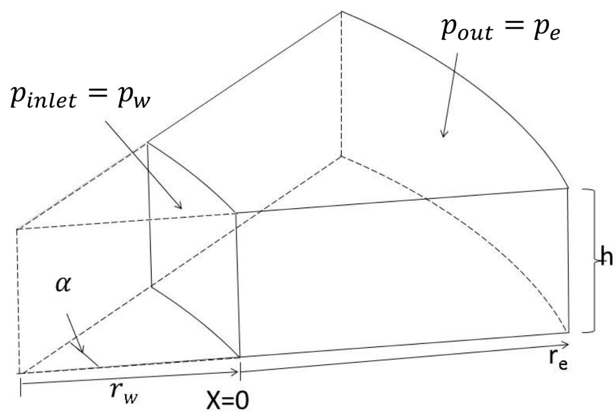


Figure 4.8: Stream Tube for Homogeneous Reservoirs (Johansen and Liu, 2016)

In **Chapter 5**, the 2D waterflooding process experiments were performed in glass-beads macro-models. The glass-beads macro-models properties are used to trace the streamlines. By applying the solution of 3D Riemann problem along each stream tube at constant pressure boundary conditions, the location of the water front at a specific time, the water breakthrough time, and the flow rates can be obtained. These simulated results are used to history match with the laboratory data. The application of the 3D Riemann approach along stream tubes in simulating macro-model waterfloodings is described in **Chapter 5**.

4.2 Streamline Modeling Case Studies in Open Hole Wells

This section describes the case studies for streamline simulation in single-phase flow with an open hole well completion *i.e.* where well completion details do not influence the stream line pattern. Two-phase flow case study results will be used to demonstrate the history matching ability for streamline simulation and will be described in **Chapter 5**, also for open hole wells. Table 4.1 summarizes cases discussed in this section.

Table 4.1: Summary of Case Studies-Open Hole Wells for Single-Phase Flow

Case	**Dimensions**	**Homogeneous/ Heterogeneous**	**Method Applied**
1	2D	Homogeneous	Fully Analytical Pollock's Method Semi-Analytical Method
2	2D	Heterogeneous- Low Permeability Sector (Two Subcases)	Pollock's Method Semi-Analytical Method
3	2D	Heterogeneous- High Permeability Sector (Two Subcases)	Pollock's Method Semi-Analytical Method
4	3D	Homogeneous anisotropic	Pollock's Method Semi-Analytical Method
5	3D	Heterogeneous	Semi-Analytical Method

4.2.1 Case 1: 2D Homogeneous Case

In a homogeneous reservoir, the permeability throughout the reservoir is constant. The domain simulated is a cylindrical ring with an inner radius of 0.05 m (wellbore) and an outer radius of 50 m. The permeability is isotropic and homogeneous which equals to 1.0 *Darcy*. The inner and outer boundaries have constant pressures of 280 *bar* and 300 *bar*, respectively. The details of the parameters used in this case are shown in Table 4.2. Figure 4.9 shows the pressure profile for a homogeneous near-wellbore reservoir. The r-axis represents the radius from the wellbore center and the p-axis represents the corresponding pressure. The colour represents the pressure in accordance with the colour bar. The pressure profile is identical for all angles since the formation is homogeneous. The symmetric pressure distribution has a funnel shape in the near-wellbore region. Pressure decreases in a logarithmic fashion in the radial direction towards the wellbore but is constant as a function of angle.

Table 4.2: Parameters used for Open Hole Case 1

Parameters	**Units**	**Values**
Wellbore Radius	m	0.05
External Radius	m	50
Radial Blocks		50
Tangential Blocks		50
Wellbore Pressure	Pa	280×10^5
External Pressure	Pa	300×10^5
Bulk Permeability	m^2	1×10^{-12}
Oil Viscosity	$Pa \cdot s$	10^{-3}

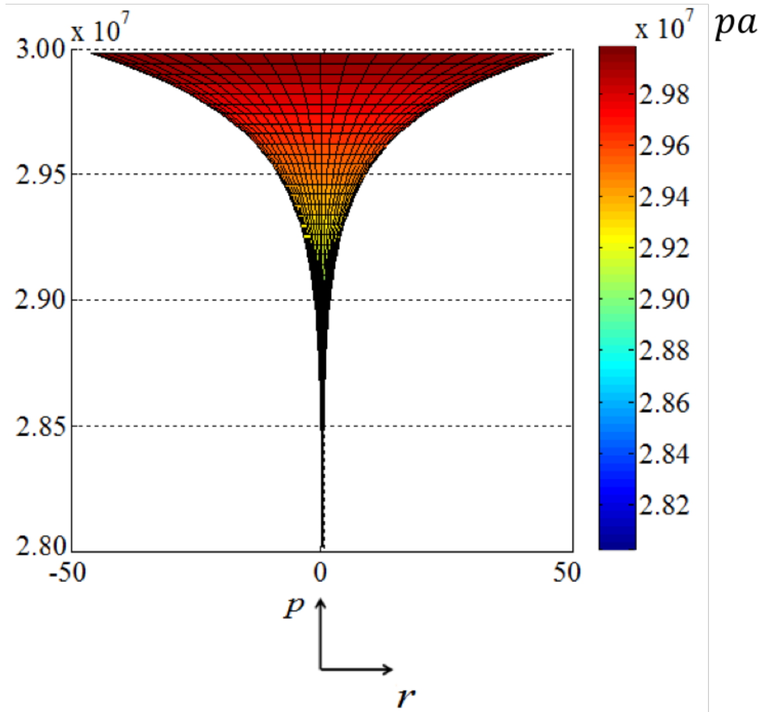


Figure 4.9: Pressure Distribution for Open Hole Case 1

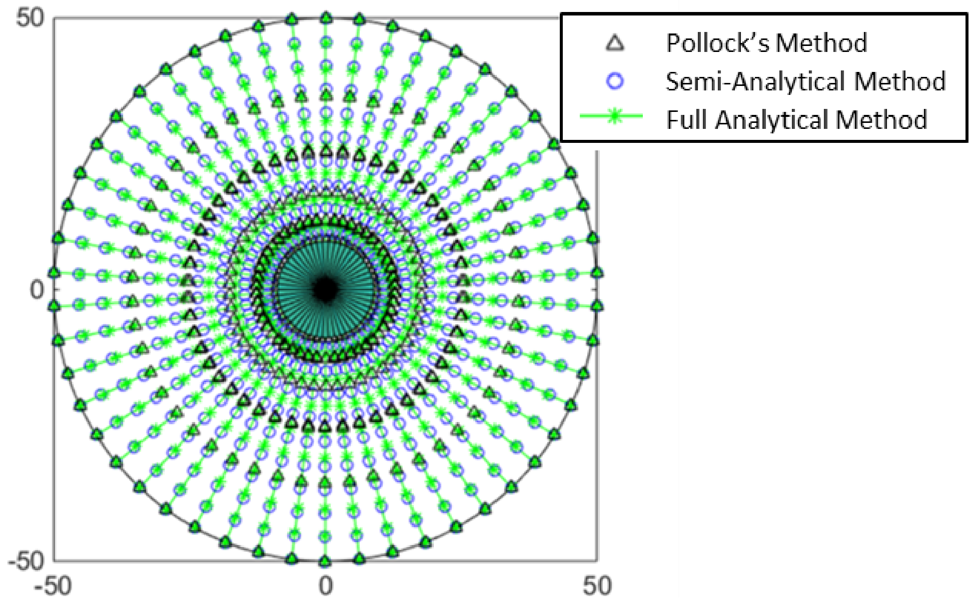


Figure 4.10: Streamline Traced by Different Methods for Open Hole Case 1

For Case 1, a fully analytical solution exists as will be described below. The streamlines are traced by three different methods: the semi-analytical method, Pollock's method, and the fully analytical method. As shown in Figure 4.10, entry angle and exit angle for each streamline are the same for all the methods. Hence streamline trajectories for all methods are identical. This is because the pressure decreases in a logarithmic fashion in the radial direction towards the wellbore and there is no pressure gradient in the θ-direction at the same radius. However, the time-of-flight values differ from the fully analytical solution for the two approximate methods. To quantify the variations, the average relative error for the time-of-flight is calculated by Equation 4.13 below. The relative error in TOF for an approximate method is defined as:

$$e_{TOF} = \sum \frac{|TOF_i - tof_i|}{TOF_i}, \tag{4.13}$$

where TOF_i and tof_i are the fully analytical and the approximately calculated incremental time-of-flight for radial interval i, respectively.

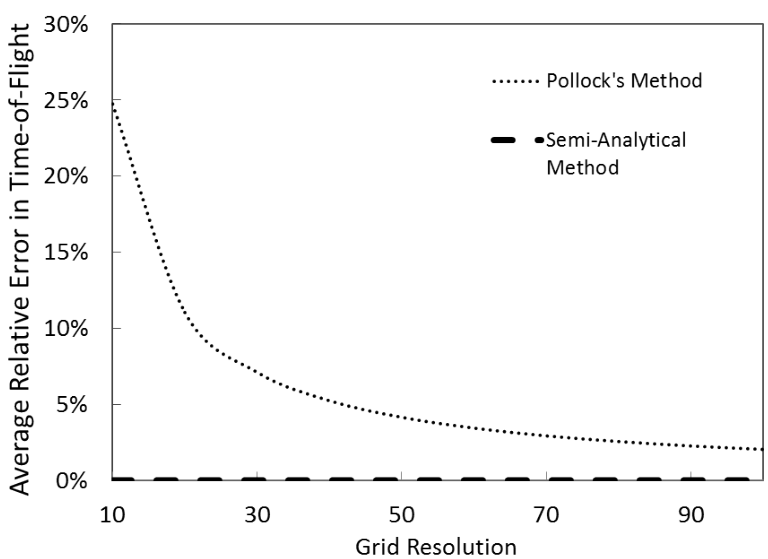


Figure 4.11: Relative Errors in Time-of-Flight for Open Hole Case 1

Table 4.3: Relative Errors in Time-of-Flight for Open Hole Case 1

Grid Block Number in the Radial Direction	Error for Pollock's Method	Error for Semi-Analytical Method
10	2.47×10^{-1}	4.02×10^{-13}
20	1.10×10^{-1}	1.21×10^{-12}
30	7.10×10^{-2}	4.00×10^{-12}
40	5.23×10^{-2}	1.48×10^{-11}
50	4.14×10^{-2}	7.51×10^{-12}
60	3.43×10^{-2}	1.40×10^{-11}
70	2.92×10^{-2}	2.92×10^{-11}
80	2.55×10^{-2}	1.04×10^{-10}
90	2.26×10^{-2}	3.72×10^{-11}
100	2.03×10^{-2}	5.08×10^{-11}

Figure 4.11 and Table 4.3 show the error for the present semi-analytical method and Pollock's method relative to the fully analytical solution. The result shows that the present semi-analytical method is in agreement with the analytical solution. When the grid resolution is low, Pollock's method to determine the time-of-flight exhibits unacceptable errors.

Next, we will prove that for the homogeneous reservoir the present semi-analytical method is mathematically identical to the fully analytical solution. Therefore, the errors for the semi-analytical method listed in Table 4.3 are caused by digital truncation only.

At steady state the pressure equation in cylindrical coordinates is:

$$\frac{d}{dr}\left(r\frac{dp}{dr}\right) = 0. \tag{4.14}$$

where r is the radius and p is the pressure. The general solution for Equation 4.16 is:

$$p(r) = Alnr + B. \tag{4.15}$$

where the constants A and B is determined from the boundary conditions. The pressure form is the same as in the present semi-analytical method.

Consider a homogeneous reservoir of inner radius r_w and outer radius r_e. The corresponding pressures are p_w and p_e. The analytical pressure for any radius is:

$$p(r) = \left(\frac{ln(r/r_w)}{ln(r_e/r_w)}\right)(p_e - p_w) + p_w. \tag{4.16}$$

The fully analytical solution for the time-of-flight is:

$$TOF = \phi\frac{(r_e^2 - r_w^2)}{2Krln(r_e/r_w)}. \tag{4.17}$$

This TOF expression is the same expression as in the present semi-analytical method as shown in Equation 3.68. Hence, for the homogeneous reservoir, the semi-analytical solution provides the same pressure and time-of-flight results as the analytical method.

As described in **3.4.4**, in the homogeneous case, the velocity assumption used in Pollock's method is equivalent to assuming a pressure function as:

$$p(r,\theta) = A(lnr) + Cr + E. \tag{4.18}$$

It is different from the fully analytical method, which causes the TOF error in Pollock's method. This points to the fact that Pollock's method in radial geometries in general hampered by systematic error. It also explains its performance in 2D heterogeneous reservoirs. In radial geometries, these errors in Pollock's method are severe because of the logarithmic (non-linear) pressure distribution. This is in contrast to Cartesian geometries, where such behaviour is not observed because the pressure is linear.

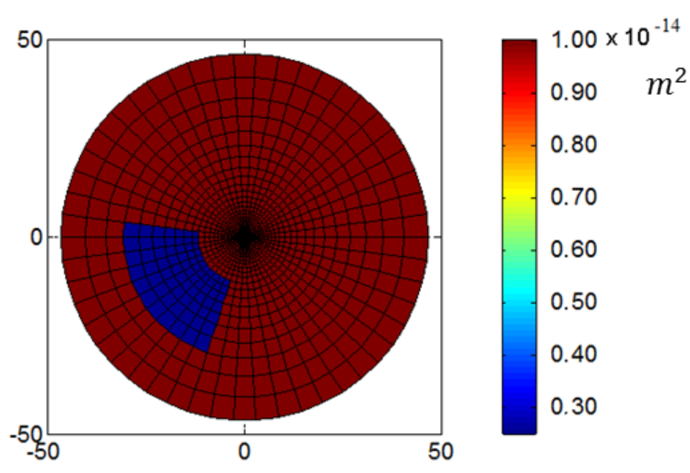
4.2.2 Case 2: 2D Heterogeneity with a Low Permeability Sector

In the near-well region, heterogeneities always exist. No analytical solution can be found for heterogeneous reservoirs except in idealized situations. The streamlines are traced by the two methods (present semi-analytical method and Pollock's method). In Case 2, a large area of low permeability sector is placed in the third quadrant of the reservoir. We show two subcases here: In Subcase 2.1 the ratio of permeabilities between the heterogeneity and the bulk reservoir is 1/4 and in Subcase 2.2 the ratio is 1/2. The detailed parameters used are shown in Table 4.4.

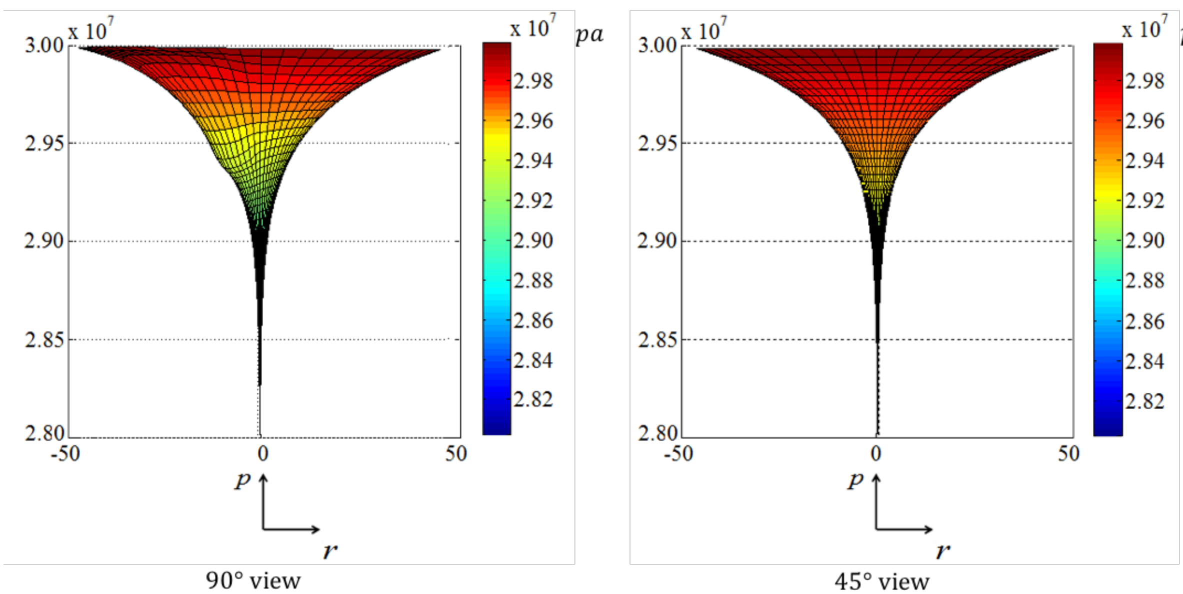
Table 4.4: Parameters used for Open Hole Case 2

Parameters	**Units**	**Values**	
		Subcase 2.1	Subcase 2.2
Wellbore Radius	m	0.05	0.05
External Radius	m	50	50
Radial Blocks		50	50
Tangential Blocks		50	50
Wellbore Pressure	Pa	280×10^5	280×10^5
External Pressure	Pa	300×10^5	300×10^5
Bulk Permeability	m^2	1×10^{-12}	1×10^{-12}
Oil Viscosity	$Pa \cdot s$	10^{-3}	10^{-3}
Radial Blocks with Low Permeability		40-47	40-47
Tangential Blocks with Low Permeability		25-36	25-36
Block Permeability	m^2	0.25×10^{-12}	0.5×10^{-12}

The permeability field is shown in Figure 4.12 (a) and the pressure distribution in Figure 4.12 (b) for Subcase 2.1. The red area represents the block with bulk permeability; the blue area represents the low permeability area. The pressure roughly decreases in a logarithmic fashion as it approaches the wellbore, as in the homogeneous case. However, the heterogeneity has a significant influence on the pressure distribution locally. The lower permeability value results in a larger radial pressure gradient within the heterogeneous sector. As can be seen from the 90° side view, the pressure for the grid blocks surrounding the heterogeneity also changes to accommodate the pressure change in the heterogeneous area (Skinner, 2011). As can be seen from the 45° side view, the pressure decreases smoothly in a logarithmic fashion towards the wellbore. The pressure distribution for Subcase 2.2 is similar to the pressure distribution for Subcase 2.1, just with a larger pressure change.



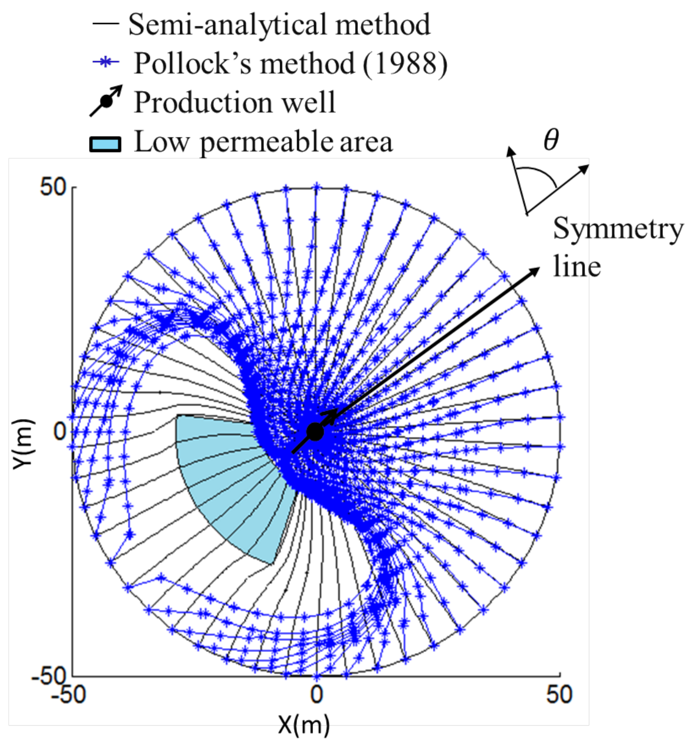
(a) Permeability Profile



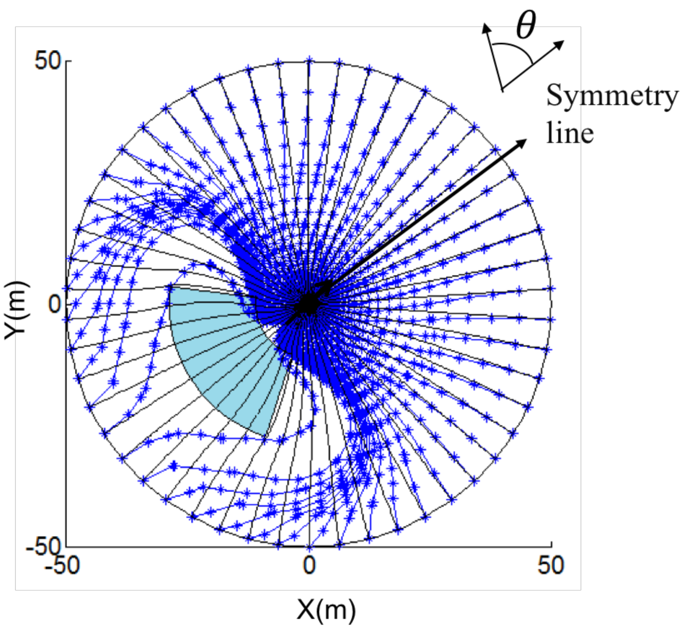
(b) Pressure Profile

Figure 4.12: Permeability and Pressure Profile for Open Hole Subcase 2.1

As can be observed from the Figure 4.13, the results from the present method and Pollock's method do not coincide. For Subcase 2.1, ($K_{block}/K_{bulk} = 1/4$) streamlines generated by Pollock's method do not flow across the low permeability area and the nearby grid blocks. This appears unphysical, as the grid blocks with the higher permeability should allow fluid flow. It is therefore concluded that Pollock's method produces a systematic error in radial cases which is also apparent in the TOF calculations. On the contrary, only some of the semi-analytical streamlines, very close to the low permeable boundaries, avoid flowing across the low permeable region which is physically far more reasonable. For Subcase 2.2 ($K_{block}/K_{bulk} = 1/2$), it is observed that, unlike the previous case, some streamlines generated by Pollock's method flow through the low permeability area. However, streamlines generated by the present semi-analytical method maintain the same trend as in the previous case, *i.e.* only some of the streamlines, very close to the low permeable boundaries, avoid flowing across the low permeable region. Pollock' s method gives unrealistic results since the streamlines avoid the low permeability area in both cases excessively. This is in contrast to the physical fact that some of the fluid will flow through these regions, and this is captured by the present semi-analytical method.



(a) Subcases 2.1: $K_{block}/K_{bulk} = 1/4$



(b) Subcases 2.2: $K_{block}/K_{bulk} = 1/2$

Figure 4.13: Streamlines for Open Hole Case 2

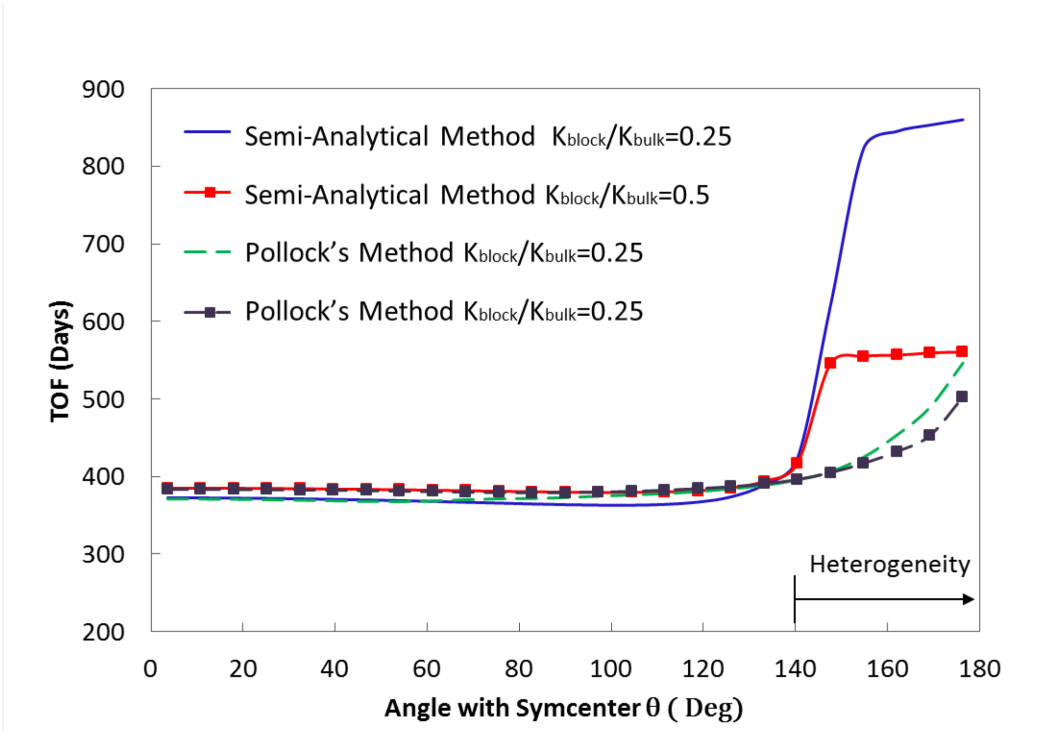


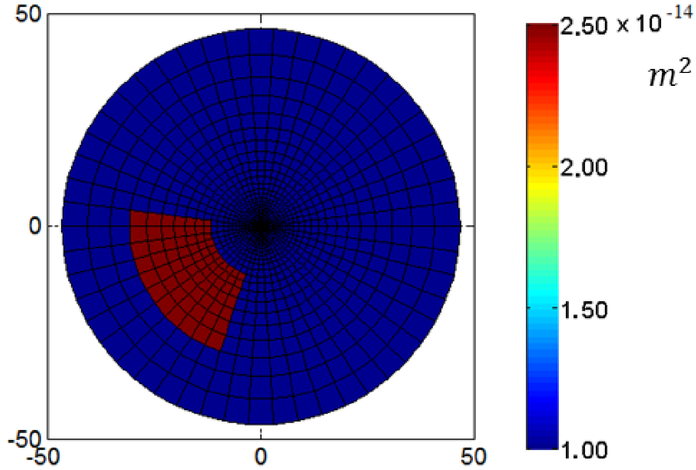
Figure 4.14: TOF for Different Methods for Open Hole Case 2

Since the reservoir is symmetric, half of the $TOFs$ of the streamlines are shown in Figure 4.14. According to this figure, $TOFs$ between Pollock's method and the present semi-analytical method have the same tendency: $TOFs$ increase as streamlines get closer to the heterogeneity. However, $TOFs$ obtained from the present semi-analytical method increase more than that from Pollock's method. $TOFs$ from the present semi-analytical method are separated into two sections: in the heterogeneity and in the bulk. They are relatively stable in their respective section. This means that almost independently of where the streamline is located, almost the same travel time is required as long as the streamlines are in the same section.

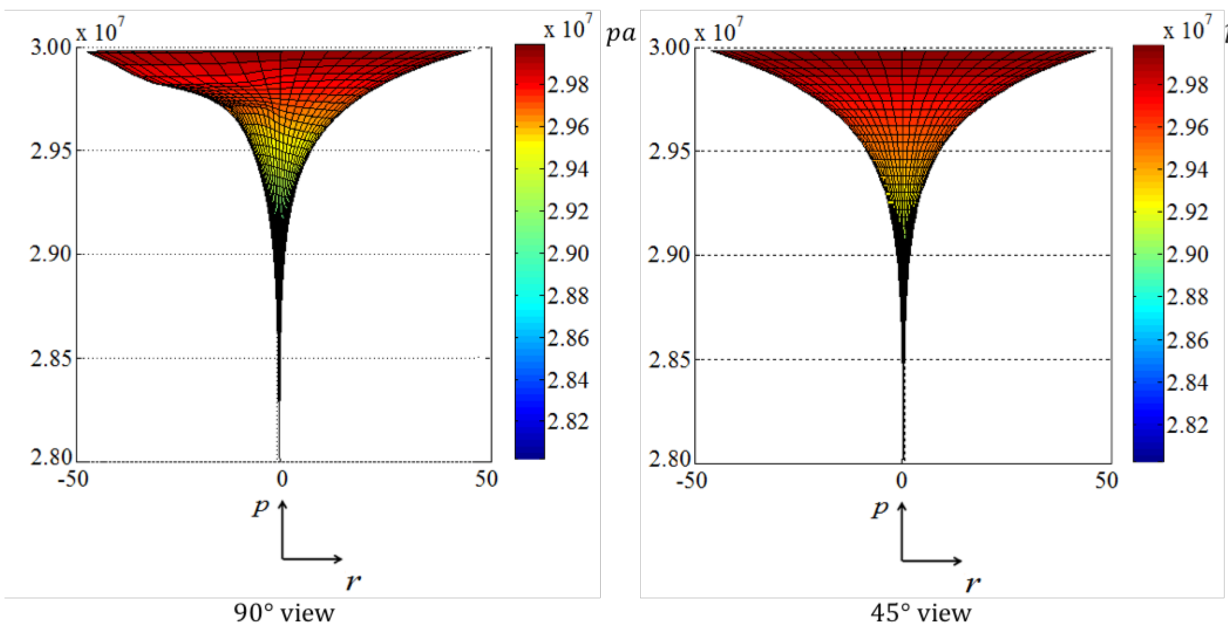
4.2.3 Case 3: 2D Heterogeneity with a High Permeability Sector

In Case 3, as opposite to the pervious case, a large area high permeability was placed in the third quadrant of the reservoir. We also show two subcases here: In Subcase 3.1 the ratio of permeabilities between the heterogeneity and the bulk reservoir is 4/1 and in Subcase 3.2 the ratio is 2/1. The remaining parameters used in this case are the same as for the two-dimensional heterogeneity case with low permeability. Figure 4.15 shows the permeability profile and pressure profile for Subcase 3.1. The red area represents a high permeability sector in the reservoir. It is noticed that Subcase 3.1 has an opposite change of pressure distribution within the heterogeneity compared to two-dimensional heterogeneity with low permeability case in **4.2.2**, showing a smaller pressure drop within the heterogeneous sector. The pressure distribution for Subcase 3.2 is similar with Subcase 3.1 with less pressure change at the heterogeneity.

As illustrated in Figure 4.16, streamlines from Pollock's method avoid the area on the sides of the heterogeneity which is unphysical. It is attributed to the same systematic errors in Pollock's method as described in **4.2.2**. Similar to the previous case, the present semi-analytical method provides a more reasonable result.

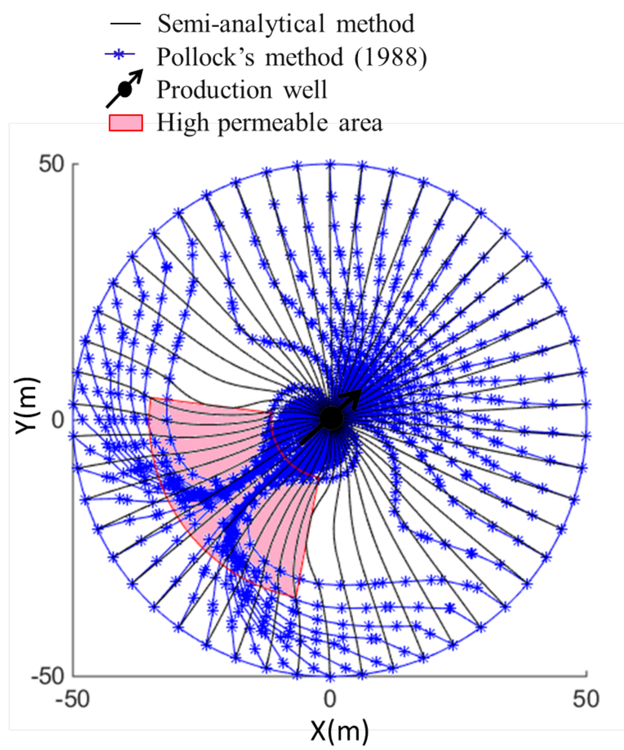


(a) $PermeabilityProfile$

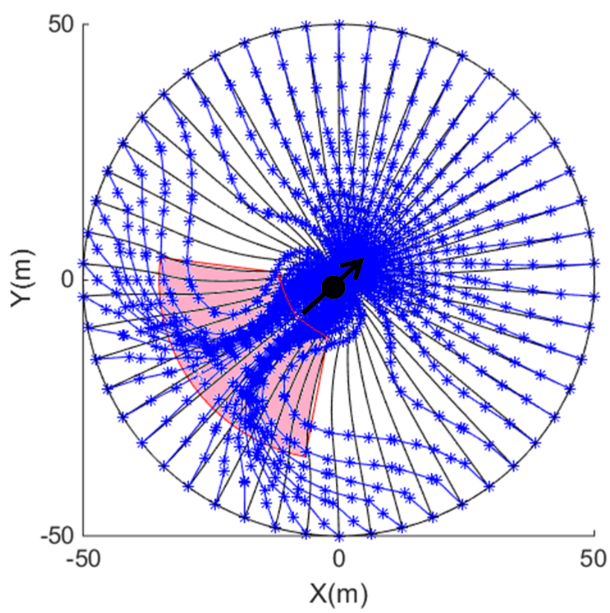


(b) $PressureProfile$

Figure 4.15: Permeability and Pressure Profile for Open Hole Subcase 3.1



(a) Subcase 3.1: $K_{block}/K_{bulk} = 4/1$



(b) Subcase 3.2: $K_{block}/K_{bulk} = 2/1$

Figure 4.16: Streamlines for Open Hole Case 3

4.2.4 Case 4: 3D Homogeneous and Anisotropic Case

In this section, we demonstrate the performance of the present method in three dimensions through a homogeneous anisotropic reservoir case. The basic 3D case is shown in Figure 4.17 and considers a homogeneous cylinder reservoir with a radius of 50 m and height of 50 m. A production well with a diameter of 0.3 m is placed in the center of the bottom layer of this reservoir, which has a fixed pressure of 150 bar. An injection ring (pressure support) with a radius of 50 m is opened on the top layer with a constant injection pressure of 200 bar. The permeability in the $z-$direction is 10 times smaller than the permeability in the radial and angular direction. The rest of the outer boundaries are no flow boundaries. The details for this case are shown in Table 4.5.

Table 4.5: Parameters used for Open Hole Case 4

Parameters	**Units**	**Values**
Wellbore Radius	m	0.3
External Radius	m	50
Radial Blocks		10
Tangential Blocks		20
Z-direction Layers		6
Wellbore Pressure	Pa	150×10^5
External Pressure	Pa	200×10^5
Bulk Permeability in x- and y-directions	m^2	1×10^{-13}
Bulk Permeability in z-directions	m^2	1×10^{-14}
Oil Viscosity	$Pa \cdot s$	10^{-3}

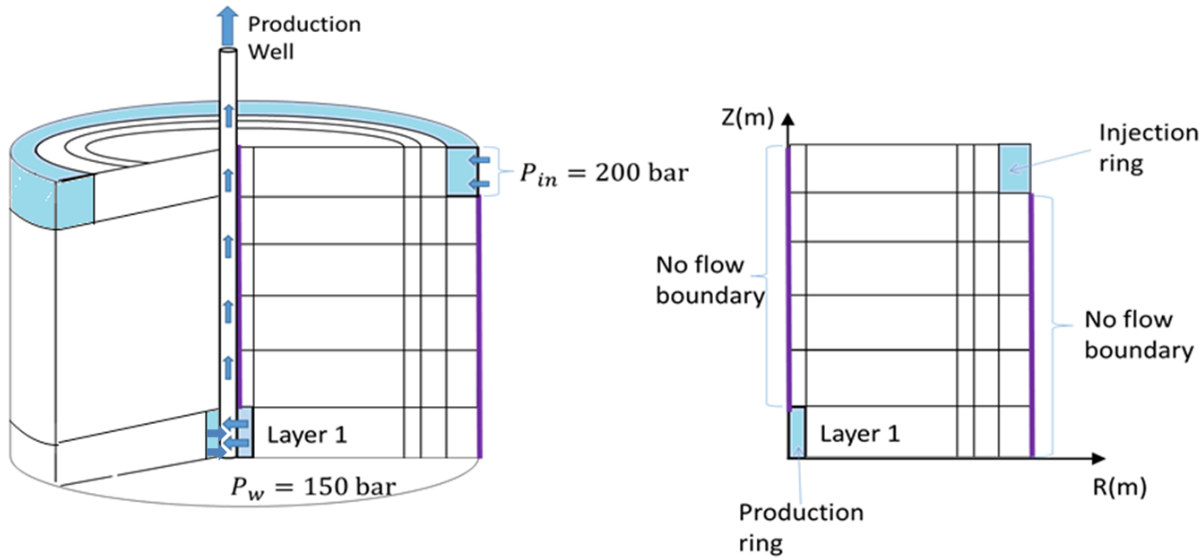
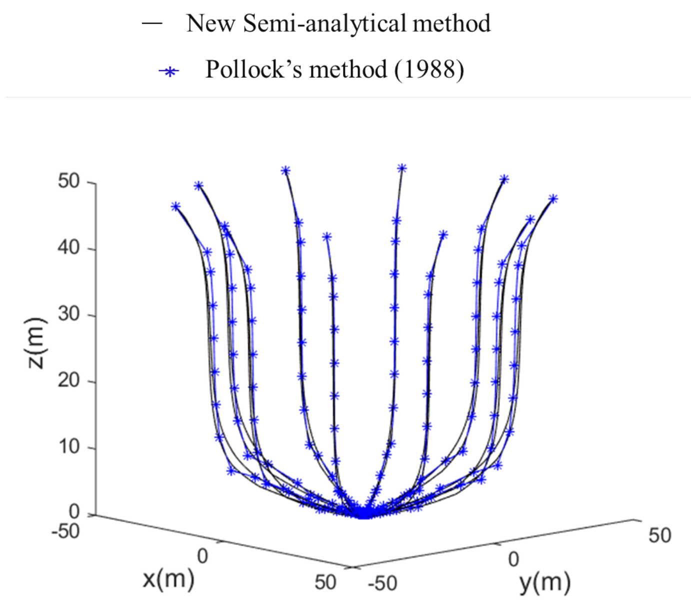


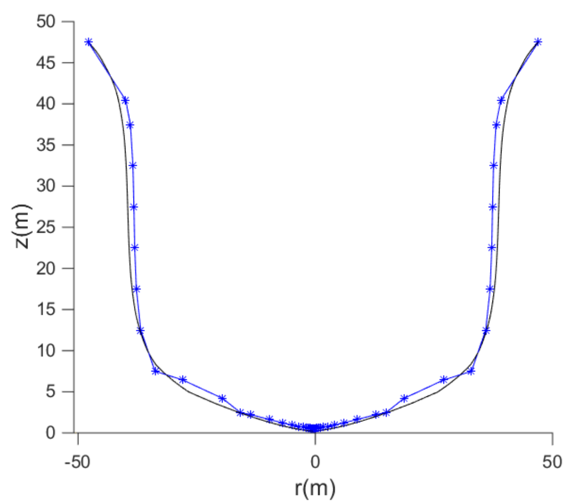
Figure 4.17: Open Hole Case 4 Structure

Both Pollock's method and the present semi-analytical method are applied to simulate this case. The results in Figure 4.18 show that the present method gives a more physically reasonable result with smoother streamline trajectories.

The TOF for a streamline from Pollock's method and the present semi-analytical method are 6.68×10^9 s and 7.42×10^9 s, respectively, a discrepancy of approximately 10%, which is significant since it is reflective of the breakthrough time of a displacement front.



(a) 3D View

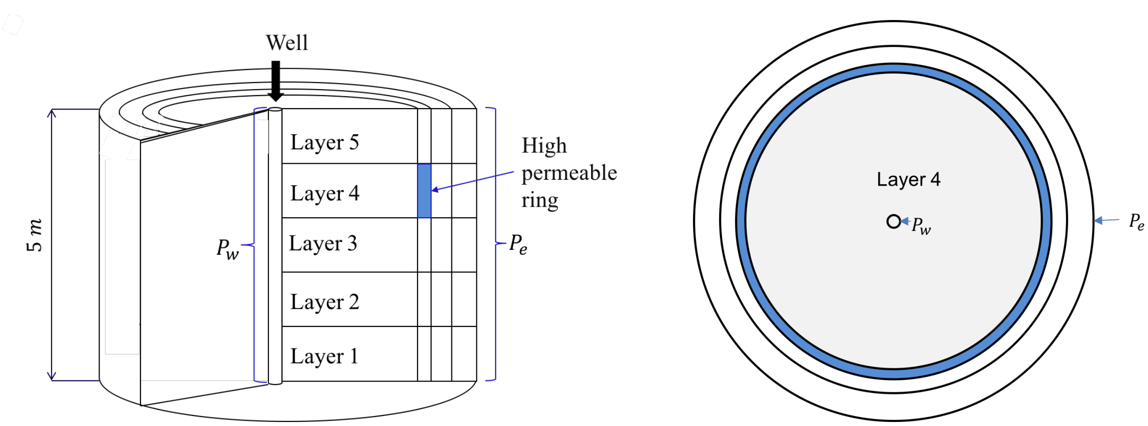


(b) Side View

Figure 4.18: Streamlines for Open Hole Case 4

4.2.5 Case 5: 3D Heterogeneous Case

In the near-wellbore region, the reservoir radius is $r_e = 12\ m$ with a well in the center $r_w = 0.05\ m$ and the reservoir thickness is $z = 5\ m$. The pressures along the wellbore and the reservoir boundary in the $z-$direction are assumed constant. A high permeability ring is placed from 3 m to 4 m in the $z-$direction, as illustrates in Figure 4.19. In the near-wellbore region, the high permeability areas usually start from the the inner boundary because of perforation; however, in order to show curvature of streamlines easily, the high permeability zone is arranged to the area very close to the outer boundary. The detailed parameters used are shown in Table 4.6. Figure 4.20 shows streamlines generated by the present semi-analytical method. Pollock's method can not provide streamlines for this case.

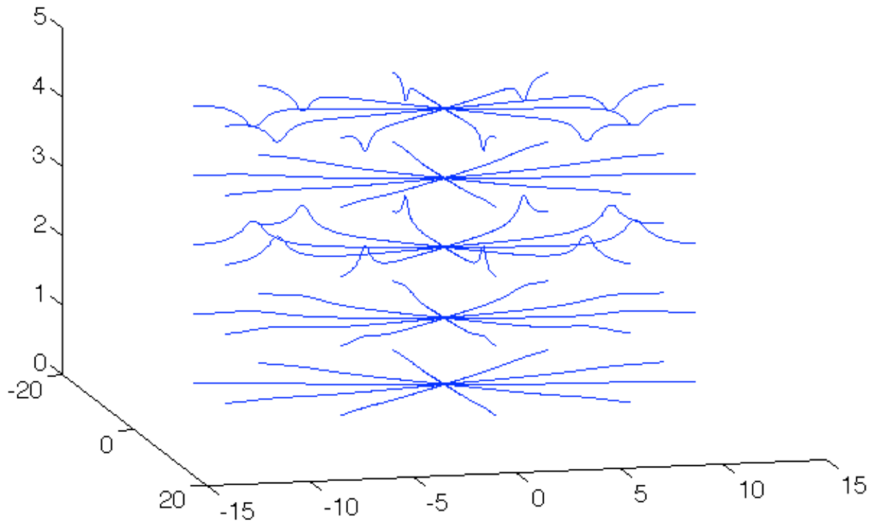


(a) Overall View (b) Top View

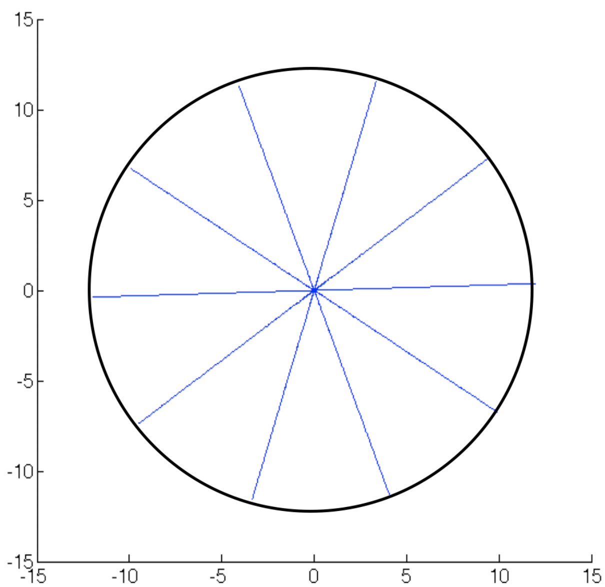
Figure 4.19: Open Hole Case 5 Structure

Table 4.6: Parameters used for Open Hole Case 5

Parameter	**Units**	**Value**
Wellbore Radius	m	0.05
External Radius	m	12
Reservoir Thickness	m	5
Radial Blocks		50
Tangential Blocks		20
Z-direction Layers		5
Wellbore Pressure	Pa	280×10^5
External Pressure	Pa	300×10^5
Bulk Permeability	m^2	1×10^{-13}
Oil Viscosity	$Pa \cdot s$	1×10^{-3}
Radial Blocks with High Permeability		48
Tangential Blocks with High Permeability		1-20
Layers with High Permeability		4
Block Permeability	m^2	1^{-12}



(a) Overall View

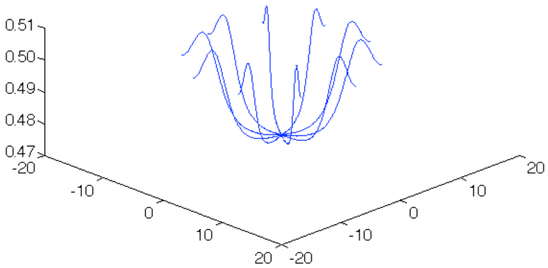


(b) Top View

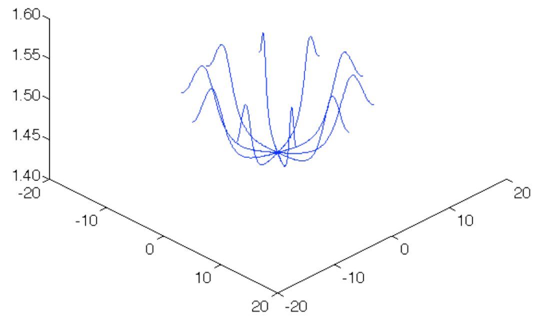
Figure 4.20: Streamlines for Open Hole Case 5

Streamlines are fluctuating in each layer however within a small range along the $z-$axis. Figure 4.21 shows the streamlines launching form the outer boundary for each layer seperately. In Figure 4.21, the $x-$axis and $y-$axis represent the spatial location through the reservoir, while the $z-$axis represents the vertical distance. Streamlines tend to approach upwards in the middle for the first 4 layers. Streamlines tend to approach downwards in the middle of Layer 5.

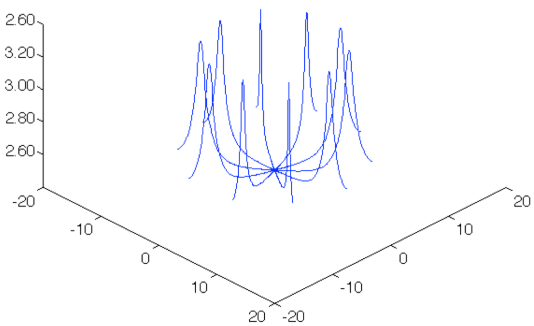
Figure 4.22 is the side sketch for Case 5. The main purpose for this figure is to show the tendency in the $z-$direction for all layers. The high permeability ring is at layer 4, as can be seen from Figure 4.22. All streamlines tend to approach to the high permeable ring and the level of tendency is depending on the distance from the high permeable ring. In other words, the closer to the high permeable ring, the more obvious the approach tendency is. Layer 4 is different with this tendency. It is because the upscaled permeability between layer 4 and layer 3 and the upscaled permeability between layer 4 and layer 5 used in the pressure calculation overcome the heterogeneity affect in this layer.



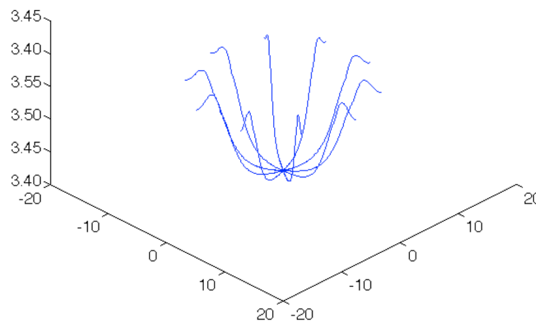
(a) Layer 1



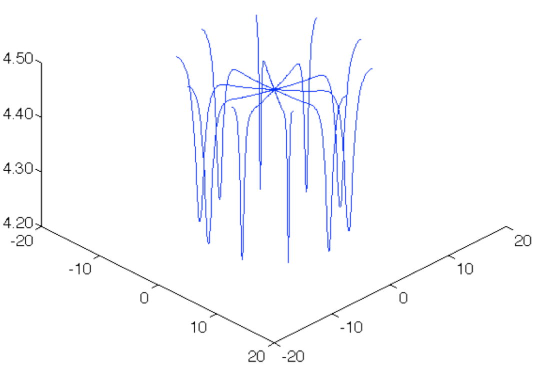
(b) Layer 2



(c) Layer 3



(d) Layer 4



(e) Layer 5

Figure 4.21: Streamlines for Open Hole Case 5 in Different Layers

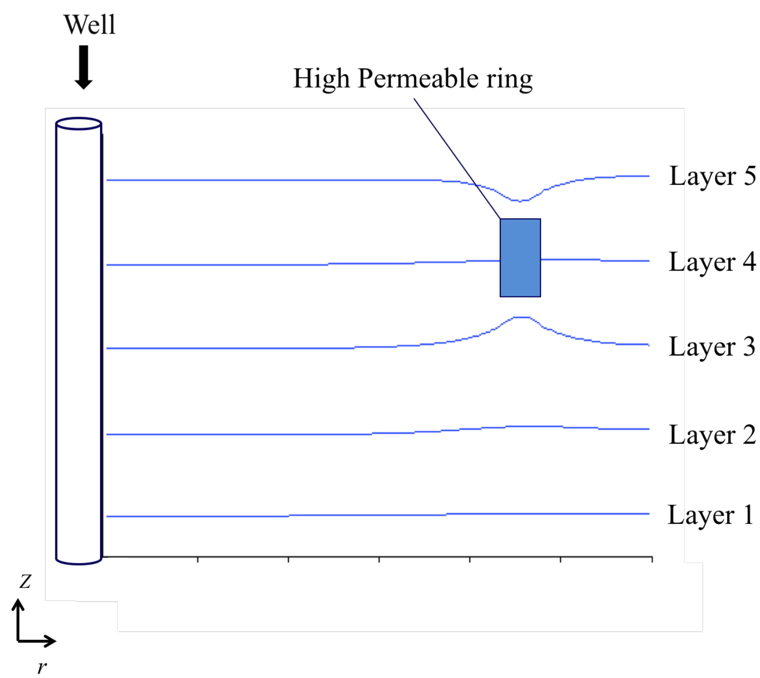


Figure 4.22: Side Sketch for the 3D Case

4.3 Skin Calculation by Using Streamline Simulation Method

In the near-wellbore region, rock properties may vary due to different factors such as drilling damage, perforation damage, crushed zone damage and other effects. This causes an additional pressure drop in the near-wellbore region, which can be expressed by a mechanical skin factor. Assuming one-phase steady-state incompressible flow in an undamaged near-well region, the flow rate in a homogeneous and isotropic formation of thickness h can be calculated by:

$$Q = \frac{2\pi K h(p_e - p_w)}{\mu ln(r_e/r_w)}, \tag{4.19}$$

where K is the permeability, r_e and r_w are the radius for an external boundary and the wellbore, respectively. The corresponding pressure at the external boundary is denoted as p_e and the pressure in the wellbore as p_w.

In a perforated well, additional pressure drop is caused by flow convergence into perforations, crushed formation in the vicinity of the perforations, and damaged zone from mud invasion (Figure 4.23). The permeability for the damaged zone is smaller than the reservoir permeability.

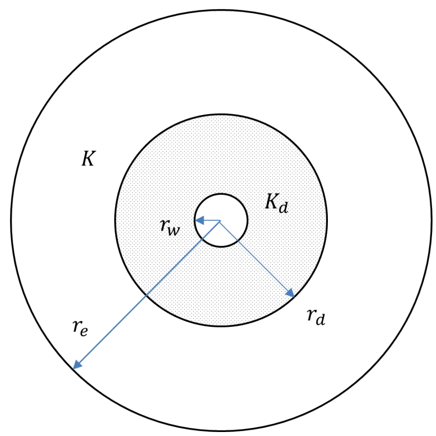


Figure 4.23: Damaged Zone in the Near-Wellbore Region

For steady-state incompressible flow, Equation 4.19 can be written for each region in Figure 4.23 separately:

$$Q = \frac{2\pi K_d h(p_d - p_w)}{\mu ln(r_d/r_w)}, \tag{4.20}$$

$$Q = \frac{2\pi K h(p_e - p_d)}{\mu ln(r_e/r_d)}. \tag{4.21}$$

Since the fluid is assumed incompressible, the flow rate Q is the same in both regions. Usually K_d and r_d are unknown. Combining Equation 4.20 and 4.21, the flow equation including skin can be written as:

$$Q = \frac{2\pi K h(p_e - p_w)}{\mu \left[ln(r_e/r_w) + S\right]}, \tag{4.22}$$

where S is the mechanical skin factor.

Each stream tube has a unique flux. The flow rate for any stream tube can be calculated by:

$$q_i = \frac{K_{ri}}{\mu} \int_{\theta_l}^{\theta_u} \frac{\partial p_i}{\partial ln r_D} \partial\theta, \tag{4.23}$$

where K_{ri} is the permeability in the radial direction for grid block i in the outer ring. Here, θ_u and θ_l are the upper and lower limit in the angular direction, respectively.

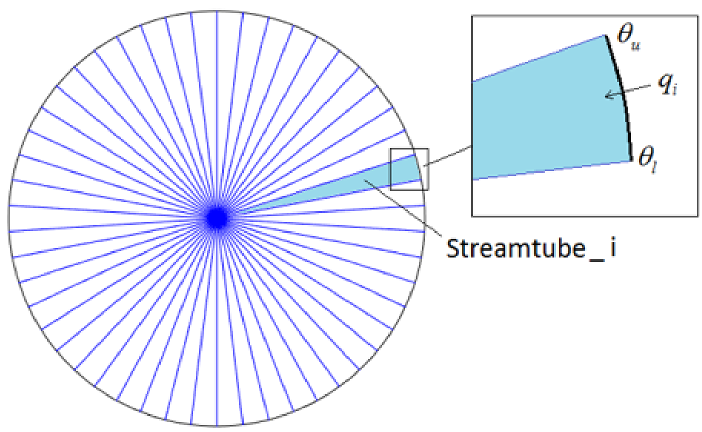


Figure 4.24: Stream Tubes Representation for Near-Wellbore Region in 2D

In this near-wellbore streamline simulation, entry points for a number of streamlines are defined on the external boundary. After calculating the streamlines using the present semi-analytical method, they are bundled into stream-tubes. Since the launching points of streamlines are known, the lower and upper limits in the-θ direction are also known.

The total flow rate for the entire reservoir can be obtained by summing the contribution for individual stream tubes, $Q = \sum q_i$. The total mechanical skin can then be calculated from Equation 4.22. The skin calculations using the streamline simulation method can be used to calculate the overall productivity of the well. Streamlines do not have a flow rate. Therefore, we use the stream tubes here.

4.3.1 The Skin Components in Perforated Wells

During the drilling process, the formation is damaged by drilling mud invasion unless under-balanced drilling is used. However, the vast majority of wells are drilled over-balanced for regulatory reasons. Such damage will result in a reduced permeability in a near well region (Figure 4.23). It is known as the damaged zone permeability K_d, which is smaller than the formation permeability K. Klotz et al. (1974) and Hong (1975) concluded that the contribution of the damaged zone to the total skin in a perforated well heavily depends on the perforation length and the damaged zone radius. Hawkins' formula (Equation 4.24) shows the relative effects of permeability impairment and the penetration of damage:

$$S = \left(\frac{K}{K_d} - 1\right) ln \left(\frac{r_d}{r_w}\right) \tag{4.24}$$

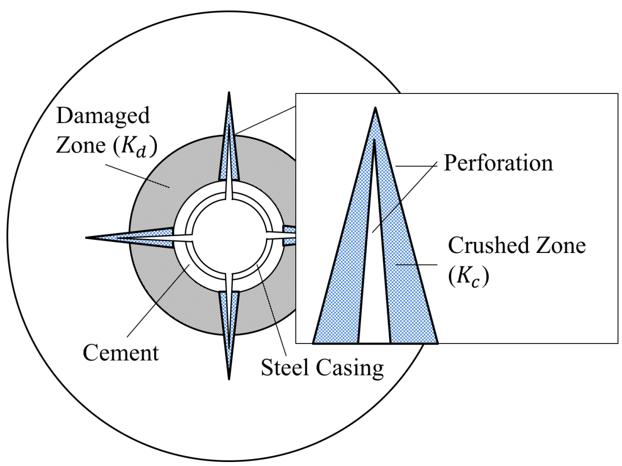


Figure 4.25: Crushed Zone in the Near-Wellbore Region

Figure 4.25 illustrates the crushed zone surrounding perforations. While creating perforations using a high powered perforation gun, the high compressive stress placed on the rock creates a crushed zone surrounding the perforation holes. This leads to a region of significantly reduced permeability (K_C). Pucknell and Behrmann (1991) reported that the crushed zone reduces the permeability in the range of 50% to 80%.

In a perforated well, the perforations do not open up the whole formation. The reservoir fluid has to flow with the flow lines converging near the penetrated area at the wellbore. The convergence of the flow lines near the wellbore causes an additional pressure drop near the wellbore, which in turn, creates a convergence skin. Streamlines directly demonstrate the flow patterns. Streamlines close to the perforations go directly into the perforation tunnel. Streamlines opposite of the perforations bypass the casing toward the perforations, as shown in Figure 4.26.

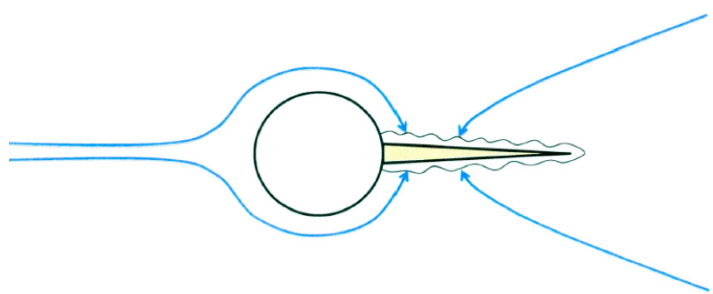


Figure 4.26: Convergence Effect in the Near-Wellbore Region (Skinner, 2011)

4.3.2 Model Representations

The perforations are distributed around the wellbore as shown in Figure 4.27. The angle between two perforations is called the phase angle; the perforation spacing is the vertical distance between two perforations; the perforation length is the length of a perforation tunnel from the wellbore. Perforations are used in cased and cemented wells. The casing and cement is a non-permeable ring that disconnects the well and the reservoir formation. No fluid can flow through this ring. Therefore, perforations are created to provide communication channels between the wellbore and the formation using a perforation gun with high powered shaped charges.

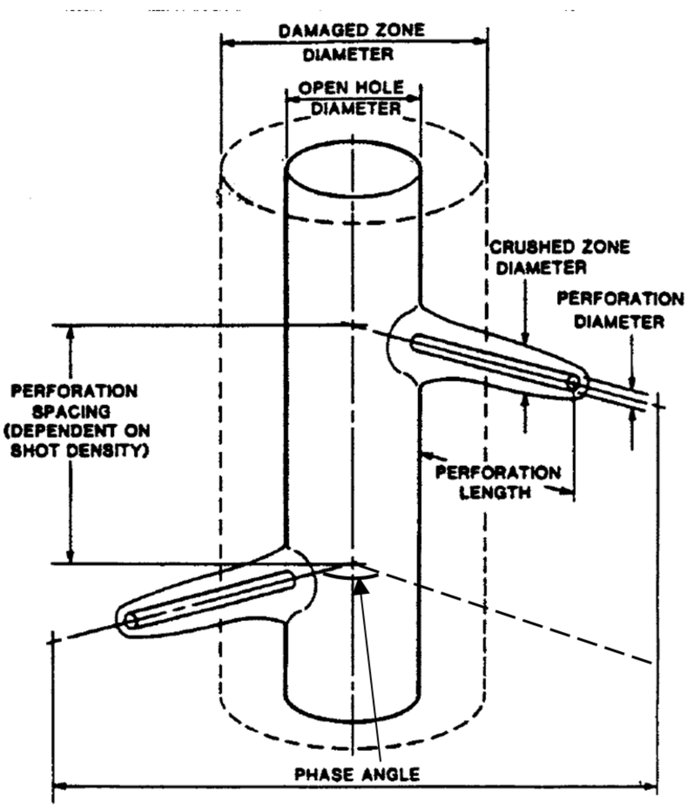


Figure 4.27: Perforation Geometry (Karakas and Tariq, 1991)

Figure 4.28 illustrates a radial grid used in streamline simulation representing a perforated wellbore. The casing is a non-permeable ring between the well and the formation, hence the transmissibility between the innermost ring and the wellbore is assigned the value of zero except inside the perforation. This approach accurately describes the real flow process. All streamlines avoid the casing as the flow converges towards the perforations. The crushed zone is represented by a set of grid blocks with a low permeability (K_c) layer adjacent to the perforation holes (Figure 4.28).

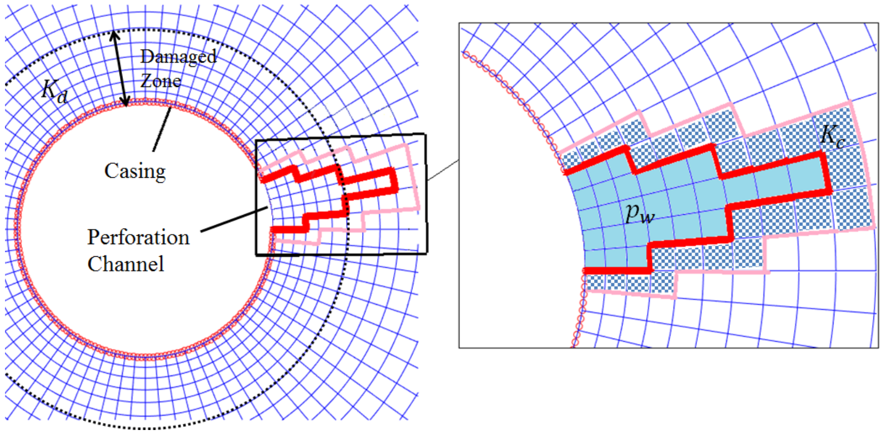


Figure 4.28: Model Representations in the Near-Wellbore Region

Perforations are the communication channels between a cased wellbore and a formation. In this work, the pressure for the grid blocks adjacent to casing inside the perforation is set equal to the wellbore pressure p_w.

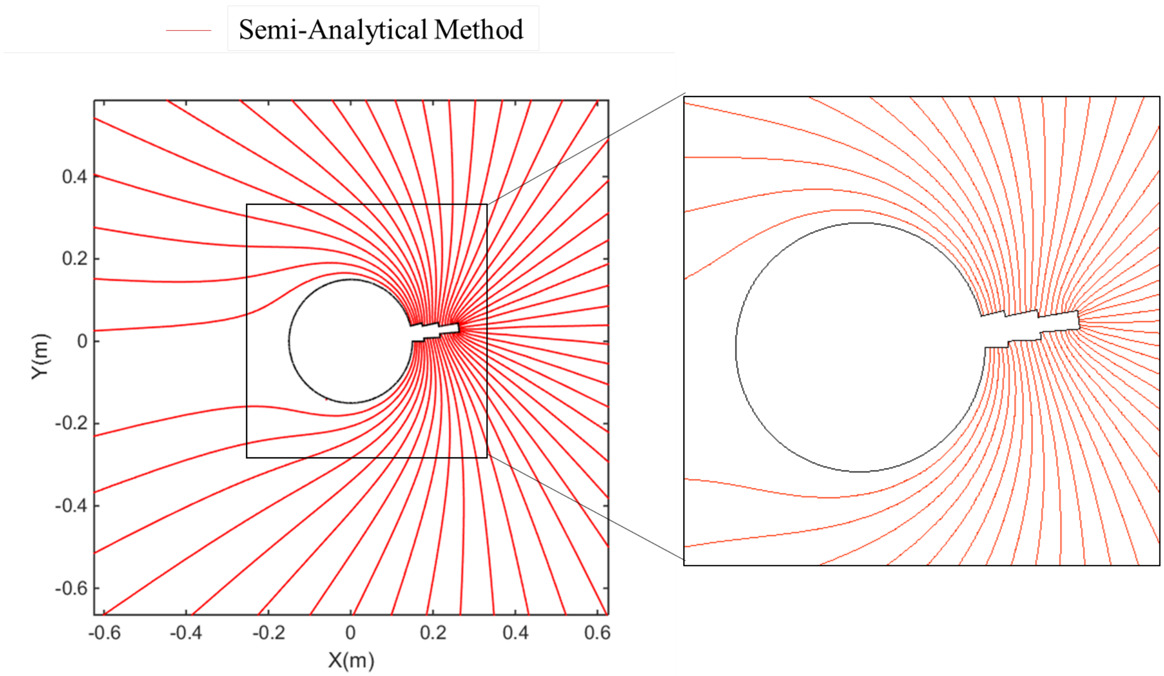
4.3.3 Skin Calculation in a Two-Dimensional Perforated Well

In a perforated well, the reservoir fluid can only flow into the well through the perforation channel. The reservoir permeability for this case is homogeneous and equal to 1 *Darcy*. The inner and outer boundaries have constant pressures of 280 *bar* and 300 *bar*, respectively. The inner-radius is 0.15 m and the outer radius is 20 m. The perforation diameter is 0.028 m, the perforation length is 0.102 m. As a simplification, we do not consider the crushed zone in this case. Both Pollock's method and the present method are applied to trace streamlines.The details of the parameters used in this case are shown in Table 4.7.

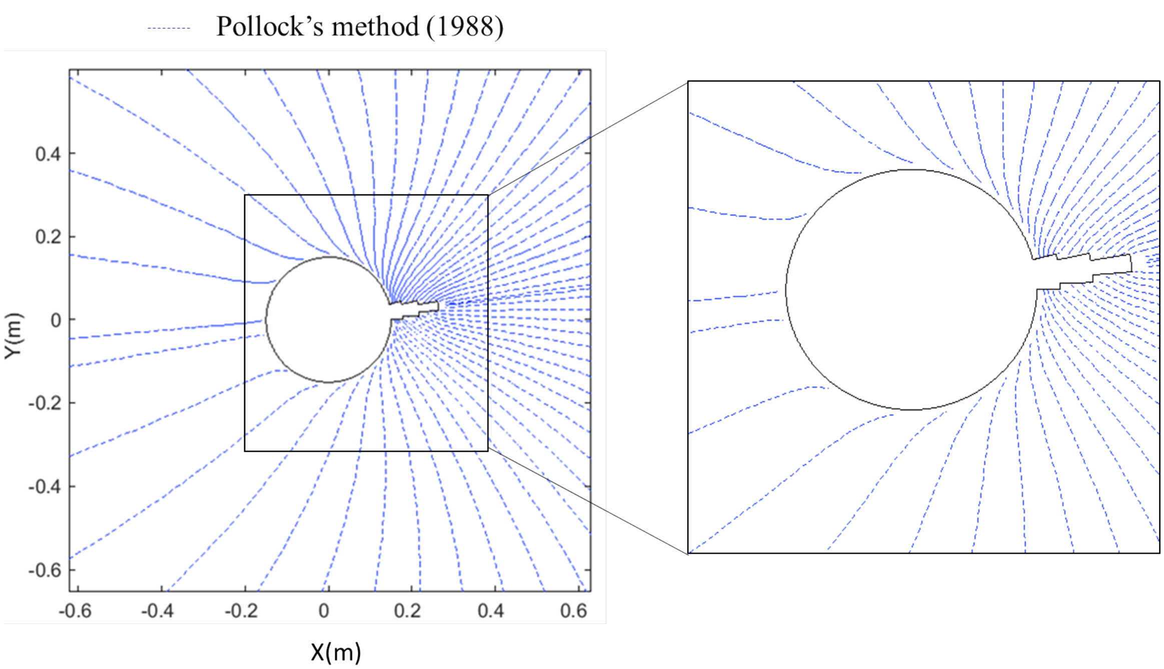
Table 4.7: Parameters used for 2D Perforated Well

Parameters	**Units**	**Values**
Wellbore Radius	m	0.150
External Radius	m	20
Perforation Radius	m	0.028
Perforation Length	m	0.102
Damaged Zone Radius	m	0.207
Damaged Zone Permeability	m^2	0.5×10^{-12}
Bulk Permeability	m^2	1×10^{-12}
Wellbore Pressure	Pa	250×10^5
External Pressure	Pa	300×10^5
Oil Viscosity	$Pa \cdot s$	0.8×10^{-3}
Radial Blocks		150
Tangential Blocks		100

For the near-wellbore region, the results in Figure 4.29 shows that Pollock's method fails to simulate the perforation case since most streamlines stop at the casing. The present method gives more physically reasonable results with streamlines bypassing the casing thereby causing convergence skin. Therefore, in order to obtain correct results, the present semi-analytical method should be applied for perforated well completions.



(a) Streamlines Traced by Present Semi-analytical Method



(b) Streamlines Traced by Pollock's Method

Figure 4.29: Streamline Traced in Perforated Well - 1 Perforation

Having generated the streamlines, we bundle them into stream-tubes, and then apply Equation 4.23 to calculate the flow rate in each stream-tube. We obtain the total flow rate for the entire reservoir ($q = 0.051m^3/s$) by adding the contributions for each stream tube. This flow rate is then used with the boundary pressures in Table 4.7 to calculate the skin factor using Equation 4.22, which is 2.77. We also use a classical skin calculation method (Karakas and Tariq, 1991) reviewed in **Appendix B** to calculate the skin factor; the value obtained is 3.45. Karakas-Tariq model is the stardard method used by the industry.

Next, we use the same parameters to generate a perforation case which has four perforation holes by using the present semi-analytical streamline simulation method. The results are shown in Figure 4.30. As can be seen from the figure, the streamlines are symmetric. Streamlines are separated by stagnations (stapled), bypassing the casing toward the closest perforation. If we take a look at a quarter of the streamlines (from Symmetry line 1 to Symmetry line 2 indicated in Figure 4.30), streamlines launching close to the symmetry lines are more curved than the streamlines launching in between the symmetry lines. Streamlines launching from the center for the quadrant converge to the closest perforation.

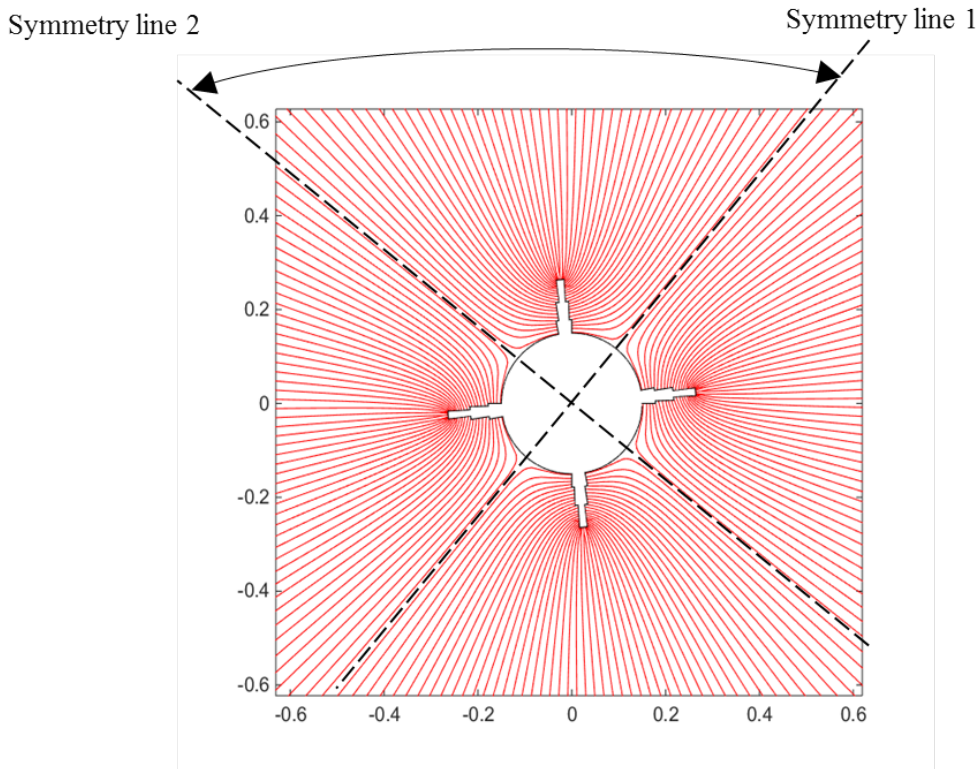


Figure 4.30: Streamline Traced in Perforated Well - 4 Perforations

4.3.4 Skin Calculation in a Three-Dimensional Perforated Well

Since Pollock's method can not accurately simulate the perforation case, only the present semi-analytical method is used in this section. Figure 4.31 shows a segment of the well along the z-direction. Two perforations were created at different heights in a three-dimensional reservoir with the phase angle for the perforations being 180°. There are two layers of streamlines starting at different heights at the wellbore radius. The red streamlines start at a higher level and the green streamlines start at a lower level. As can be seen from the figure, the streamlines are separated by a stagnant surface. The streamlines tend to approach the perforations in the z-direction, rather than bypassing the casing. Basically, how the streamlines approach the perforation holes depend on their location in the angular direction.

This 3D case is also used to study the effect of the perforation length on the total skin. The results are used to compare with a classical skin calculation method (Karakas and Tariq, 1991). The details of their parameters used in this case are shown in Table 4.8.

Table 4.8: Parameters used for 3D Perforated Well

Parameters	**Units**	**Values**
Wellbore Radius	m	0.15
External Radius	m	20
Perforation Radius	m	0.03
Radial Blocks		50
Tangential Blocks		100
Z-direction Layers		10
Crushed Zone Radius	m	0.045
Damaged Zone Length	m	0.207
Damaged Zone Permeability	m^2	0.5×10^{-12}
Crushed Zone Permeability	m^2	0.1×10^{-12}
Phase Angle	$^\circ$	180
Perforation Spacing	m	0.16
Bulk Permeability	m^2	1×10^{-12}
Wellbore Pressure	Pa	250×10^5
External Pressure	Pa	300×10^5
Oil Viscosity	$Pa \cdot s$	0.8×10^{-3}
Radial Blocks		50
Tangential Blocks		100
Z-direction Layers		10

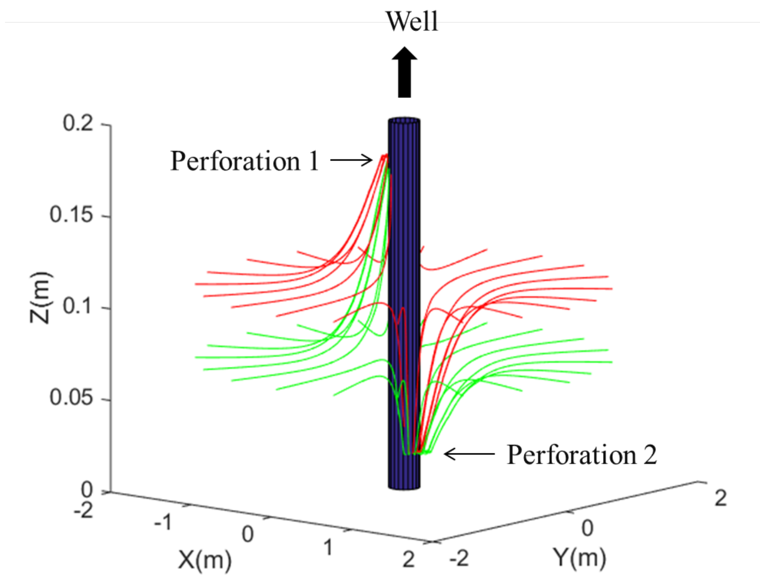


Figure 4.31: Streamlines Traced in 3D Perforated Well

The model geometry of a perforation and the corresponding geometry used in Karakas-Tariq method is shown in Figure 4.32.

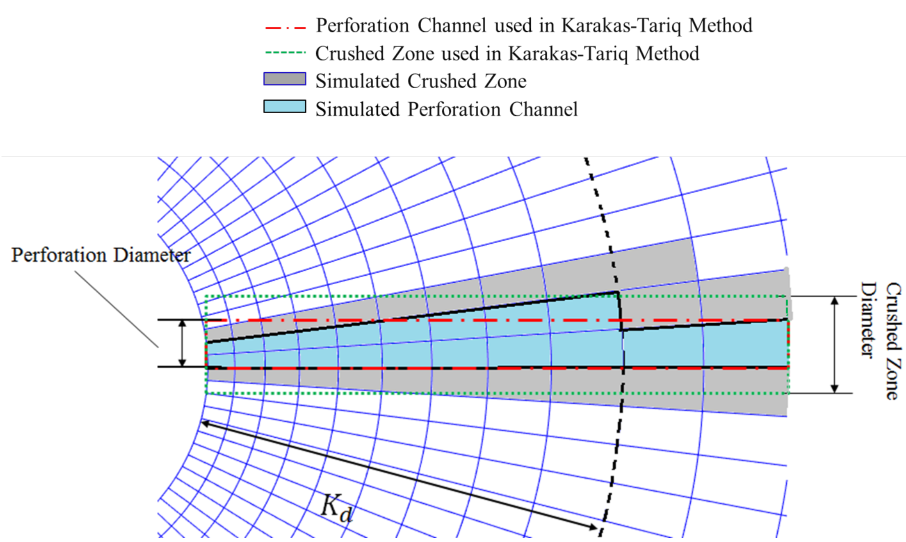


Figure 4.32: Perforation Parameters Representation

As illustrated in Figure 4.33, the skin calculated by the Karakas-Tariq method exhibits an unphysical behavior with increasing perforation length beyond the damaged zone. Skin obtained from the present semi-analytical method provides a more physically reasonable result, since skin monotonically decreases with increasing perforation length, while the Karakas-Tariq method creates an unphysical bump when the perforation length is close to the damaged zone outer boundary. The Karakas-Tariq methods is an industry standard. It is concluded that the new skin calculation procedure presented in this research thesis is superior both in accuracy and flexibility.

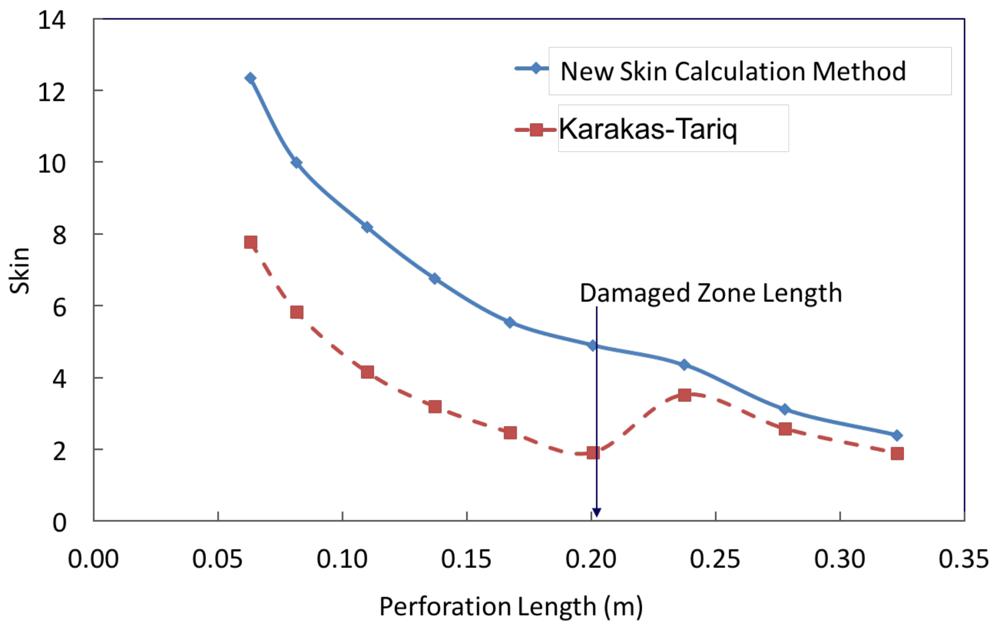


Figure 4.33: Effect of Perforation Length to Skin

4.3.5 Case Studies Conclusion

In a homogeneous reservoir, the present semi-analytical streamline simulation method provides identical TOF as the fully analytical solution while Pollock's method is hampered by errors.

Pollock's method leads to a systematic error in the near well-bore streamline tracing for the heterogeneous cases. As we observe in the 2D heterogeneous cases Pollock's method exhibits unrealistic behavior. Pollock's method also fails to trace the streamlines in perforated wells and in 3D heterogeneous case. However, the present semi-analytical streamline simulation method provides physically reasonable results in general and for perforated wells in particular. The present semi-analytical streamline simulation method provides both reasonable streamline paths and TOF values for the heterogeneous reservoirs. These errors in Pollock's method are severe because of the logarithmic (non-linear) pressure distribution.

The present semi-analytical streamline simulation method can be used to calculate perforation skin. The skin values obtained from the present semi-analytical method provides a more physically reasonable result than standard method. Unlike in the standard skin calculation method, which creates an unphysical bump when the perforation length is close to the damaged zone outer boundary, the new skin calculation method provides a monotonically decreasing skin value with increasing perforation length. Hence, the new skin calculation method is believed to be superior to existing models.

Chapter 5

Two-Dimensional Waterflooding Visualization Experiments

Waterflooding is a conventional secondary recovery method in oil production. In this research thesis, several waterflooding visualization experiments (James, 2012) are performed using unconsolidated glass bead filled macro-models in the radial geometry, shown in Figure 5.1. The purpose of these experiments is to visually observe what happens in the heterogeneous near-wellbore region during waterflooding. It also provides the physical data to demonstrate the history matching ability of the streamline simulation.

Reservoir waterfloods can be operated at constant injection rate or constant bottom hole pressure. In this research thesis, the laboratory scale waterflooding experiments are performed under constant differential pressure boundary conditions. The corresponding parameters such as the location of water displacement front, together with oil and water flow rates are recorded as functions of time. By applying the solution of the 3D Riemann problem (discussed in **Section 4.1**) in the stream tube simulation at constant pressure boundary conditions, the location of the waterfront at a specific time, the water breakthrough time, and flow rates can be obtained. By tuning the relative permeabilities, these simulated results are used to history match with the laboratory data.

In this chapter, the experimental set-up, design of experiments and experimental procedure are first introduced, followed by the comparisons between experiments and the history matched simulated results.

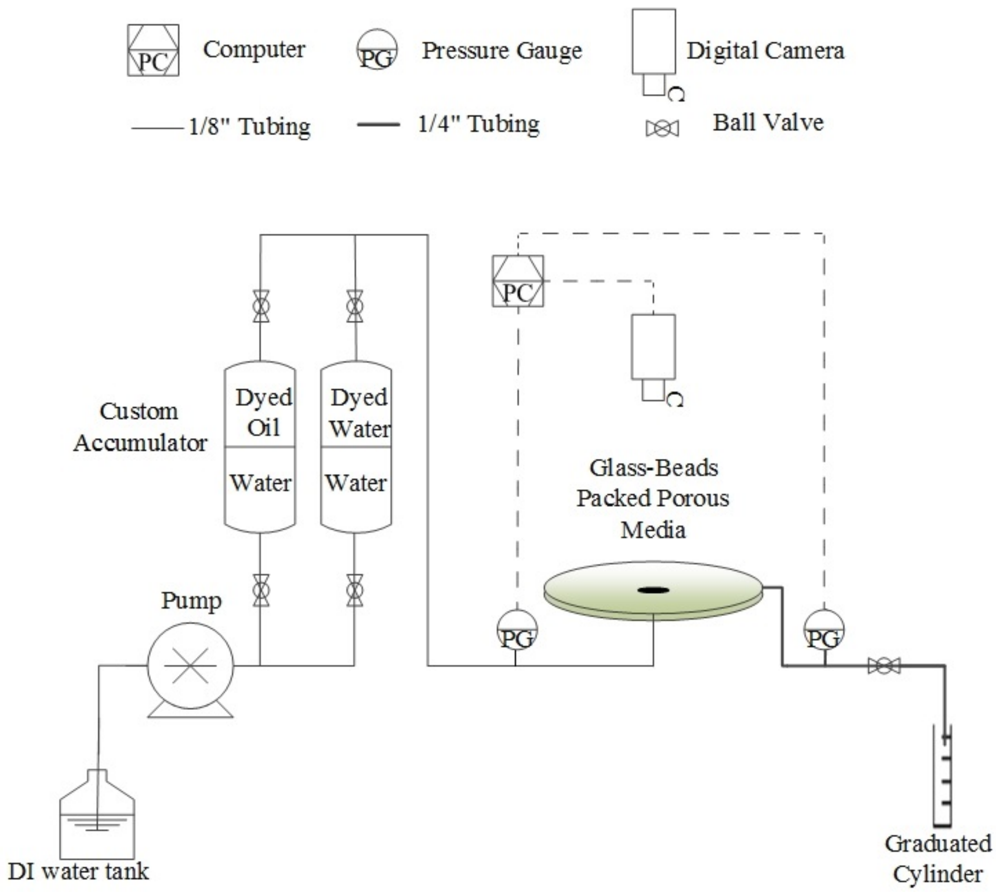
5.1 Experimental Set-up

The overall set-up for this experiment is shown in Figure 5.1. A custom glass-bead pack macro-model was designed for the radial glass cell (James 2012). A radial glass cell was also fabricated based on this design. Uniform-sized glass beads (BT-3) are filled into the glass cell to pack the porous media. An injection well is placed in the center of the porous media. A 1/4″ tubing with perforations is placed at the outer radial radial boundary as the production well. During the experiment, dyed oil and water are injected through two custom made accumulators into the porous media through the center point and produced from the outer boundary. Two OMEGA PX 409-100AUSB pressure gauges with 0.001 psi precision at the inlet and outlet are used to measure the pressure at the boundaries. Pressure at the outlet is kept at atmospheric pressure, the injection pressure is kept constant by adjusting the flow rate of the pump. Hence, the pressure difference between the inlet and outlet is kept constant at 2.7 and 2.6 *psi* for homogeneous and heterogeneous experiments, respectively (pressure profile is shown in **Appendix D**). Five graduated cylinders

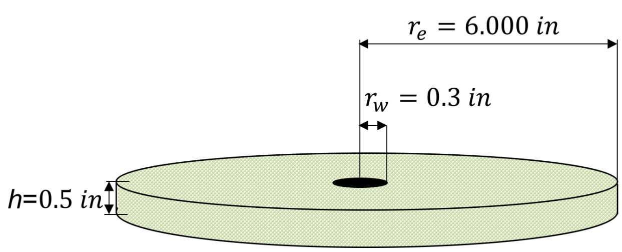
(50 ml and 10 ml) are used to measure the produced fluid volume. The images of the waterfronts are captured by using a Cannon Rebel XS digital camera with a resolution of 10.10 Megapixels. Table 5.1 describes in detail the experimental equipment and components used in the experiments.

Table 5.1: Apparatus List

Parameters	**Type**	**Quantity**
Pump	ISCO 500D	1
Accumulator	Custom Made (2 L)	2
Computer	IBM Think Station	1
Camera	Cannon Rebel XS	1
Light Box	Custom made	1
Graduated Cylinder	50 ml, 10 ml	5
1/8″ 2-way Ball Valve	Swagelok® SS-83KF2	5
Female Branch Tee, 1/4″	Swagelok® SS-400-3TTF	6
Union Cross, 1/8″	Swagelok® SS-200-4	2
Custom made bracket	Custom Made	2
1/8″ tubing	Swagelok® SS-T2-S-028-20	15
1/4″ tubing	Swagelok® SS-T4-S-035-6ME	2
Pressure Gauge	OMEGA PX 409-100AUSB	2



(a) Overall Setup



(b) Porous Media Schematic

Figure 5.1: Experimental Schematic for 2D Water Flooding Visualization Experiment

5.2 Design of Experiment

Design of experiment (DOE) is a method for systematically analyzing the relationship between the factor(s) and the response(s) of an experiment. It is the process of planning experiments so that appropriate data can be analyzed by statistical methods. The DOE procedure mainly involved the following tasks: selecting one or more independent variables, manipulating their effects on one (or some) dependent parameters, and determining the sensitivity of dependent variable(s) upon changing the independent parameters (Montgomery and Runger, 2006). The guideline for design experiments are:

1. Define the objectives.
2. Choice of factors and responses.
3. Select of experimental design.
4. Performing the experiment.
5. Analysis of data.
6. Obtain conclusion and recommendations.

5.2.1 Objectives

Although the analytical solution exists for the homogeneous reservoir, we still need to perform the homogeneous experiment for two reasons: to measure the absolute permeability, and to obtain a history matched relative permeability to independently simulate the heterogeneous experiments. The purpose of the heterogeneous experiments is to visually observe what happens in the heterogeneous near-wellbore region during waterflooding. The experimental results also provide the physical data to demonstrate the history matching ability of the streamline simulation.

5.2.2 Choice of factors

In the 2D waterflooding experiments the factors can be:

- Reservoir type: reservoir type can be homogeneous and heterogeneous.
- Permeability: permeability is depending on the size of glass beads used to pack the porous media. Since we used a tubing with perforations to perform as the production ring at the outer boundary, we have to make sure the unconsolidated glass beads can not flow out of the porous media through the perforation holes. In other words, the diameter of glass beads must larger than the perforation diameter. However, if the diameter of the glass beads is too large, it will result in a very short breakthrough time. Hence, BT-3 glass is the most appropriate choice.
- Fluid viscosity: in order to obtain a stable front, the viscosity of the displaced fluid should be relatively small. Hence, varsol oil is used in the experiments. As can be seen from the mobility ratio obtained in Equation 5.14, the choice of varsol ensures a stable front as designed.
- Pressure difference: the pressure difference should be kept in a small range to prevent the expanding of the porous media shell.

In the 2D waterflooding experiments, direct responses are waterfront location, the breakthrough time and the volume of the produced fluid.

5.2.3 Select of experimental design

When designing the experiments, we have to consider limitations mentioned in **5.2.2** for the experimental model. A total of 3 experimental runs are designed to be performed in order to study effects of heterogeneity in the near-wellbore region.

5.3 Experimental Procedures

Three main stages are involved in the visualization experiment: 1. Initial imbibition; 2. Primary drainage; 3. Waterflooding (Secondary imbibition).

In the initial imbibition process, injected water displaces the air inside the porous media. In this step, porosity and absolute permeability can be calculated (described in **5.4.3** and **5.4.4**). Once the air is eliminated from the system, the injection flow rate and the pressure difference can be used to calculate the absolute permeability. Porosity can be obtained since the total amount of water injected and produced are recorded.

In the primary drainage process, oil is injected and used to displace the water inside the system. At the end of this stage, a reservoir with oil and connate water inside the pore channel is formed. At this point, the connate water saturation and the initial oil in place can be obtained (described in **5.4.5**).

During waterflooding, water is injected into the model to displace the oil at connate water saturation. The pressure difference between the inlet and outlet is kept constant and recorded with time. The corresponding inflow rate and outflow rate are recorded as well. The waterfront movement as a function of time is captured by the camera. Breakthrough time is also recorded as the time when the first water droplet appears in the production ring. In this process, the flow rate is not smooth, hence relative permeabilities cannot be accurately measured. The relative permeabilities for oil and water are considered as uncertainties and are determined by the history matching method using the semi-analytical streamline simulator. The residual oil saturation is calculated at the end of the waterflooding process (described in **5.4.6**). The steps for the water flooding process are:

1. Change the inflow fluid to dyed water.
2. Set up the experiment according to the Figure 5.1.
3. Determine the differential pressure.
4. Set up the alarm for the pressure boundaries on the computer.
5. Start the pump with an initial injection value.
6. Record the displacement front using a camera before breakthrough once per minute.
7. Adjust the flow rate to keep a constant differential pressure.
8. Record the boundary pressures and the flow rate.
9. Measure the oil and water flow cumulative production.
10. Shut down the pump when all dyed water has been injected.

5.4 Properties Characterization

5.4.1 Porous Media Dimensions

Before the flooding process, the porous media dimensions are measured and shown in Table 5.2. Porous media thickness are changing in each experiment since they are dependent on the pressure we pack the glass cell with. Data for the porous media heights are shown in each experiment separately.

Table 5.2: Porous Media Dimensions

Parameters	**Values**	
	SI Units	Lab Units
Wellbore Radius (r_w)	0.0079 m	0.003 in
External Radius (r_e)	0.1524 m	6.000 in

5.4.2 Fluid Viscosities

Fluid viscosities are measured by a Cambridge PVT Viscometer. Viscosities for water and oil are shown in Table 5.3.

Table 5.3: Fluid Viscosities

Fluids	Values	
	SI Units	Lab Units
Water	$1 \times 10^{-3}\ Pa \cdot s$	$1.00\ cP$
Oil	$1.19 \times 10^{-3}\ Pa \cdot s$	$1.19\ cP$

Next, we will show the absolute permeability, porosity, connate water saturation and residual oil saturation measurement procedures and calculation methods. The sample calculations and the error analysis are shown in **Appendix D**.

5.4.3 Absolute Permeability Measurement

The same uniform-sized glass beads (BT-3) are used to pack the porous media. The permeability measurement for this kind of glass beads is performed in the homogeneous model. In the initial imbibition process, once the air is eliminated from the system, the injection flow rate and the pressure difference can be used to calculate the absolute permeability by using Equation 4.19. Once air is displaced from the system, the permeability test starts:

1. Set the pump to a constant flow rate q.
2. Measure the flow rate at the outlet; make sure it reaches steady state.
3. Record the pressure difference Δp when the outlet flow rate is equal to injection rate q.

4. Repeat the measurement for several different flow rates.

According to Equation 4.19, permeability can be calculated by:

$$K = \frac{q\mu ln\left(\frac{r_e}{r_w}\right)}{2\pi h \Delta p}, \tag{5.1}$$

where K is the permeability, r_e and r_w are the radius for an external boundary and the wellbore, respectively, q is the flow rate and Δp is the pressure difference between the inlet and the outlet.

For each flow rate, one permeability value can be obtained (**Appendix D**). Permeability values obtained from each flow rate are averaged for further calculation. The permeability value ($1.58 \pm 0.03 \times 10^{-12}\ m^2$) measured from the homogeneous glass-beads macro-model is used as the bulk permeability in the heterogeneous experiments. Here, the permeability value measured from the glass-beads macro-model is smaller than the BT-3 glass-beads permeability ($408 \pm 67 \times 10^{-12}\ m^2$) used in Sohrab (2010). Sohrab used the falling head measurement technique to measure the permeability. However, in our glass-beads macro-model, the injection well in the center and the production ring at the outer ring are not fully open hole. The partially perforated production ring has a large effect on the permeability, hence we can not use the permeability value obtained from the falling head measurement. We have to treat these boundaries as part of the reservoir, hence the value we obtained is smaller than that in Sohrab (2010). In order to obtain the accurate flow rate, the measured absolute permeability ($1.58 \pm 0.03 \times 10^{-12}\ m^2$) for our porous media is applied in this research thesis.

5.4.4 Porosity Measurement

The porosity of the unconsolidated glass-beads pack is measured using the fluid saturation method in the imbibition process. The total injection volume is V_1; the total production volume is V_2. The total bulk volume V_t is calculated by the shell dimension. The porosity is:

$$\phi = \frac{V_1 - V_2}{V_t}. \tag{5.2}$$

There is another way to measure the porosity. During this process, water is injected to the dry porous media shell and fully saturates the reservoir. The mass of the dry porous media shell M_1 and mass of the porous media shell saturated with water M_2 are measured, respectively. The density of water ρ_w is known. The total bulk volume V_t is calculated by the shell dimension. Porosity can be calculated by:

$$\phi = \frac{M_1 - M_2}{\rho_w V_t} = \frac{V_1 - V_2}{V_t}. \tag{5.3}$$

5.4.5 Connate Water Saturation Measurement

At the end of primary drainage process, the connate water saturation can be obtained. The test procedure and the calculation of the connate water saturation are:

1. Change the injection fluid from water to oil and start the pump to inject the oil at a constant flow rate (5 ml/min).

2. Continue injection until no water is produced at the outlet.

3. Read the volume of injected oil V_{oi} from the pump and read the volume of oil

produced in the graduated cylinder V_{op}. Initial oil in the porous media is:

$$V_{oil} = V_{oi} - V_{op}. \tag{5.4}$$

4. The pore volume was known from the imbibition stage. The volume of connate water that remains in the porous media is:

$$V_{wc} = V_1 - V_2 - V_{oil}. \tag{5.5}$$

5. Calculate the connate water saturation:

$$S_{wc} = \frac{V_1 - V_2 - V_{oil}}{V_1 - V_2}, \tag{5.6}$$

where V_1 and V_2 are the volumes of water injected and produced in the initial imbibition process, respectively; V_{oil} is the volume of initial oil in place; and S_{wc} is the connate water saturation.

5.4.6 Residual Oil Saturation Measurement

Residual oil saturation is calculated at the end of the waterflooding process. At the end of this stage, the volume of water injected (V_{wi}) and the volume of oil produced (V_{op2}) are recorded from the pump and the graduate cylinder at the outlet, respectively. The volume of oil remaining in the system is:

$$V_{or} = V_{oil} - V_{op2}. \tag{5.7}$$

The residual oil saturation then can be calculated as:

$$S_{or} = \frac{V_{or}}{V_1 - V_2}, \tag{5.8}$$

where V_1 and V_2 are the volume of water inject and produced in initial imbibition process, respectively, S_{or} is the residual oil saturation.

5.5 History Matching of Experimental Results

In this section, we will first describe the history matching approach and demonstrate the respective results for the homogeneous experiment. Then, the Corey model obtained from the homogeneous experiment is used to independently simulate the heterogeneous experiments and compare with the experimental results. Finally, the history matched results for the heterogeneous experiments are demonstrated.

There are many models available to history match the relative permeabilities. With all the available models, the choice of which model we should apply to history match is extremely important. Corey model can be incorporated to the 3D Riemann solution described in **4.1**. The flow rate can be obtained as a function of time, which can be compared with the experimental flow rate. The history matched homogeneous Corey model is used to independently validate the heterogeneous experiments. This approach ensures the accuracy of the developed streamline model.

5.5.1 Approach Description

For the homogeneous experiment, the area of each stream tube along radii can be explicitly determined. By applying the 3D Riemann solution along stream tubes as discussed in **Section 4.1** and tuning the relative permeabilities, the exact solution of the displacement process such as breakthrough time and flow rate in the homogeneous reservoir are obtained.

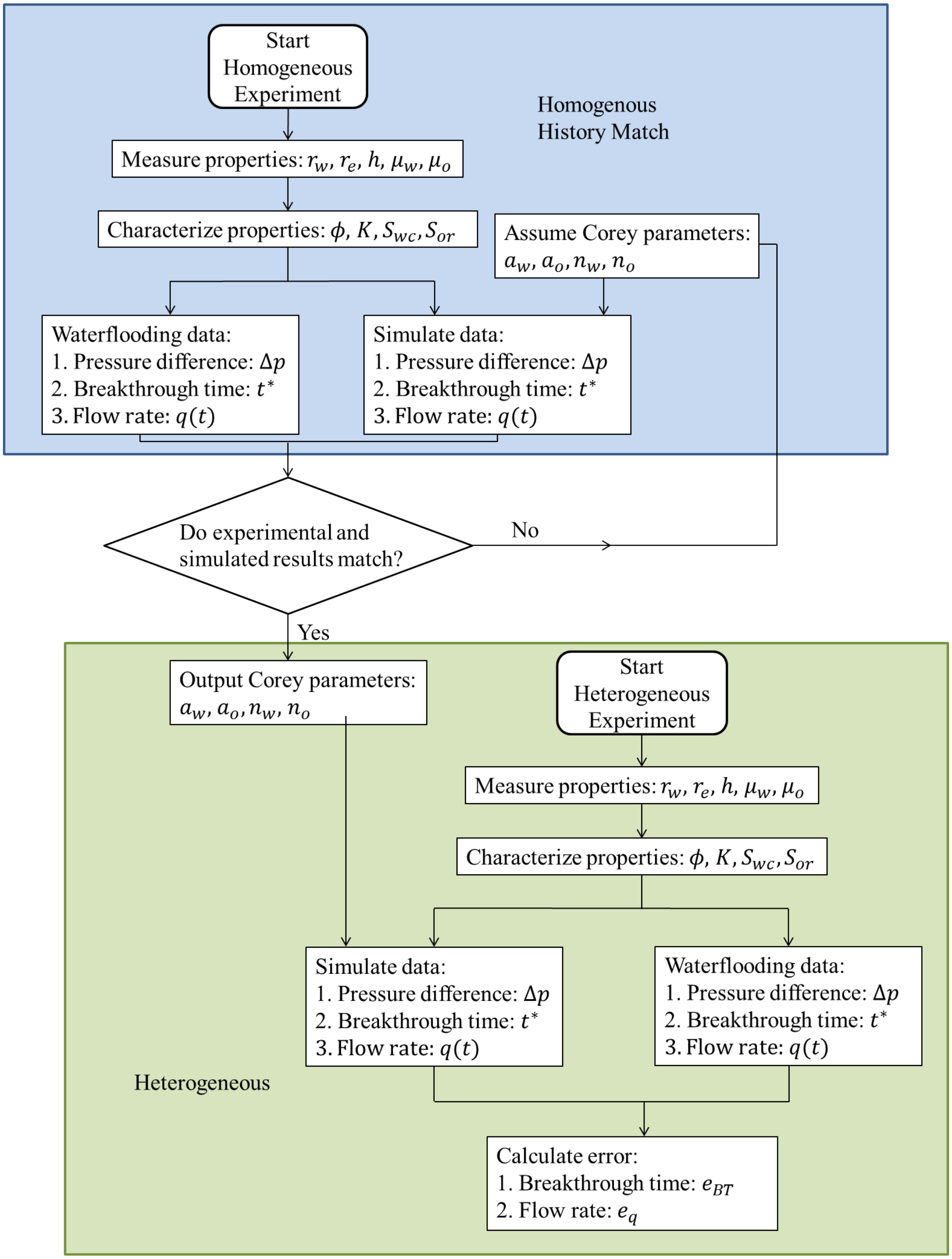


Figure 5.2: History Matching Approach

Only one experiment with homogeneous porous media was performed. Two heterogeneous experiments are used to study the flow behavior in the near-wellbore region. The main purpose of the homogeneous experiment is to measure the absolute permeability of the glass-beads macro-model, which is also used as the bulk permeability in the heterogeneous experiments. The history matched relative permeabilities from the homogeneous experiment are used to validate if they can be applied to the heterogeneous experiments. The validation approach is shown in Figure 5.2.

The relative permeability curves are estimated by matching the Corey relative permeability model (1954):

$$K_{rw} = a_w \left(\frac{S_w - S_{wc}}{1 - S_{or} - S_{wc}} \right)^{n_w}, \tag{5.9}$$

$$K_{ro} = a_o \left(\frac{1 - S_{or} - S_w}{1 - S_{or} - S_{wc}} \right)^{n_o}, \tag{5.10}$$

where K_{rw} and K_{ro} are the relative permeabilities for water and oil, respectively; S_w is the water saturation; S_{wc} and S_{or} are the connate water saturation and the residual oil saturation, respectively; and the exponents n_o and n_w range from 1 to 6.

Trial and error is applied to history match the relative permeabilities. The detailed procedure is:

1. Measure the homogeneous porous media dimensions r_w, r_e, h and fluid viscosities μ_w and μ_o.

2. Characterize the homogeneous porous media absolute permeability K (described in **5.4.3**), porosity ϕ (described in **5.4.4**) and connate water saturation S_{wc} (described in **5.4.5**), residual oil saturation S_{or} (described in **5.4.6**).

3. Assume values of a_w, a_o, n_w and n_o in Corey model described in Equation 5.9 and 5.10.

4. Apply Equation 4.5 to calculate the flow rate $q(t)$.

5. Determine breakthrough time by integration of Equation 4.7.

6. Calculate the post breakthrough flow rate by using Equation 4.10 if the simulated breakthrough time is the same as the experimental breakthrough time. If breakthrough times are different, go back to 3.

7. Calculate the flow rate error between the simulated and experimental results.

8. Output values of a_w, a_o, n_w and n_o in the Corey model for later use if the flow rate relative error between the simulated and experimental results at the end is less than 10%. If the error is larger than 10%, go back to 3. The relative error is defined as:

$$e_q = \frac{1}{N} \sum \frac{|q_e - q_s|}{q_e}, \tag{5.11}$$

   where q_e and q_e are the experimental and simulated flow rate, respectively, and N is the number of measurements.

9. Apply the values of a_w, a_o, n_o and n_w in the homogeneous experiment and the heterogeneous properties in to Equation 4.5 and 4.7 to simulate the breakthrough time.

10. Compare the simulated breakthrough time with the experimental breakthrough time for heterogeneous experiments.

5.5.2 Homogeneous Reservoir Experiment and History Matched Results

The fluids and reservoir properties are measured according to the procedures described in **5.4**, and the results are shown in Table 5.4. The simulator applied the same values to history match the actual waterflooding process. This is the first time a radial water flooding experiment is demonstrated to match an analytical solution for such flow, and with excellent agreement.

The experimental flow rates and history matched flow rates are shown in Figure 5.3. The inlet flow rates fluctuate because they are manually changed to keep constant differential pressure between the inlet and the outlet. Hence, we cannot directly use the experimental data to calculate the relative permeabilities. However, the history matched flow rate provides a smooth result and history matched relative permeabilities.

Table 5.4: Experimental Parameters used for the Homogeneous Experiment

Parameters	**Homogeneous Experiment**	
	SI Units	Lab Units
Wellbore Radius	$0.0079\ m$	$0.3\ in$
External Radius	$0.1524\ m$	$6.000\ in$
Reservoir Thickness	$0.0119\ m$	$0.5\ in$
Porosity	0.454	0.454
Differential Pressure	$17249.9\ Pa$	$2.7\ psi$
Bulk Permeability	$1.58 \times 10^{-12}\ m^2$	$1.58\ Darcy$
Connate Water Saturation	0.277	0.277
Residual Oil Saturation	0.166	0.166
Oil Viscosity	$1.19 \times 10^{-3}\ P \cdot s$	$1.19\ cP$
Water Viscosity	$1 \times 10^{-3}\ Pa \cdot s$	$1.00\ cP$
Breakthrough Time	$33\ min$	$33\ min$

Figure 5.4 and Table 5.5 demonstrate the cumulative production rate for the experimental and history matched results. In Figure 5.4, the red color represents data for oil, blue color represents data for water and black color represents the data for total flow rate. The dots are the experimentally measured values and the solid lines are the history matched results. The breakthrough times for the experimental and the simulation are both 33 min. At the breakthrough time, the recovery difference is only 0.06%. After breakthrough, there is minor variation. As can be seen from Table 5.5, the variations for the cumulative water produced decrease less with increasing time. At 48 min, the difference in cumulative oil produced between the experimental and

history matched results is 0.49% and the difference of the cumulative water produced is 2.65%.

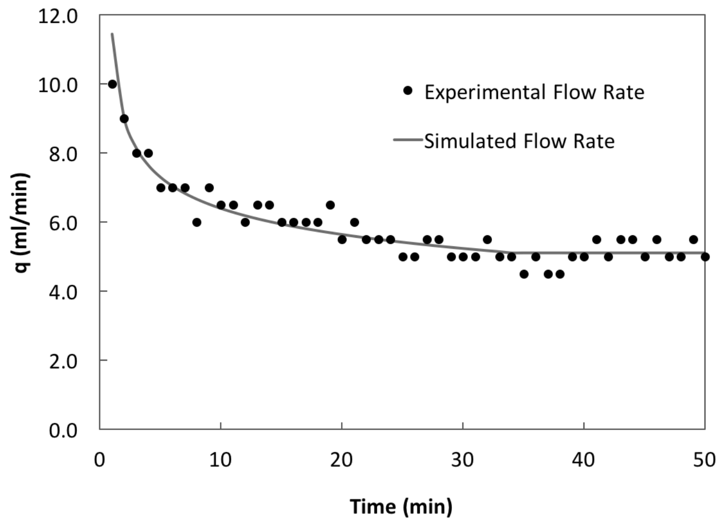


Figure 5.3: Flow Rate Comparison for the Homogeneous Experiment

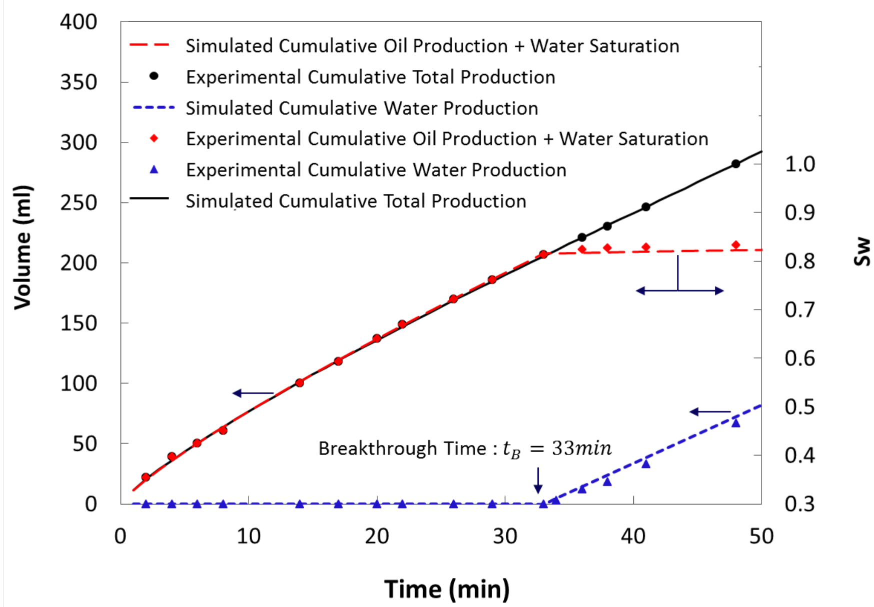


Figure 5.4: Cumulative Production Comparison for the Homogeneous Experiment

Table 5.5: Data for Cumulative Production for the Homogeneous Experiment

Time(min)	**History Matched Cumulative Production** (ml)			**Experimental Cumulative Production** (ml)			**Error** (%)		
	Total	Oil	Water	Total	Oil	Water	Total	Oil	Water
2	20.7	20.7	0	21.0	21.0	0	1.22	1.22	-
4	36.7	36.7	0	38.0	38.0	0	3.33	3.33	-
6	48.8	48.8	0	50.0	50.0	0	2.41	2.41	-
8	64.8	64.8	0	62.0	62.0	0	4.59	4.59	-
14	102.6	102.6	0	100.0	100.0	0	2.58	2.58	-
17	120.3	120.3	0	118.0	118.0	0	1.93	1.93	-
20	137.4	137.4	0	138.0	138.0	0	0.44	0.44	-
22	148.5	148.5	0	149.0	149.0	0	0.31	0.31	-
26	170.3	170.3	0	170.0	170.0	0	0.17	0.17	-
29	186.2	186.2	0	186.0	186.0	0	0.08	0.08	-
33	206.9	206.9	0	207.0	207.0	0	0.06	0.06	-
36	222.3	207.6	14.6	220.0	209.0	11.0	1.03	0.55	33.20
38	232.5	208.1	24.4	230.0	210.0	20.0	1.10	0.79	22.13
41	247.9	208.8	39.1	246.0	211.0	35.0	0.78	0.92	11.72
48	283.9	210.5	73.4	282.0	215.0	71.0	0.66	0.49	2.65

The history matched relative permeabilities are shown in Figure 5.5 (Equation 5.12 and 5.13).

$$K_{rw} = 0.258 \left(\frac{S_w - 0.277}{0.557} \right)^4 , \tag{5.12}$$

$$K_{ro} = 0.855 \left(\frac{0.834 - S_w}{0.557} \right)^2 . \tag{5.13}$$

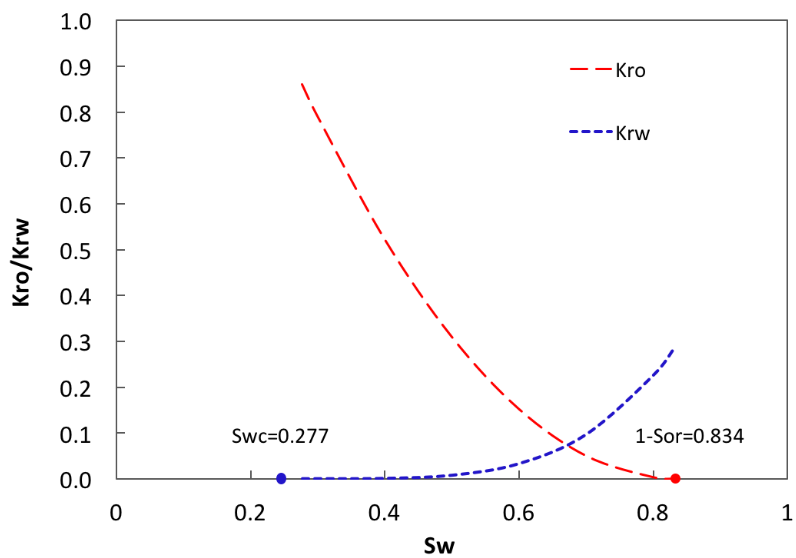


Figure 5.5: History Matched Relative Permeabilities for the Homogeneous Experiment

Figure 5.6 shows the history matched displacement fronts after 1 minute of injection and for every five minutes thereafter. We can observe that the simulated displacement fronts are concentric circles for the homogeneous reservoir. As the front moves closer to the production ring (outer boundary), the displacement front moves in a shorter distance in the same time interval. This radial frontal movement has not been analyzed before and is highly non-trivial in its analytical calculation.

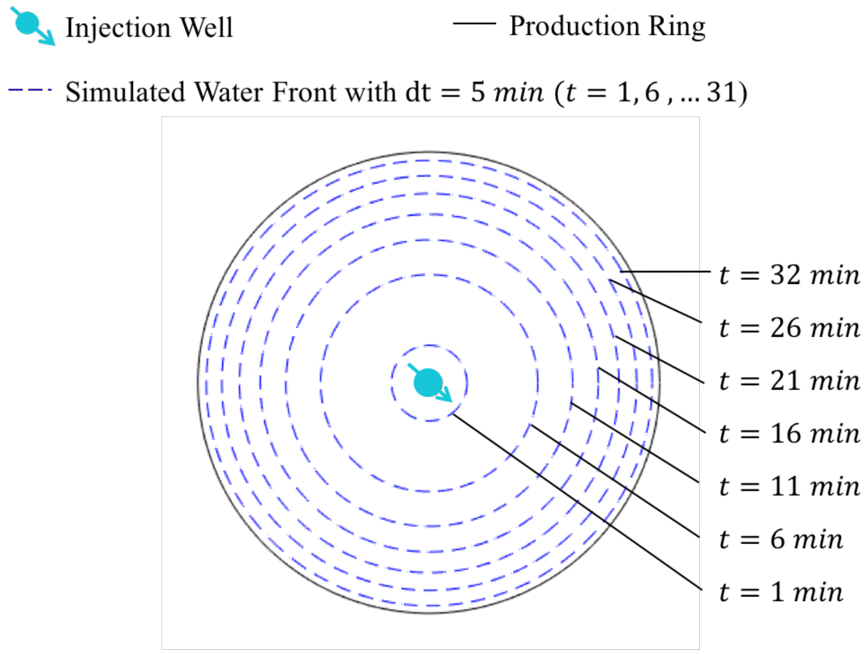


Figure 5.6: History Matched Displacement Fronts for the Homogeneous Experiment

The camera is used to capture the overall shape of the fronts at each time interval. Figure 5.7 shows the experimental displacement and the history matched fronts every three minutes. In the experiment, to visually differentiate the waterfront, oil and water are dyed red and blue, respectively. The injection well is at the center of the glass-beads packed porous media. Blue water is injected at the center, and fluid is produced at the outer ring. The displacement front is moving from the center towards the production ring as a stable front because the mobility ratio is favorable. The value of mobility ratio shown in Equation 5.14:

$$M = \frac{\frac{K_{rw}(S_{or})}{\mu_w}}{\frac{K_{ro}(S_{wc})}{\mu_o}} = 0.36. \tag{5.14}$$

The black lines are the history matched waterfront with the corresponding injection time. This homogeneous experiment shows the concentric movement of the waterfront. The agreement between the history matched waterfronts and the experimental waterfront is excellent. At later times, experimental fronts are not as stable as at earlier times. This is due to the technical limitations for the production ring created in the experiment. A densely and randomly perforated tube with hole size smaller than the size of glass beads connects to the glass-beads. Four bigger holes in the perforated tube which is used as the production ring in the experiment are connected to the outlet. In the experiment, the production ring is partially perforated, while in simulation, it is treated fully open.

A non-linear regression could be employed to the history matched results. However, we have measured the fractional flow function of water and the total rate as a function of time (Figure 5.4). For a regression process, we would need $f(S)$ as s function of water saturation. The determination of which would require using the calculus of variation, since $f(S)$ appears in the integral equation (Equation 4.6). This is beyond the scope of this research thesis. According to the history matched results from the homogeneous experiment, we can obtain the following results: using regression to obtain a best fit with a Corey model is deemed unnecessary since the trial and error applied in the homogeneous experiment provides a very good match after a few trials. This also, therefore, indicates that the Corey model is adequate for these experiments.

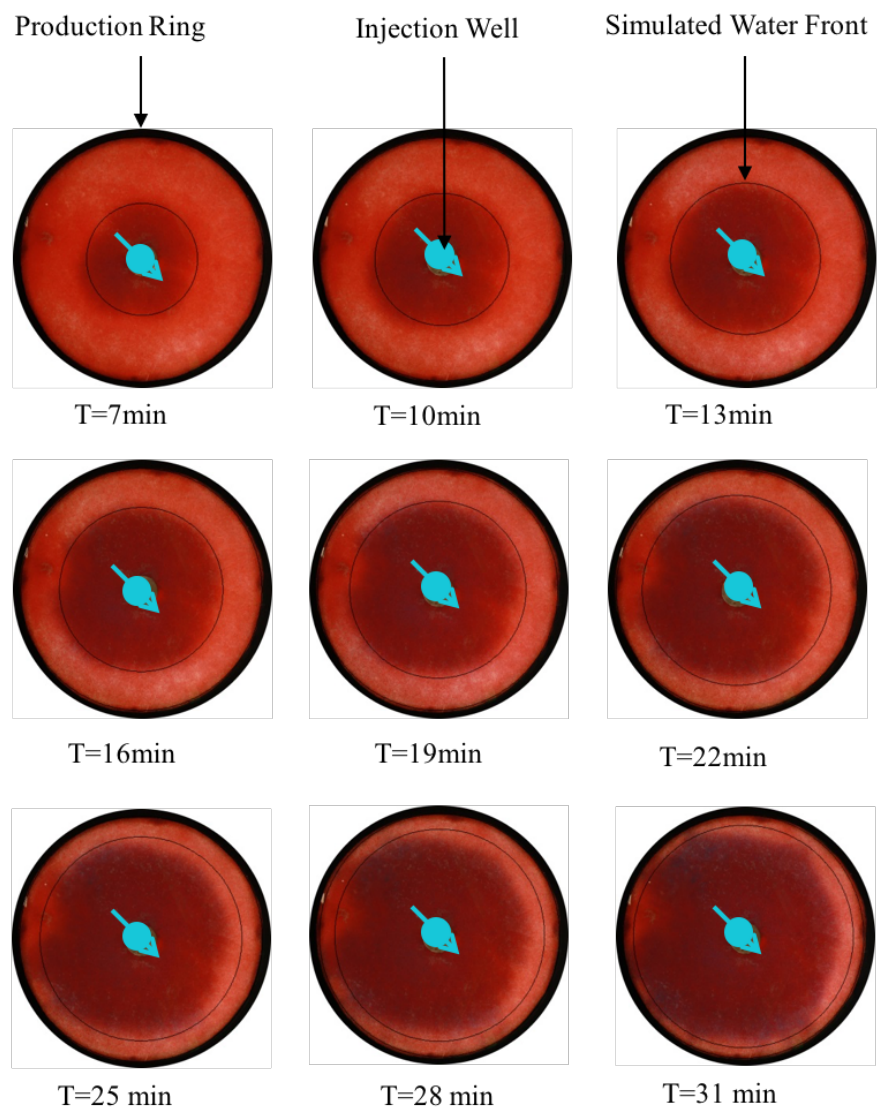


Figure 5.7: Displacement Fronts Comparison for the Homogeneous Experiment

5.5.3 Heterogeneous Reservoir Experiments Simulation using Homogeneous Corey Model

Two heterogeneous experiments with a low permeability sector, performed under the same differential pressure, are described in this section. The low permeable sectors

Table 5.6: Experimental Parameters used for the Heterogeneous Experiments

Parameters	**Heterogeneous Replicate 1**		**Heterogeneous Replicate 2**	
	SI Units	Lab Units	SI Units	Lab Units
Wellbore Radius	0.0079 m	0.3 in	0.0079 m	0.3 in
External Radius	0.1524 m	6.000 in	0.1524 m	6.000 in
Reservoir Thickness	0.0115 m	0.5 in	0.0112 m	0.5in
Porosity	0.426	0.426	0.417	0.417
Differential Pressure	17926 Pa	2.6 Psi	17926 Pa	2.6 Psi
Bulk Permeability	$1.58 \times 10^{-12}\ m^2$	1.58 $Darcy$	$1.58 \times 10^{-12}\ m^2$	1.58 $Darcy$
Block Permeability	$1.08 \times 10^{-12}\ m^2$	1.08 $Darcy$	$1.08 \times 10^{-12}\ m^2$	1.08 $Darcy$
Connate Water Saturation	0.360	0.360	0.364	0.364
Residual Oil Saturation	0.193	0.193	0.208	0.208
Oil Viscosity	$1.19 \times 10^{-3}\ P \cdot s$	1.19 cP	$1.19 \times 10^{-3}\ Pa \cdot s$	1.19 cP
Water Viscosity	$1 \times 10^{-3}\ Pa \cdot s$	1.00 cP	$1 \times 10^{-3}\ Pa \cdot s$	1.00 cP
Breakthrough Times	61 min	61 min	63 min	63 min

are cut from a consolidated glass-beads core. Permeability for this low permeable core is pre-tested in a core holder. We performed two experiments under the same differential pressure for replication. The parameters for these two experiments are shown in Table 5.6.

Since the heterogeneous sector takes up less than 10% of the total volume, the values of a_w, a_o, n_o, and n_w in the homogeneous experiment and the heterogeneous properties r_w, r_e, h, ϕ, K, S_{wc}, and S_{or} with the differential pressure Δp are used to simulate the flow rate and the breakthrough time for heterogeneous replicate 1 and heterogeneous replicate 2. Figure 5.8 shows the simulated flow rates by using the homogeneous Corey model and the experimental flow rates. As can be seen from this figure, simulated flow rates by using the homogeneous Corey model are well matched with the experimental flow rates. However, the breakthrough times are different between the simulated and experiment results.

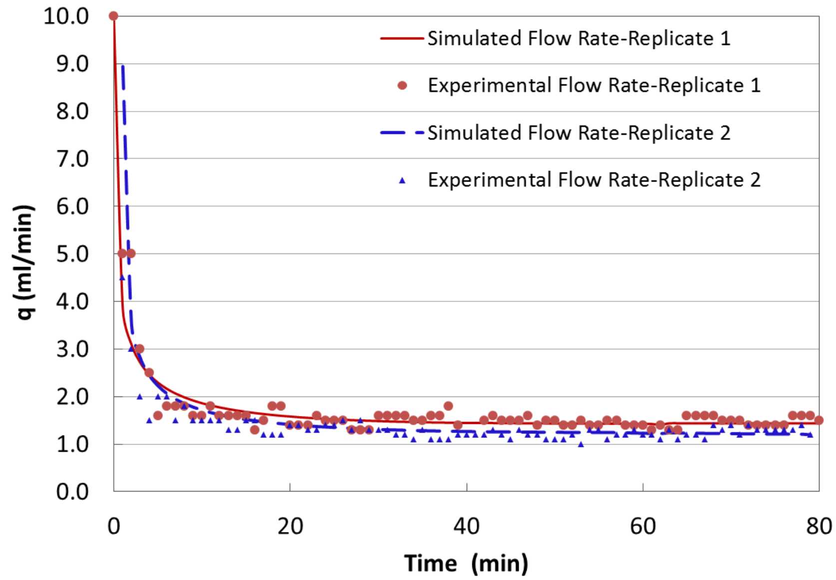


Figure 5.8: Flow Rate Comparison for the Heterogeneous Experiments using Homogeneous Corey Model

The simulated breakthrough times are 58 min and 57.5 min by using the homogeneous Corey model, respectively. The relative error between the simulated and the experimental breakthrough time are shown in Table 5.7.

Table 5.7: Relative Errors in Breakthrough Time for the Heterogeneous Experiments

Replicate	**Simulated Breakthrough Time** (min)	**Experimental Breakthrough Time** (min)	**Error** (%)
1	58.0	61.0	4.92
2	57.5	63.0	8.73

As can be seen from Table 5.7, the breakthrough time errors are acceptable for both heterogeneous experiments. However, they are different for these two heterogeneous experiments. We define the dimensionless time as:

$$t_{BTr} = \frac{t}{t_{BT}}, \tag{5.15}$$

where t is the real time and t_{BT} is the experimental breakthrough time.

Next, we use the dimensionless time to compare simulated cumulative productions, which obtained by using the homogeneous Corey model, with the experimental flow rate for the heterogeneous experiments. Table 5.8 and 5.9 show the detailed cumulative total production data for the heterogeneous experiments and the corresponding errors. Figure 5.9 shows the cumulative total productions for the heterogeneous experiments.

According to the cumulative production data, the errors are different for replicate experiments. Heterogeneous replicate 2 shows larger errors in both breakthrough time and cumulative production. This may be because we pack the porous media for each experiment and also the connate water saturations and the residual oil saturations are different in the heterogeneous experiments.

Table 5.8: Cumulative Production Data for Heterogeneous Replicate 1 using Homogeneous Corey Model

Dimensionless Time	**Simulated Cumulative Production** (ml)	**Experimental Cumulative Production** (ml)	**Error** (%)
0.02	9.9	6.0	64.97
0.10	23.9	27.0	11.36
0.18	33.6	35.0	3.98
0.26	43.8	43.0	2.13
0.34	49.7	51.0	2.47
0.44	58.5	60.0	2.42
0.54	67.0	69.0	2.86
0.62	73.9	76.0	2.72
0.69	79.0	83.0	4.78
0.77	86.3	90.0	4.14
0.87	94.2	100.0	5.78
1.00	104.9	107.0	1.94
1.03	107.8	110.0	20.2
1.08	112.1	116.0	3.39
1.15	116.4	122.0	4.60
1.21	122.2	128.0	4.52
1.31	131.1	136.0	3.62

Table 5.9: Cumulative Production Data for Heterogeneous Replicate 2 using Homogeneous Corey Model

Dimensionless Time	**Simulated Cumulative Production** (ml)	**Experimental Cumulative Production** (ml)	**Error** (%)
0.02	9.8	5.0	95.97
0.10	20.7	14.0	47.61
0.17	30.1	23.0	30.79
0.25	37.6	28.0	34.25
0.33	44.2	36.0	22.75
0.44	53.4	46.0	16.16
0.52	59.4	53.0	12.17
0.60	66.1	58.0	13.90
0.67	70.0	64.0	9.34
0.73	75.1	69.0	8.91
0.83	82.1	77.0	6.68
1.00	95.5	86.0	11.11
1.03	97.2	89.0	9.23
1.10	101.7	96.0	5.97
1.16	106.2	102.0	4.12
1.25	112.8	109.0	3.46

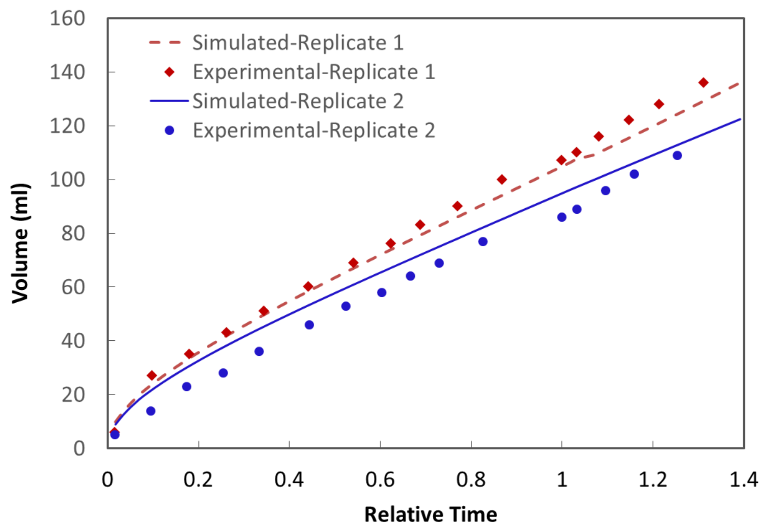


Figure 5.9: Cumulative Production Comparison for the Heterogeneous Experiments

We also use the average values (porosity, connate water saturation and residual oil saturation) of the two heterogeneous experiments in Table 5.10 and the homogeneous Corey model to simulate the average breakthrough time and the average flow rates. The simulated average breakthrough time is 60.5 *min*. Compared with the average experimental breakthrough time, which is 62.0 *min*, the error is only 2.42 %. Then, we plot the average cumulative production of the two heterogeneous experiments, which obtained by using the homogeneous Corey model, as a function of dimensionless time. Meanwhile, we plot the experimental the average cumulative production and standard deviations as a function of dimensionless time. Figure 5.10 shows the average cumulative total production for the heterogeneous experiments.

Table 5.10: Average Values for the Heterogeneous Experiments

Parameters	**Replicate 1**	**Replicate 2**	**Average**	**Standard Deviation**
Porosity	0.426	0.417	0.422	0.006
Connate Water Saturation	0.360	0.364	0.362	0.003
Residual Oil Saturation	0.193	0.208	0.201	0.011
Breakthrough Time (min)	61.0	63.0	62.0	1.4
Simulated Breakthrough Time (min)	58.0	57.5	57.8	0.4
Breakthrough Time Error (%)	4.92	8.73	6.77	–

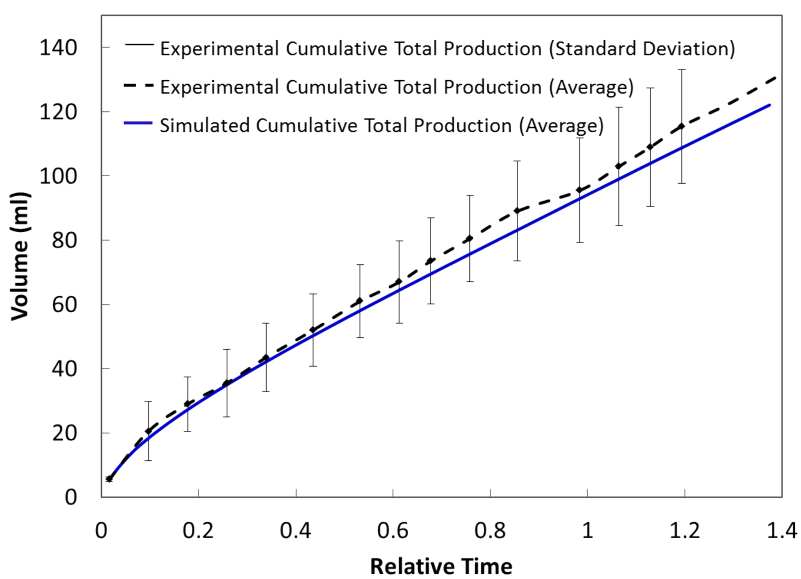


Figure 5.10: Cumulative Production Comparison for Heterogeneous Experiments

According to the cumulative production data, even through the errors are different for replicate experiments. The simulated and experimental average breakthrough time and average flow rates are well matched. Hence, the homogeneous Corey model can be applied in the heterogeneous experiments.

5.5.4 Heterogeneous Reservoir Experiments and History Matched Results

Breakthrough time is the most important parameter in the waterflooding experiment. The simulated breakthrough times for the heterogeneous experiments by using the homogeneous Corey model are different from the experimental breakthrough time, hence we use the trial and error method described in **5.5.1** to history match the relative permeabilities for the heterogeneous experiments to get a better match. We tuned the relative permeabilities for the two heterogeneous experiments separately to obtain a better agreement in breakthrough time. As illustrated in Figure 5.11, history matched flow rates match almost perfectly with the experimental flow rates for both experiments. The experimental dots are very close to the history matched line for both experiments.

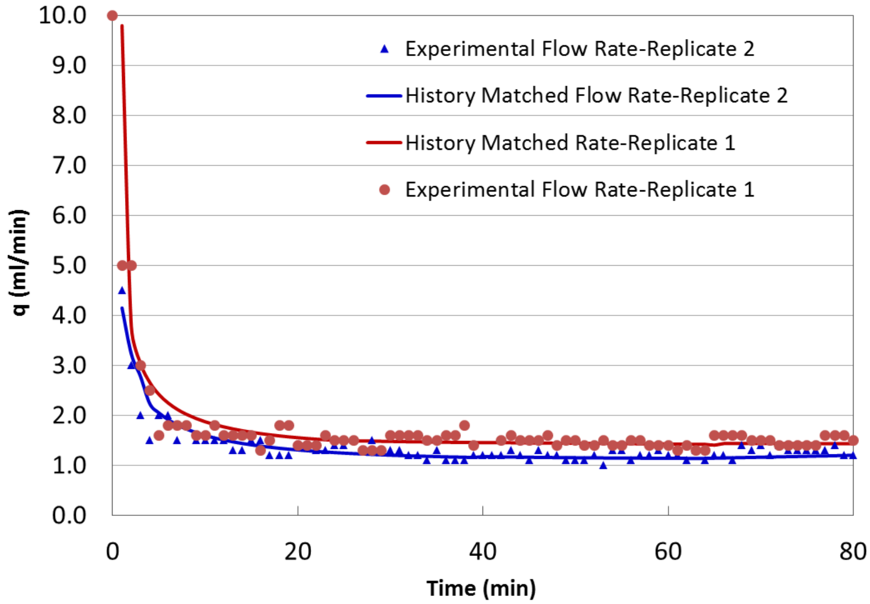


Figure 5.11: Flow Rate Comparison for the Heterogeneous Experiments

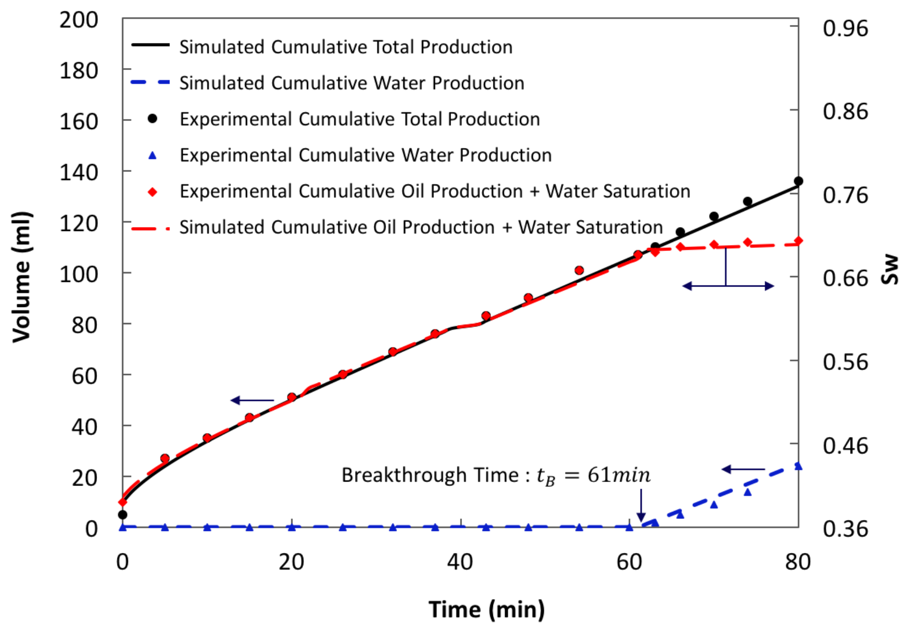
The cumulative production rates for the heterogeneous experiments are shown in Table 5.11 and 5.12 and plot in Figure 5.12. As for the homogeneous experiment, we also use the breakthrough time as the main parameter for the history match process in the heterogeneous experiments. The cumulative production rates are also matched with excellent agreement. At 80 min, the total cumulative flow rate difference between the history matched and experimental results are 1.44% and 2.25% for each replication, respectively.

Table 5.11: Cumulative Production Comparison for Heterogeneous Replicate 1

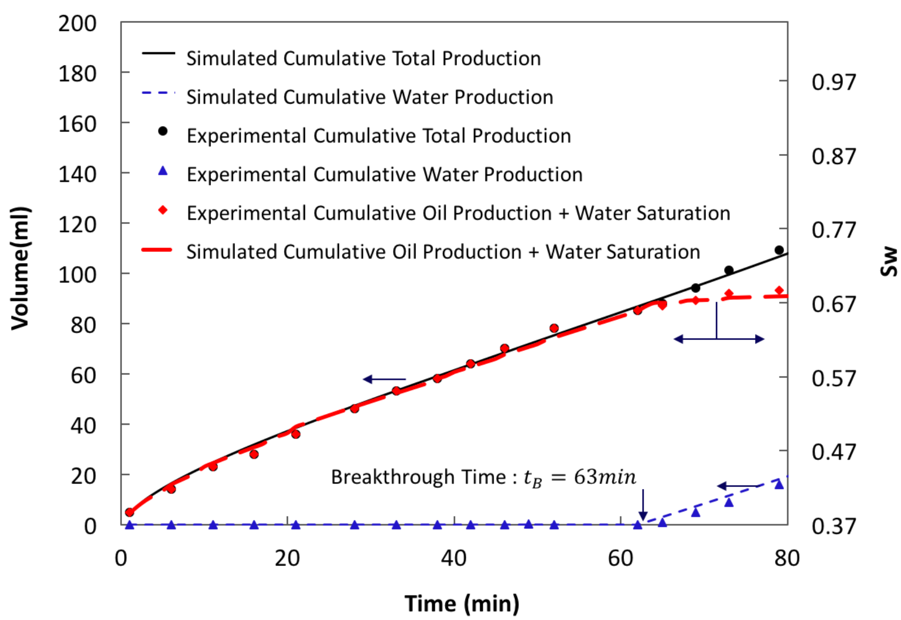
Time(min)	**History Matched Cumulative Production** (ml)			**Experimental Cumulative Production** (ml)			**Error** (%)		
	Total	Oil	Water	Total	Oil	Water	Total	Oil	Water
1	9.8	9.8	0	6.0	6.0	0	63.20	63.20	-
6	23.9	23.9	0	27.0	27.0	0	11.38	11.38	-
11	33.7	33.7	0	35.0	35.0	0	3.76	3.76	-
16	42.2	42.2	0	43.0	43.0	0	1.94	1.94	-
21	50.0	50.0	0	51.0	51.0	0	1.91	1.91	-
27	59.0	59.0	0	60.0	60.0	0	1.62	1.62	-
33	67.9	67.9	0	69.0	69.0	0	1.61	1.61	-
38	75.2	75.2	0	76.0	76.0	0	1.06	1.06	-
42	81.0	81.0	0	83.0	83.0	0	2.40	2.40	-
47	88.2	88.2	0	90.0	90.0	0	1.97	1.97	-
53	96.8	96.8	0	100.0	100.0	0	3.15	3.15	-
59	105.4	105.4	0	106.0	106.0	0	1.47	1.47	-
61	106.8	106.8	0	107.0	107.0	0	0.18	0.18	-
63	109.7	107.1	2.6	110.0	108.0	2.0	0.28	0.84	29.73
66	114.0	107.5	6.5	116.0	110.0	6.0	1.76	2.31	8.37
70	119.7	108.0	11.7	122.0	110.0	11.0	1.88	2.75	6.84
74	125.4	108.4	17.0	128.0	112.0	16.0	1.99	3.18	6.27
80	134.0	109.2	24.9	136.0	112.5	23.5	1.44	2.97	5.89

Table 5.12: Cumulative Production Comparison for Heterogeneous Replicate 2

Time(min)	**History Matched Cumulative Production** (ml)			**Experimental Cumulative Production** (ml)			**Error** (%)		
	Total	Oil	Water	Total	Oil	Water	Total	Oil	Water
1	5.0	5.0	0	5.0	5.0	0	2.93	2.93	-
6	16.9	16.9	0	14.0	14.0	0	21.70	21.70	-
11	25.1	25.1	0	23.0	23.0	0	9.13	9.13	-
16	32.2	32.2	0	28.0	28.0	0	13.83	13.83	-
21	38.6	38.6	0	36.0	36.0	0	7.37	7.37	-
28	47.2	47.2	0	46.0	46.0	0	0.03	0.03	-
33	53.2	53.2	0	53.0	53.0	0	0.40	0.40	-
38	59.0	59.0	0	58.0	58.0	0	1.72	1.72	-
42	63.7	63.7	0	64.0	64.0	0	0.41	0.41	-
46	68.4	68.4	0	69.0	69.0	0	0.89	0.89	-
52	75.3	75.3	0	77.0	77.0	0	2.20	2.20	-
63	87.8	87.8	0	86.0	86.0	0	2.05	2.05	-
65	90.0	87.0	3.1	89.0	87.0	2.0	1.20	0.09	56.3
69	94.8	87.4	7.3	96.0	90.0	6.0	1.34	2.98	23.12
73	99.4	87.7	11.7	102.0	91.0	11.0	2.53	3.60	6.26
79	106.5	88.3	18.2	109.0	93.0	19.0	2.25	5.03	4.11

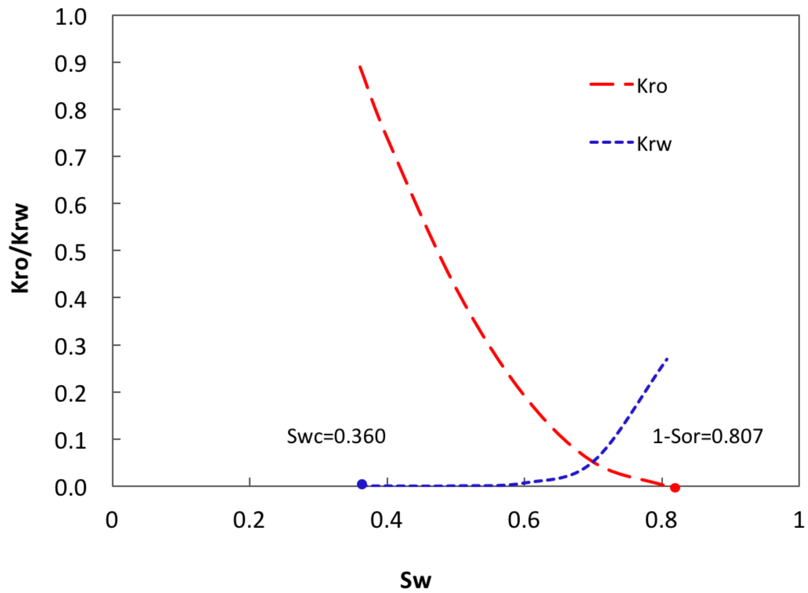


(a) Replicate 1

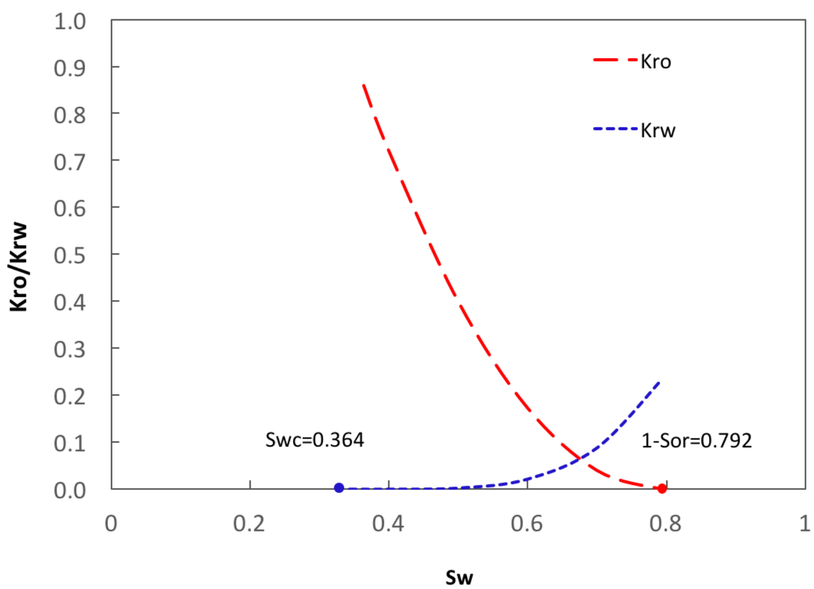


(b) Replicate 2

Figure 5.12: Accumulated Production Comparison for the Heterogeneous Experiments



(a) Replicate 1



(b) Replicate 2

Figure 5.13: History Matched Relative Permeabilities for the Heterogeneous Experiments

The relative permeability curves determined by history matching are shown in Figure 5.13. The matched Corey relative permeability models for the heterogeneous experiments are:

$$K_{rw1} = 0.235 \left(\frac{S_w - 0.360}{0.447} \right)^4, \tag{5.16}$$

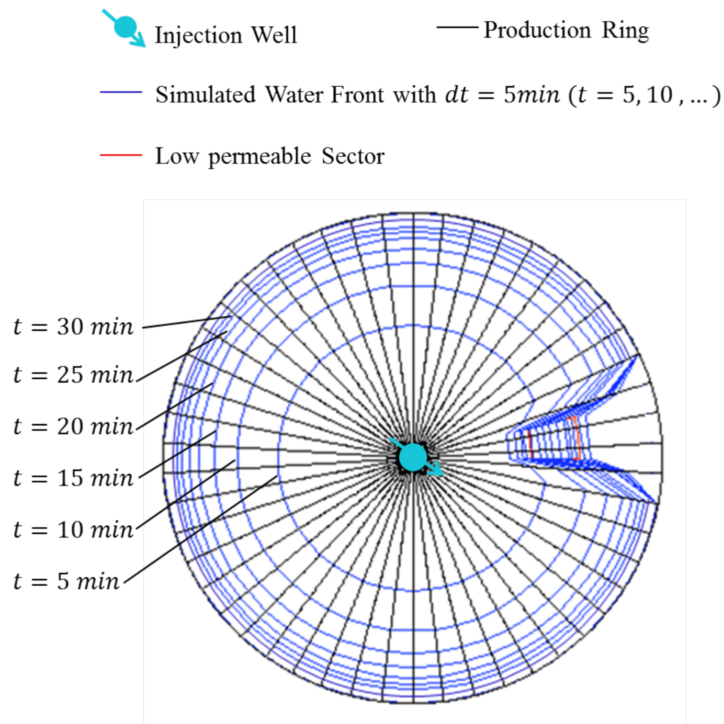
$$K_{ro1} = 0.891 \left(\frac{0.807 - S_w}{0.447} \right)^2, \tag{5.17}$$

$$K_{rw2} = 0.228 \left(\frac{S_w - 0.364}{0.428} \right)^4, \tag{5.18}$$

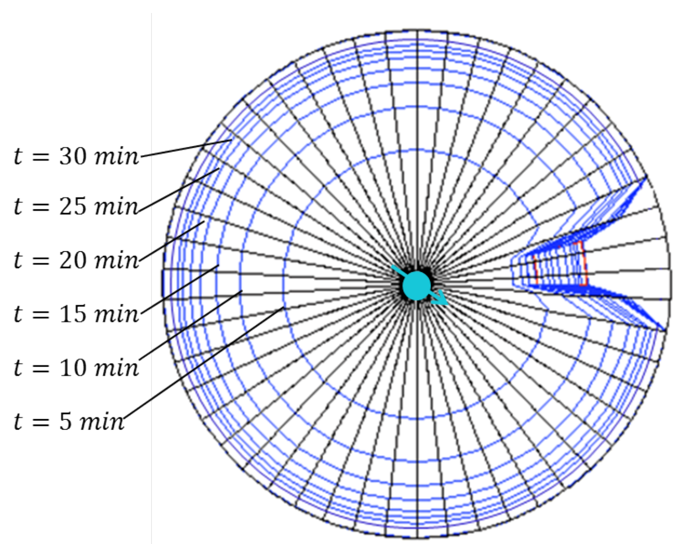
$$K_{ro2} = 0.867 \left(\frac{0.792 - S_w}{0.428} \right)^2. \tag{5.19}$$

The overall Corey model for the heterogeneous experiments is shown in **Appendix D**.

Figure 5.14 shows the simulated streamlines and displacement fronts every five minutes. The black lines are the streamlines, and the blue lines are the displacement fronts. Since the pressure differences for replicate 1 and replicate 2 are the same, streamlines are identical for both experiments. We observe that some of the streamlines, very close to the low permeable boundaries, try to avoid flowing across the low permeable region. This trend is not very obvious since the difference between the block and the bulk permeability is small. The simulated displacement fronts show delayed movements in the low permeable sector. For both experiments, fronts move slowly when getting closer to the boundary. More specifically, it is hard to differentiate the waterfronts in the bulk area for later times.



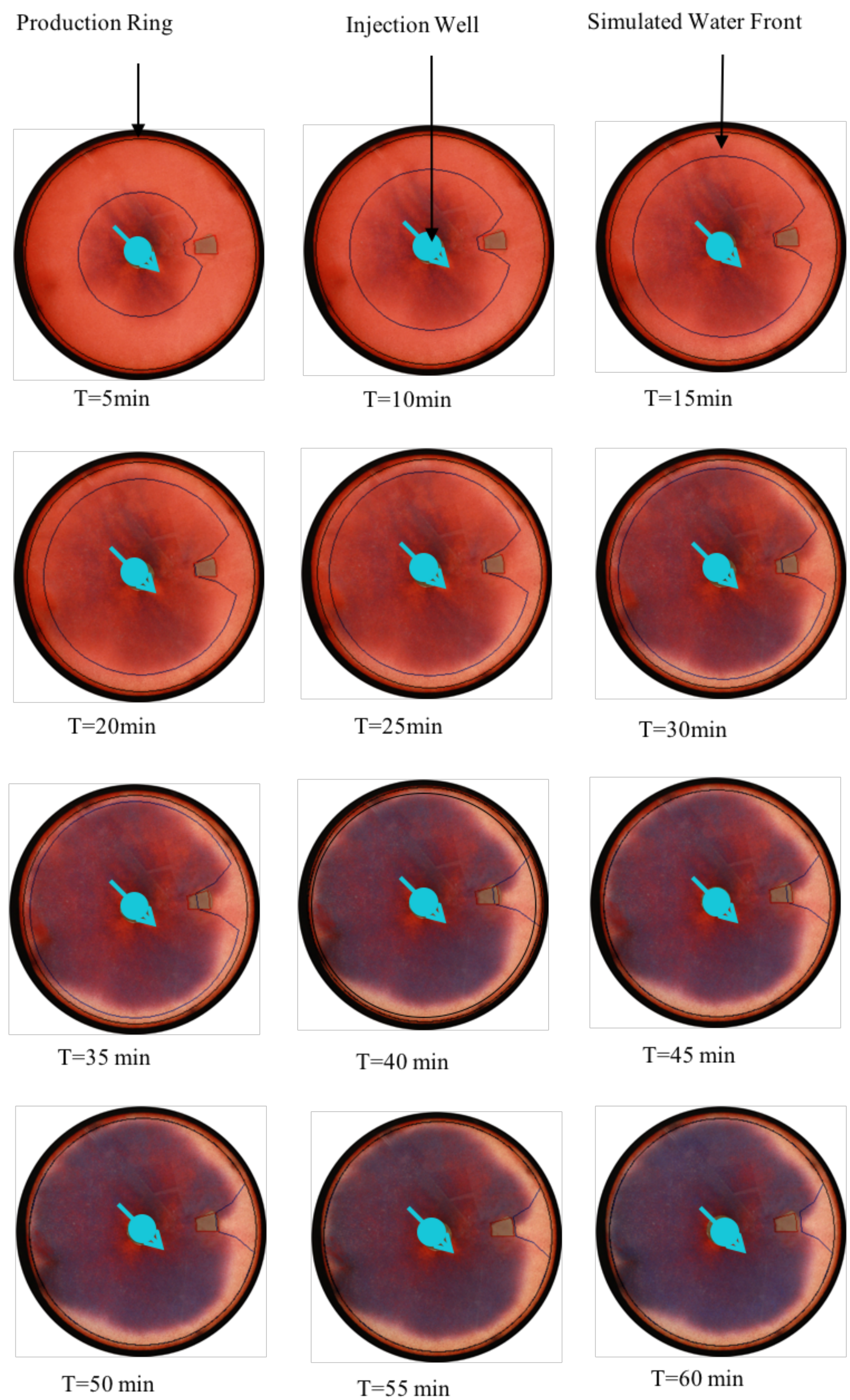
(a) Replicate 1



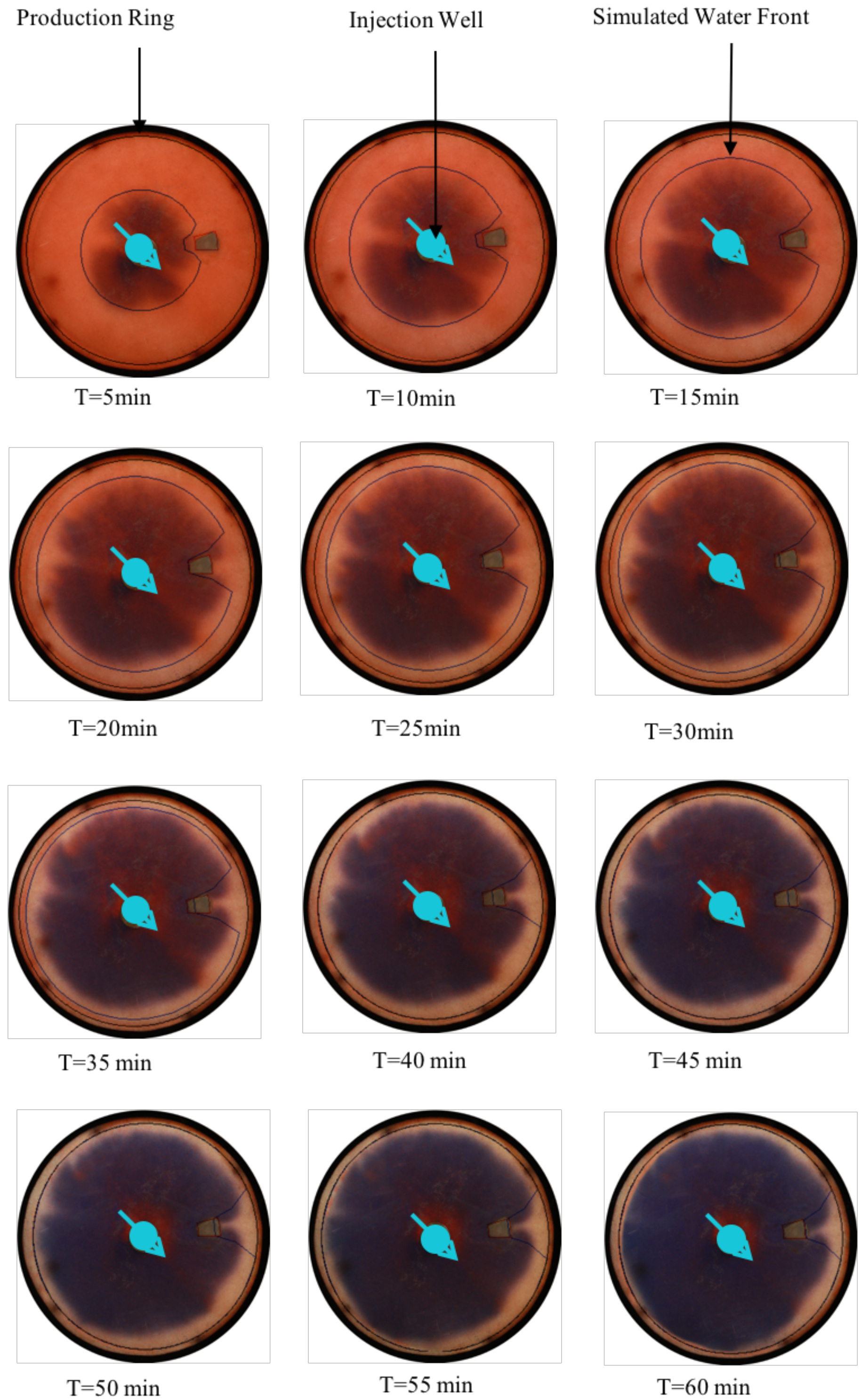
(b) Replicate 2

Figure 5.14: Simulated Displacement Fronts for the Heterogeneous Experiments

History matched and experimental displacement fronts are shown in Figure 5.15. In the experiments, oil and water are dyed to red and blue, respectively. Blue water is injected at the center, and fluid is produced at the outer ring. As can be seen from Figure 5.15, the blue water moves toward the outer boundary. When the water front reaches the heterogeneous sector, the water front moves slower in the heterogeneous sector than in the rest of the area. In the simulation, the black lines represent the production ring, and the blue lines are the simulated waterfronts at different times. The red lines indicate the low permeable sector. As shown in the figures, the simulated waterfronts describe the experimental fronts accurately. However, at later times, experimental fronts are somewhat different from the simulated front. The same behavior is observed in the homogeneous experiment. This is because the production ring created in the experiment is a perforated tube. In the simulations, the production ring is treated fully open. This causes the frontal differences at late times.



(a) Replicate 1



(b) Replicate 2

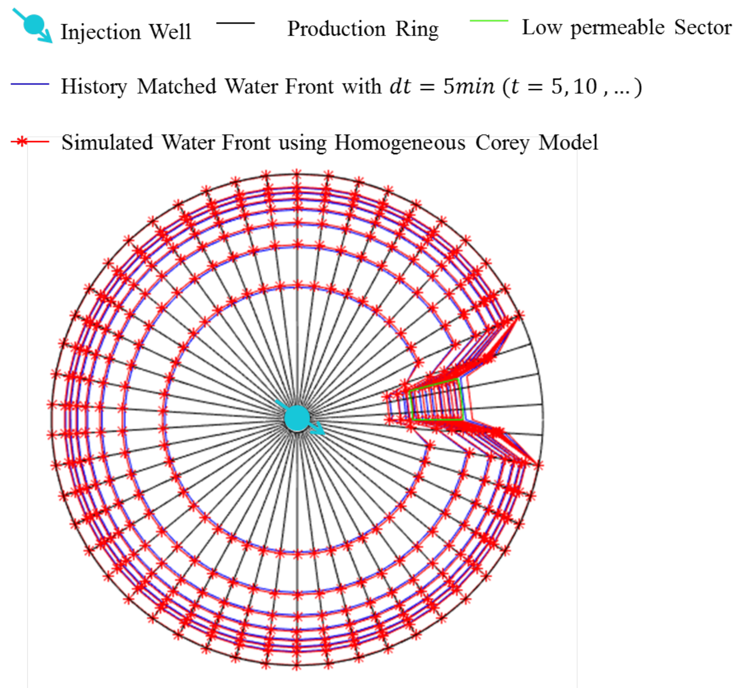
Figure 5.15: Displacement Fronts Comparison for the Heterogeneous Experiments

5.5.5 Simulated and History Matched Waterfront Comparison

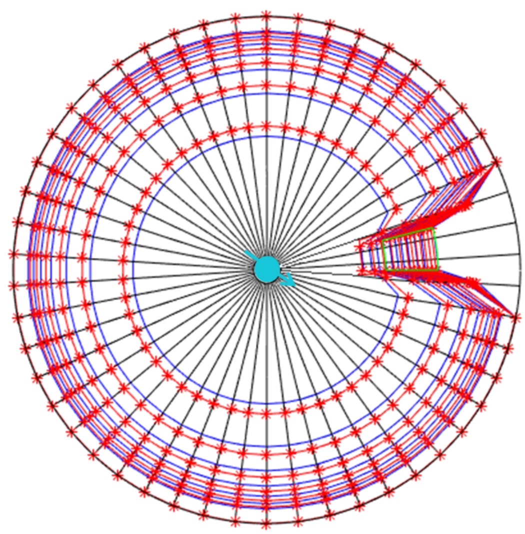
The simulated waterfronts for heterogeneous experiments by using the homogeneous Corey model and the history matched waterfronts are shown in Figure 5.16. As can be observed, the simulated waterfronts move faster than the history matched waterfronts. This is in accordance with the breakthrough time (Table 5.13) since the simulated breakthrough times are earlier than the history matched breakthrough time. The simulated and history matched waterfronts for replicate 1 are very close. Because the breakthrough time has a larger difference for replicate 2, the simulated and history matched waterfronts have a larger difference for replicate 2. However, the error between the average simulated and history matched breakthrough time is only 2.89%.

Table 5.13: Simulated and History Matched Breakthrough Times for the Heterogeneous Experiments

Replicate	**Simulated Breakthrough Time** (min)	**History Matched Breakthrough Time** (min)	**Error** (%)
1	58.0	61.3	5.38
2	57.5	63.2	9.02
Average	60.5	62.3	2.89



(a) Replicate 1



(b) Replicate 2

Figure 5.16: Simulated and History Matched Displacement Fronts for Heterogeneous Experiments

5.6 Experiments Conclusion

The absolute permeability test was performed for the unconsolidated glass beads using the radial cell and was found to be $1.58 \times 10^{-12}\ m^2$. The permeability of the small heterogeneous sector was $1.08 \times 10^{-12}\ m^2$.

We first history matched the relative permeabilities for the homogeneous experiment. The homogeneous Corey model parameters were then applied to simulate the heterogeneous experiments. The breakthrough times and flow rates obtained from the simulator were compared with the experimental results. Breakthrough time error between the simulated result and the experimental results were 4.92% and 8.73% for replicate 1 and replicate 2, respectively. The flow rate errors for replicate 1 were acceptable. However, the flow rate errors for replicate 2 were relatively large.

The breakthrough time errors and the flow rate errors were different for these two heterogeneous experiments, which indicate the porous media changed in each test. Hence, we used the dimensionless time to compare the average breakthrough time and average flow rate for the two heterogeneous experiments. The average breakthrough time error was only 2.42%. The simulated average flow rates well matched with the experimental results.

We also used the history match method to determine the relative permeabilities for the heterogeneous experiments separately. The history matched breakthrough times were the same as the experimental breakthrough times for both heterogeneous experiments. According to breakthrough time and the flow rate errors, we can conclude that by tuning relative permeabilities for each experiment separately, excellent results were obtained. The Corey model parameters, obtained by the history match method, were different from the homogeneous Corey model parameters. They are listed in Table

5.14. According to the history matched results, this approach provides excellent agreements. Hence, the history match approach may further be applied to relative permeability tests by using the Corey model.

Table 5.14: Corey Model Parameters

Parameters	**Replicate 1**	**Replicate 2**	**Homogeneous**
a_w	0.235	0.228	0.258
a_o	0.891	0.867	0.855
n_w	4	4	4
n_o	2	2	2

This is the first time a radial water flooding experiment is demonstrated to match an analytical solution for such flow. It is believed that these experiments confirm the accuracy of the semi-analytical streamline simulation method.

In the 2D water flooding experiments, the porous media is water wet. In a water wet system, capillary pressure is a positive force in the water drainage process. When history matching the experiment process, we ignored the capillary pressure. However, capillary pressure will increase the recovery factory in the experiment.

Chapter 6

Summary

6.1 Conclusions

The streamline simulation method is an effective and complementary technology in reservoir simulation. It demonstrates the effectiveness in solving full field problems. The near-wellbore streamlines simulation researches are shown in very little literature. The semi-analytical streamline simulation method presented in this research thesis is performed in the near-wellbore region for a single well and shows the advantage in simulating the near-wellbore region, especially when heterogeneities exist and in perforated wells. The model presented in this research thesis is derived for vertical wells. It can easily be applied to horizontal wells just by changing the directional permeability calculation method.

The semi-analytical streamline simulation method applied in this research thesis is superior in the pressure distribution compared to the industry standard streamline simulation method (Pollock's method). For the incompressible system without considering the gravity and diffusive effects, an elliptic pressure equation is obtained. The finite difference method is then applied to obtain the node pressures. We then impose flux continuity, pressure continuity across each grid block boundary and local mass conservation to calculate the corner pressures from the finite difference pressure nodes. In this research thesis, streamlines are defined by a closed formula derived from the 2D log-lin pressure approximation function and 3D bilin-log pressure approximation function. The pressure analysis proves that the pressure assumption used in the present semi-analytical streamline simulation method satisfies Laplace equation at each point inside each simulation grid block. It also shows the advantage of satisfying the principle of pressure continuity across the block boundaries. Hence, the pressure assumptions ensure the accuracy of the present semi-analytical streamline simulation method.

The present semi-analytical streamline simulation method shows its advantage in modeling single-phase flow in 2D in both open hole wells and perforated wells. Streamline trajectories for the present semi-analytical method, Pollock's method, and the fully analytical method are identical; however, the TOF result shows that the present semi-analytical method is identical to the fully analytical solution, with errors caused by digital truncation only. Pollock's method, on the other hand, exhibits unacceptable errors in TOF especially with low grid resolutions. Streamlines capture flow paths which accurately represent the distribution of permeability, as streamlines are denser in the high permeability area. The 2D heterogeneity case with a low permeability sector illustrates that streamlines are strongly influenced by the reservoir permeability. Streamlines generated from Pollock's method do not flow across the low permeable area and the nearby grid blocks which produce systematic errors. On the contrary, only some of the semi-analytical streamlines, very close to the low permeable boundaries, avoid flowing across the low permeable region which is physically more reasonable. Streamlines generated by the present semi-analytical method in the two-dimensional heterogeneity case with a high permeability contrast sector show more reasonable results compared to the standard method. The TOF results show that fluid flows faster in the high permeable area. Hence, to provide maximum productivity, the lower permeability area should be avoided by selective perforation or inflow control devices. The present streamline simulator obviously shows the advantage in modeling the streamlines for the perforated wells. Pollock's method fails to trace the streamlines in the 2D perforation case. Pollock's method in Polar coordinates was derived in the context of Cartesian coordinates. Although it works very well in Cartesian coordinates, it is not necessarily applicable in the Polar/Cylindrical coordinates. In contrast, the present streamline simulator gives physically reasonable streamline paths.

Most significantly, it is revealed through this research thesis that the semi-analytical streamline simulation method developed is the only known streamline method with sufficient accuracy and efficiency for streamline simulation in polar/cylindrical geometries.

The new skin calculation method for the perforated wells introduced in this research thesis is superior to the classical skin calculation method (Karakas-Tariq method). This new skin calculation method is based on the continuous pressure assumption in the present semi-analytical streamline simulation method. The skin calculation method established in this research thesis represents an accurate determination of the flow rate for both two-dimensional and three-dimensional cases. The 3D perforated case demonstrates that the present semi-analytical streamlines can handle the 3D complex geometry and provides an accurate result. This is a novel and non-trivial extension of streamline simulation. In other words, regardless of complex geometry in the near-wellbore region, it provides an accurate solution. Unlike the Karakas-Tariq method which exhibits the unphysical behavior with increasing perforation length near the damaged zone, the new skin calculation method creates physically reasonable results. Furthermore, it captures the effect of flow convergence. Results clearly indicate that as the perforation length increases, skin value decreases, as opposed to the standard method. The new skin calculation method introduced in this research thesis can also easily be used to study the effect of other perforation parameters.

Two-dimensional waterflooding visualization experiments are performed in radial glass-beads macro-models to represent the near-wellbore region. They are used to observe visually what happens in the homogeneous and heterogeneous near-wellbore region during water flooding at constant differential pressure. This is the first time water flooding experiment is performed in the radial geometry in macro-models. The 2D waterflooding experiments are operated under constant differential pressure conditions. Experimental results visually help us to understand what happens in the heterogeneous near-wellbore region during water flooding. The stream tube simulation method is used to history match the laboratory scale displacements successfully.

Because some unphysical behavior shows up in streamlines generated from Pollock's method in the single-phase flow, we abandoned Pollock's method in the stream tube simulation for the two-phase flow in this research thesis. In two phase flow simulations, we coupled the 3D Riemann solution and the present semi-analytical stream tube simulation to describe two-phase flow in the homogeneous and heterogeneous porous media under the constant differential pressure condition. The cross section area is changing along the stream tube in the near-wellbore region. Utilizing the 3D Riemann solution along each stream tube transformed the 3D problem into a set of 1D problems. Each stream tube is treated as a 1D system along which solutions of mass conservation equations are solved. In the constant pressure boundary case, the flow rate becomes a function of time as the flood progresses. The present semi-analytical streamline method using constant pressure boundary conditions is demonstrated for the first time. As shown in the history matching results, the streamline simulation can be used to describe the fluid movement in the experiments accurately. This demonstrates the history matching ability of the present streamline simulation. The excellent match between the waterflooding experimental and the simulated results also provides the evidence of the accuracy of the present semi-analytical streamline simulator for heterogeneous reservoirs.

According to the TOF result in the homogeneous reservoir, the perfect matching for the experimental results and the results from the perforated wells, we can conclude that the present semi-analytical method is superior to Pollock's method in homogeneous reservoirs, heterogeneous reservoirs and perforated wells. The present semi-analytical method also provides more reasonable total skin results compared to the standard method.

6.2 Significance of Research

The typical streamline simulation model is designed for full field planning, and uses simplistic models for near-well flow calculations. The streamline simulation presented in this research thesis focuses on the near-wellbore region. It is significant in the petroleum industry.

The semi-analytical streamline simulation method is a powerful tool to visualize how heterogeneities affect flow distribution in the near-wellbore region. Meanwhile, it can model the nature of the components contributing to skin, which would help the engineers make the final choice of well completion. The potential application for this semi-analytical streamline simulation method in the industry can be calculation of exact productivity and skin factor for individual wells; well completion optimization (perforation optimization); determination of waterfront in the near well region. The semi-analytical streamline simulation method can be used to determine the local flow characteristics, which are subsequently incorporated in the reservoir simulator. It will bridge the gap between completion and reservoir technology.

6.3 Limitations

The stream tube approach discussed in this research thesis has its limitations. The 3D Riemann solution applied in this research thesis requires an explicit expression for stream tube cross section area. In a radial geometry for a homogeneous case, stream tube areas as a function of the arc length for each stream tube can be explicitly expressed. For a heterogeneous case, no explicit expression can be obtained. Hence the geometry of the stream tube is used to calculate the cross section area numerically.

In the streamline simulation method, certain assumptions are applied. In an oil reservoir, oil and water are assumed incompressible. If we want to apply this semi-analytical streamline simulation method in a gas well, compressibility should be taken into consideration. It is reasonable to ignore gravity in vertical wells since the near-wellbore region radius is not long compared to the vertical thickness of the reservoir. For a horizontal well, we can invoke the same method presented in Bratvedt et al. (1996).

The stream tube approach cannot represent any cross flow between stream tubes caused by diffusion such as capillary pressure in two-phase flow and mechanical dispersion in miscible flow. To incorporate gravity and dispersion, the present semi-analytical streamline method, operator splitting techniques can be invoked.

6.4 Recommendations

Throughout this research thesis streamline simulation has only been performed with anisotropy in the $z-$ direction. In other words, we can handle only the case with permeability in $z-$ direction being different from the horizontal (radial and angular) direction ($K_z \neq K, K = K_r = K_t$). However, it can be extended to anisotropy also in the radial direction and the angular direction ($K_r \neq K_t$). The present semi-analytical streamline simulation method can be applied to the anisotropic reservoir as long as the pressure distribution is obtained. The pressure solution method for a fully anisotropic case in horizontal direction is more complicated than the isotropic case. For an anisotropic reservoir, two approaches can be applied to obtain the pressure distribution. The first approach is to apply the full Laplace equation. This will include K_θ described in Equation 3.28 into the numerical discretion equation. Once the pressure is determined, the streamline expression can be obtained as described in this research thesis. The second approach is to use an anisotropic transform that maps an anisotropic medium onto an isotropic one. The detailed procedure to apply this transform is described in Johansen et al. (2016). The application of this transform is recommended in the future research. It will provide a more flexible application of the work in this research thesis.

This research thesis is based on quadrilateral grid blocks. Depending on the grid block shape, errors may occur in the representation of the geometry of a perforated well. Unstructured grids may better satisfy the complex geometry in a perforated well. The streamline simulation method introduced in this research thesis can also be applied for the triangular grid blocks. As the geometries get more complex, the triangular grid blocks may satisfy multiple constrains that are difficult for structured grids (Gupta and King, 2007). For triangular grid blocks, the pressure can be calculated by the finite element method. The log-lin pressure assumption is applied in each triangle. Hence, with three pressure nodes, three pressure equations can be obtained. One flux continuous equation can be introduced for each grid to solve the continuous pressure distribution.

Bibliography

[1] Aziz, K. and Settari, A., 1979. Petroleum Reservoir Simulation. *Applied Science Publishers LTD*, London.

[2] Batycky, R. P., 1997. A Three-Dimensional Two-Phase Field Scale Streamline Simulator. *Doctoral dissertation*, Stanford University, Dept. of Petroleum Engineering, Stanford, CA.

[3] Bratvedt, F., Bratvedt, K., Buchholz, C.F., Holden, L., Holden, H., and Risebro, N. H., 1992. A New Front-Tracking Method for Reservoir Simulation. *SPE Reservoir Engineering*,107-116.

[4] Bratvedt, F., Gimse, T., and Tegnander, C., 1996. Streamline Computations for Porous Media Flow including Gravity. *Transport in Porous Media*, 63-78.

[5] British Geological Survey, 2013. Modelling flow converging to a pumped borehole. http://www.bgs.ac.uk/research/environmentalModelling/Modellingflowtoboreholes.html

[6] Brock, D. C., and Orr Jr. F. M., 1991. Flow Visualization of Viscous Fingering in Heterogeneous Porous Media. *SPE*.

[7] Buckley, S. E., and Leverett, M. C., 1942. Mechanism of Fluid Displacement in Sands. *Transactions of the AIME*, 146, 01,107-116.

[8] Butler, R. M., and Mokrys, I. J., 1993. Recovery of heavy oils using vaporized hydrocarbon solvents: further development of the VAPEX process, *JCPT*, 32(6), 56-62.

[9] Chatenever, A. and Calhoun, J. C., 1952. Visual Examinations of Fluid Behaviour in Porous Media-Part I. *Society of Petroleum Engineers*, 4, 06.

[10] Chatzis, I., Morrow, N., and Lim, H. T., 1983. Magnitude and detailed structure of residual oil saturation. *Society of Petroleum Engineers Journal*, 23, 311-325.

[11] Chatzis, I., 2002. Pore scale phenomena of heavy oil recovery using vapor process. *International Symposium of Society of Core Analysts*.

[12] Chaudhari, L., Hadia, N. J., Mitra, S. K. and Vinjamur, M., 2011. Flow Visualization of Two-Phase Flow through Layered Porous Media. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, 33:10, 948-958.

[13] Chen, J. D., 1987. Radial Viscous Fingering Patterns in Hele Shaw Cell. *Experiments in Fluids*, 5, 363-371.

[14] Chuoke, R. L., P. van Meurs, and C. van der Poel., 1959. The instability of slow, immiscible, viscous liquid-liquid displacements in permeable media. *Petroleum Transactions*, AIME, Volume 216, 1959, pages 188-194.

[15] Corey, A. T., 1954. The Interrelation between Gas and Oil Relative Permeabilities. *Producers Monthly*, 19(1), 38-41.

[16] Craft, B.C. and Hawkins, M., 1991. *Applied Petroleum Reservoir Engineering*, ISBN 0130398845.

[17] Datta-Gupta, A. and King, M. J. A., 1995. Semianalytical Approach to Tracer Flow Modelling in Heterogeneous Permeable Media. *Advances in Water Resources*, 9-14.

[18] Datta-Gupta, A., and King, M. J., 2007. Streamline Simulation: Theory and Practice.*Society of Petroleum Engineers*, Vol. 11.

[19] Dawe, R. A., Wheat, M. R., and Bidner, M. S., 1992. Experimental Investigation of Capillary Pressure Effects on Immiscible Displacement in Lensed and Layered Porous Media. *Transport in porous media*, 7(1), 83-101.

[20] Emanuel, A. S., Alameda, G. K., Behrens, R. A., and Hewett, T. A., 1989. Reservoir Performance Prediction Methods Based on Fractal Geostatics. *SPE Reservoir Engineering*, 311-318.

[21] Fay, C. H. and Pratts, M., 1951. The Application of Numerical Methods to Cycling and Flooding Problems. *3rd World Petroleum Congress.*

[22] Glimm, J., Lindquist, B., McBryan, O. A., Plohr, B., and Yaniv, S., 1983. Front Tracking for Petroleum Simulation. *SPE Reservoir Simulation Symposium*, San Francisco, CA.

[23] Glimm, J., Grove, W. J, Li. X. L., and Zhao, N., 1999. Simple Front Tracking. *American Mathematical Society*, Volume 238.

[24] Hadibeik, H, Pour, L. A., Torres-Verdin, C, Sephernoori K, and Shabro V., 2011. 3D Multiphase Streamline-Based Method for Interpretation of Formation Tester Measurements Acquired in Vertical and Deviate Wells. *Society of Petroleum Engineers*. Denver, Colorado, USA.

[25] Higgins, R. V. and Leighton, A. J., 1962. A Computer Method to Calculate Two-Phase Flow in Any Irregularly Bounded Porous Medium. *Journal of Petroleum Technology*,14 (679-683).

[26] Higgins, R. V. and Leighton, A. J., 1962. Computer Prediction of Water Drive of Oil and Gas Mixtures Through Irregularly Bounded Porous Media for Three-Phase Flow. *Journal of Petroleum Technology*,14 (1048-1054).

[27] Hong, K. C., 1975. Productivity of Perforated Completions in Formations With or Without Damage. *JPT*, 1027-38.

[28] Huang, H., Zhang, F., Callahan, P., and Ayoub, J., 2012. Fluid Injection Experiments in 2D Porous Media. *SPE Joural*.

[29] James, L. A., 2003. A Closer Look at VAPEX. *Master Thesis*, University of Waterloo, Ontario, Canada.

[30] James, L. A. and Chatzis, I., 2004. Details of gravity drainage of heavy oil during vapor extraction. *In International Symposium of the Society of Core Analysts*.

[31] James, L. A., Rezaei, N., and Chatzis, I., 2008. VAPEX, Warm VAPEX and Hybrid VAPEX-The State of Enhanced Oil Recovery for In Situ Heavy Oils in Canada. *Journal of Canadian Petroleum Technology*, Waterloo, Ontario, Canada.

[32] James, L. A., 2009. Mass Transfer Mechanisms during the Solvent Recovery of Heavy Oil. *Doctoral dissertation*, University of Waterloo, Ontario, Canada.

[33] James, L. A., 2012. Experimental personal communication.

[34] Johansen, T. E., 2010. A New Semi-Analytical Method for Streamline Simulation. http://www.petreng-thormod.ca

[35] Johansen, T. E. and James, L. A., 2015. Solution of multi-component, two-phase Riemann problems with constant pressure boundaries, *Journal of Engineering Mathematics*, Volume 96, Issue 1, pp 23-35, Springer Verlag.

[36] Johansen, T. E., James, L. A., and Liu, X. 2016. On the Buckley-Leverett Equation With Constant-Pressure Boundary Conditions, *Society of Petroleum Engineering Journal*.

[37] Johansen, T. E. and Liu, X., 2016. Solution of Riemann Problem for Hyperbolic Systems of Conservation Laws Modeling Two Phase Flow in General Stream Tube Geometries, *Journal of Engineering Mathematics*, Springer Verlag.

[38] Johansen T. E., Hender D. G., James L. A., 2016. Productivity Index for Arbitrary Well Trajectories in Laterally Isotropic, Spatially Anisotropic Porous Media, in print, *SPE Journal*.

[39] Juanes, R., and Matringe, S. F., 2009. Unified formulation for high-order streamline tracing on two-dimensional unstructured grids, *Journal of Scientific Computing*, 38(1), 50-73.

[40] Karakas, M., and Tariq, S.M., 1991. Semianalytical Productivity Models for Perforated Completions, *SPE*.

[41] Klotz. J. A., Krueger, R. F., and Pye, D. S., 1974. Effect of Perforation Damage on Well Productivity, *JPT* , 1033-44.

[42] Lagrange, J. L., 1781. Mémoire sur la théorie du mouvement des fluids. Oeuvres de Lagrange, Tome IV, pp. 695-748.

[43] Lake L. W., Johnston J. R., Stegemeier G. L., 1981. Simulation and Performance Prediction of A Large-Scale Surfactant/Polymer Project. *Society of Petroleum Engineers Journal*, 21(731-739).

[44] LeBlanc, J. L. and Caudle, B. H., 1971. A Streamline Model for Secondary Recovery. *Society of Petroleum Engineers Journal*, 1(7-12).

[45] Mathews, J. L., Emanuel, A. S., and Edwards, K. A.,1989. Fractal Methods Improve Mitsue Miscible Predictions. *Journal of Petroleum Technology*, 1136-1149.

[46] Matringe, S. F., and Gerritsen, M. G., 2004. On Accurate Tracing of Streamlines. *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.

[47] Martin, J. C., Woo, P. T., and Wagner, R. E., 1973. Failure of Stream Tube Methods to Predict Waterflood Performance of An Isolated Inverted Five-Spot At Favorable Mobility Ratios. *Journal of Petroleum Technology*, 25(02, 151-153).

[48] Martin, J. C., and Wegner, R. E., 1979. Numerical solution of multiphase, two-dimensional incompressible flow using streamtube relationships. *Society of Petroleum Engineers Journal*, 19(5), 313-323.

[49] Montgomery, D.C., and Runger, G.C., 2006. *Applied statistics and probability for engineers, student solutions manual*, John Wiley and Sons, Inc.

[50] Morel-Seytoux, H.,1965. Analytical-Numerical Method in WaterFooding Predictions. *Society of Petroleum Engineers Journal*, 247-258.

[51] Muskat, M. and Wyckoff, R.,1934. A Theoretical Analysis of Waterflooding Networks, *Trans. AIME*, 107, 62-77.

[52] Muskat, M., 1937. The Flow of Homogeneous Fluids Through Porous Media. *International Human Resources Development Corporation*, ISSN: 0038-075X.

[53] Nilsen, H., and Lie, K., 2009. Front-Tracking Methods for Use in Streamline Simulation of Compressible Flow. *SPE Reservoir Simulation Symposium*, Woodlands: Society of Petroleum Engineers.

[54] Pasari, U., and Arihara, N., 2005. Application of Streamline Method to Hot Water-Flooding Simulation for Heavy Oil Recovery. *Paper SPE 93149 presented at the SPE Asia Pacific Oil and Gas Conference and Exhibition*, Jakarta, Indonesia.

[55] Paterson, L., 1981. Radial Fingering in a Hele Shaw Cell. *J. Fluid Mech*, 113, 513-529.

[56] Peddibhotla, S., Datta-Gupta, A., and Xue, G., 1997. Multiphase Streamline Modeling in Three Dimensions: Further Generalizations and a Field Application. *SPE Reservoir Simulation Symposium*, Dallas, TX: Society of Petroleum Engineers.

[57] Pitts, G. N. and Crawford, P. B., 1970. Low Areal Sweep Efficiencies in Flooding Heterogeneous Rock. *9th Biennial Symposium, Production Techniques*, Witchita Falls, TX: Society of Petroleum Engineers.

[58] Pollock, D. W., 1988. Semianalytical Computation of Path Lines for Finite-Difference Models. *Ground Water*, 26(6, 743-750).

[59] Pucknell, J. K. and Behrmann, L. A., 1991. An Investigation of the Damaged Zone Created by Perforation. *Paper SPE*, 22811, presented at the 66th SPE Annual Technical Conference and Exhibition, Dallas,Texas, USA.

[60] Rodriguez, P. G., Segura, M. K. C., and Moreno, F. J. M., 2003. Streamline Methodology using an Efficient Operator Splitting for Accurate Modelling of Capillarity and Gravity Effects. *SPE Reservoir Simulation Symposium*.

[61] Roti, F., and Dawe, R. A., 1993. Modelling Fluid Flow in Cross-bedded Sections. *Transport in Porous Media*, 12: 143-159.

[62] Silva, J. M. and Dawe, R. A., 2003. Effects of Permeability and Wettability Heterogeneities on Flow in Porous Media. *SPE Latin American and Caribbean Petroleum Engineering Conference*, Port-of-Spain, Trinidad, West Indies, 2003.

[63] Skinner, J. H., 2011. Near Wellbore Streamline Modelling for Advanced Well Completions. *Master Thesis*, Memorial University.

[64] Skinner, J. H. and Johansen, T. E., 2012. Near Wellbore Streamline Modelling: Its Novelty, Application, and Potential Use. *International Symposium and Exhibition on Formation Damage Control*, Lafayette, Louisiana, USA.

[65] Sohrab Z., 2010. Investigation of Gravity Drainage in Fractured Porous Media. *Doctoral dissertation*, University of Waterloo, Ontario, Canada.

[66] Tang, R. W., Behrens, R. A., Emanuel, A. S., 1989. Reservoir Studies Using Geostatistics To Forecast Performance, *SPE Symposium on Reservoir Simulation*, Houston, TX, USA.

[67] Thiele, M. R., 1994. Modeling Multiphase Flow in Heterogeneous Media using Streamtubes. *Doctoral dissertation*, Stanford University.

[68] Thiele, M. R., 2001. Streamline Simulation. *6th International Forum on Reservoir Simulation*, Schloss Fuschl, Austria.

[69] Thiele, M. R., Batycky, R. P., and Fenwick, D. H., 2010. Streamline Simulation for Modern Reservoir-Engineering Workflows. *Journal of Petroleum Technology Distinguished Author Series*, (64-70).

[70] van der Marck, S.C., and J. Glas., 1997. Pressure measurements during forced imbibition experiments in micro-models. *Europen Journal Mech*, B 16:681-692.

[71] Valsecch, P., McDuff, D., Chang, D. L., Huang, H., Burdette, J., Long, T., and Karmonik,C., 2012. Simulation and visualization of near-well flow. *Society of Petroleum Engineers.*

[72] Welge, H. J., 1952. A simplified method for computing oil recovery by gas or water drive, *Trans, AIME 3.108.*

Appendixes

Appendix A: Buckley-Leverett Theory

In this appendix, the Buckley-Leverett (1942) analytical 1D Riemann solution is briefly described for water flooding of an reservoir.

For 1D waterflooding of an oil reservoir, the conservation of mass is described by:

$$\phi \frac{\partial S}{\partial t} + u_t \left(\frac{\partial f_w}{\partial x} \right) = 0, \tag{A-1}$$

where ϕ is the porosity, S is the water saturation, t is the time, u_t is the total velocity $u_t = u_w + u_o$, f_w is the water fractional flow function $f_w = u_w/u_t$, and x is the travel distance from the inlet.

We assume initial saturations for the porous media and injected saturations are constant. The saturation boundary conditions are:

$$S_L = S(0, t) = 1 - S_{or}, t \geq 0, \tag{A-2}$$

$$S_R = S(x, 0) = S_{wc}, x \in [0, L], \tag{A-3}$$

where S_L and S_R are the water saturation at the inlet and outlet of the stream tube, respectively, and S_{or} is the residual saturation and S_{wc} is the connate water saturation.

Each saturation value propagates with a velocity given by:

$$v(S) = \frac{u_t}{\phi} f'(S), \tag{A-4}$$

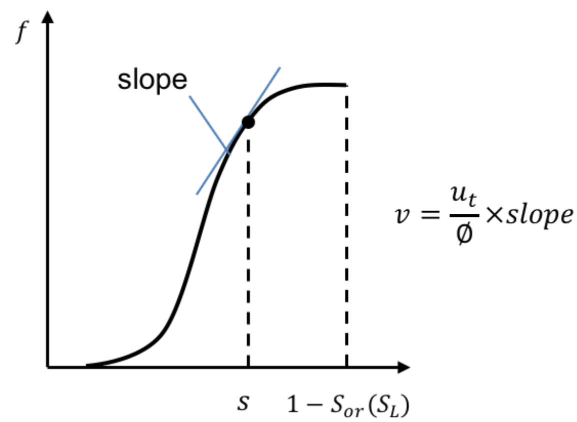


Figure A-1: Propagating Velocity for 1D Riemann Solution (Craft and Hawkins, 1991)

The location of any saturation at time t is:

$$x(t) = x_0 + \frac{u_t}{\phi} f'(S) \cdot t. \tag{A-5}$$

Consider three saturation values on the initial water saturation the slopes for each velocity are shown in Figure A-2.

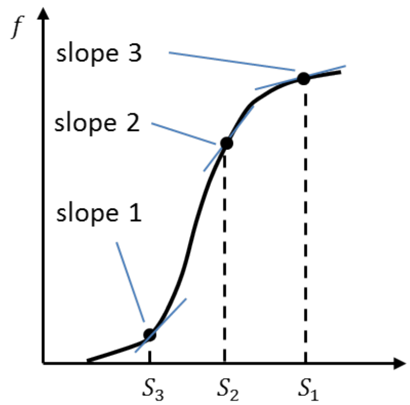


Figure A-2: Propagating Velocities for Three Different Saturation Points (Craft and Hawkins, 1991)

The associated locations for these three saturations are shown in Figure A-3. As can be seen from this figure, this may lead two saturation at the same location (x_1), which is physically impossible.

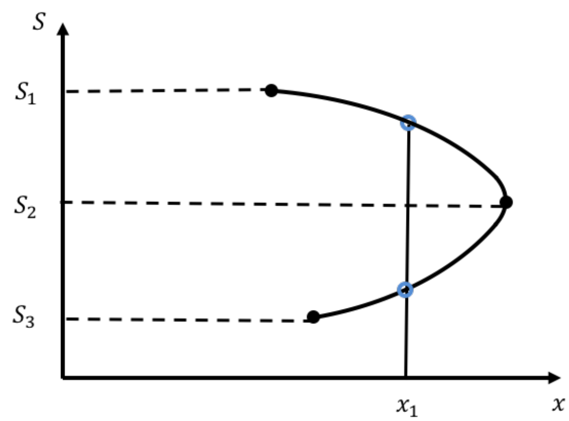


Figure A-3: Propagating Velocities for Three Different Saturation Points (Craft and Hawkins, 1991)

When velocity at trailing end of a transport chain is higher than at the leading edge, a shock wave will form.

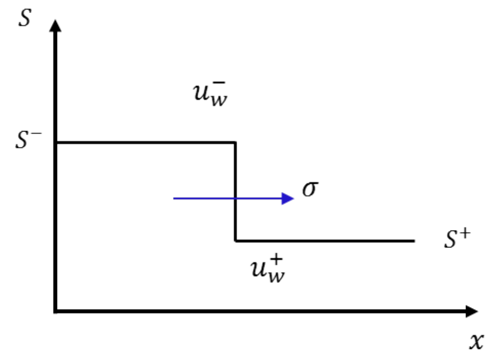


Figure A-4: Schematic of a Shock Wave (Craft and Hawkins, 1991)

According to material balance, net volume (mass) traveling into shock must equal the volume of water transported by the shock itself:

$$u_t[f(S^-) - f(S^+)]A\Delta t = \sigma(S^- - S^+)A\phi\Delta t. \tag{A-6}$$

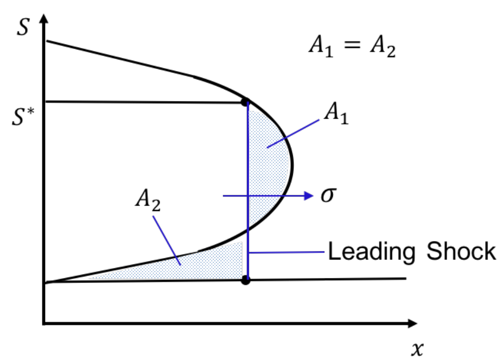


Figure A-5: Material Balance for the Shock (Craft and Hawkins, 1991)

Figure A-5 shows the material balance for the shock. In this figure, σ is the velocity of the shock value, and S^* is the water saturation value behind the shock front.

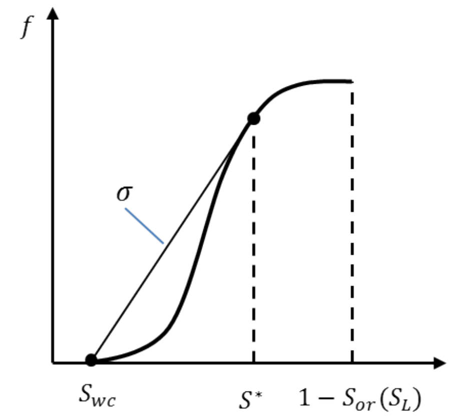


Figure A-6: Welge's Tangent Method for 1D Riemann Solution (Welge, 1952)

The shock wave with velocity σ can be determined by the tangent procedure in Figure A-6 (Welge, 1952) and must satisfy:

$$\sigma = \frac{u_t}{\phi} \frac{f(S^-) - f(S^+)}{S^- - S^+}, \tag{A-7}$$

For a shock, the velocity behind the shock is larger than ahead of shock.

$$v(S^-) \geq v(S^+) \tag{A-8}$$

The overall velocity must increase from injected state to the initial state as shown in Figure A-7.

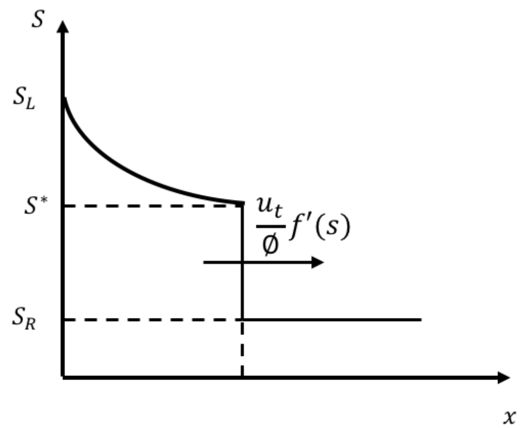


Figure A-7: Analytical 1D Riemann Solution (Welge, 1952)

Appendix B: Karakas-Tariq Skin Calculation Method

In this appendix, the main procedures for the typical skin calculation method (Karakas-Tariq, 1991) is described. It is based on finite element simulation in 2D subdomains.

1. Calculate horizontal component of skin, s_H:

$$s_H = ln(r_w/r_{we}). \tag{B-1}$$

The effective well radius, r_{we}, is given by:

$$r_{we} = \begin{cases} \frac{1}{4}L_p & if\ \theta = 0^\circ \\ \alpha_\theta(r_w + L_p) & otherwise. \end{cases} \tag{B-2}$$

Where L_p is the perforation length, r_w is the wellbore radius. The parameter α_θ is given by Figure B-1.

TABLE 1—DEPENDENCY OF r_{we} ON PHASING	
Perforation Phasing (degrees)	$r_{we}/(r_w + L_p)$
0 (360)	0.250
180	0.500
120	0.648
90	0.726
60	0.813
45	0.860

Figure B-1: α_θ used in Karakas-Tariq Method (Karakas-Tariq, 1991)

2. Calculate wellbore skin, s_{wb}:

$$s_{wb} = c_1(\theta)exp[c_2(\theta)r_{wD}], \tag{B-3}$$

where $r_{wD} = r_w/(L_p + r_w)$. Equation B-3 is valid for $0.30 \leq r_{wD} \leq 0.90$, and c_1 and c_2 are given in Figure B-2.

TABLE 2—VARIABLES c_1 AND c_2 IN EQ. 9		
Perforation Phasing (degrees)	c_1	c_2
0 (360)	1.6×10^{-1}	2.675
180	2.6×10^{-2}	4.532
120	6.6×10^{-3}	5.320
90	1.9×10^{-3}	6.155
60	3.0×10^{-4}	7.509
45	4.6×10^{-5}	8.791

Figure B-2: c_1 and c_2 used in Karakas-Tariq Method (Karakas-Tariq, 1991)

3. Calculate vertical skin, S_v:

$$s_v = 10^a h_D^{b-1} r_{pD}^b, \tag{B-4}$$

where $a = a_1 log_{10} r_{pD} + a_2$, and $b = b_1 r_{pD} + b_2$. The values of a_1, a_2, b_1, and b_2 are given in Figure B-3

TABLE 4—VERTICAL-SKIN CORRELATION COEFFICIENTS

Perforation Phasing (degrees)	a_1	a_2	b_1	b_2
0 (360)	−2.091	0.0453	5.1313	1.8672
180	−2.025	0.0943	3.0373	1.8115
120	−2.018	0.0634	1.6136	1.7770
90	−1.905	0.1038	1.5674	1.6935
60	−1.898	0.1023	1.3654	1.6490
45	−1.788	0.2398	1.1915	1.6392

Figure B-3: Parameters to Calculate Vertical Skin used in Karakas-Tariq Method (Karakas-Tariq, 1991)

The parameter h_D is defined as:

$$h_D = (h/L_p)\sqrt{K_H/K_v}, \tag{B-5}$$

and $r_{pD} = (r_p/2h)(1+\sqrt{K_H/K_v})$. K_H and K_v are the permeability in horizontal direction and vertical direction, respectively. Equation B-4 is valid for $h_D \leq 10$ and $r_{pD} \geq 0.01$.

4. Determine the combined skin effect caused by perforations, s_p:

$$s_p = s_H + s_v + s_{wb}. \tag{B-6}$$

5. Calculate the crushed zone effects:

$$s_c = \frac{h}{L_p}\left(\frac{K}{K_c} - 1\right) ln\left(\frac{r_c}{r_p}\right), \tag{B-7}$$

where K is the reservoir permeability and K_c is the crushed zone permeability.

6. Add crushed zone effect to the perforation skin:

$$s_p' = s_p + s_c. \tag{B-8}$$

7. Add damaged-zone effects. For perforations within damaged zone, the skin

caused by the combined effects of perforations and damage is:

$$s_t = \left(\frac{K}{K_d} - 1\right) ln \left(\frac{r_d}{r_w}\right) + \left(\frac{K}{K_d} - 1\right)(s_p + s_x), \tag{B-9}$$

where s_x (given in Figure B-4) is negligible for $r_d \geq 1.5(r", +L_p)$.

TABLE 5—SKIN CAUSED BY BOUNDARY EFFECT, 180° PHASING

$r_e/(r_w + L_p)$	s_x
18.0	0.000
10.0	−0.001
2.0	−0.002
1.5	−0.024
1.2	−0.085

Figure B-4: Parameters to Calculate Crushed Zone Skin used in Karakas-Tariq Method (Karakas-Tariq, 1991)

For perforations extending beyond the damaged zone, modify the well radius and the perforation length: perforations extending beyond the damaged zone, modify the well radius and the perforation length:

$$L'_p = L_p - [1 - (K_d/K)]L_d, \tag{B-10}$$

$$r'_w = r_w[1 - (\sqrt{K_d/K})]L_d, \tag{B-11}$$

where L_d is the damaged zone length.

Appendix C: Numerical Integration Methods for 3D Riemann Solution

In this appendix, the detailed steps for the numerical integration methods used in the 3D Riemann solution are demonstrated.

In the 3D Riemann solution, we use the numerical integration method to calculate the value of $\mathcal{J}(S)$. For the front saturation S^*, $\mathcal{J}(S^*)$ is:

$$\mathcal{J}(S^*) = \int_{S^*}^{S_L} \frac{f''(S)dS}{A^2\left[V^{-1}[V((x(S^*,t))\frac{\phi f'(S)}{f'(S^*)}\right]\lambda(S)}, \tag{C-12}$$

where $x(S^*,t)$ is travel distance for front saturation S^* from the injection point at time t, $V(x)$ is the volume of the stream tube from injection to x, $A(x)$ is the cross section area for the stream tube at x, $f'(S)$, $f''(S)$ is the first and second derivative of water fractional flow function with respect to water saturation S, respectively, λ is the total fluid mobility $\lambda = \lambda_o + \lambda_w$, and λ_R is the total mobility at the S_R, $\lambda_R = \lambda_{S_R}$. We use the following steps to calculate the $\mathcal{J}(S^*)$ value:

1. Use the saturation interval ΔS to define n saturation values S_i from lower integration limit S^* to the upper integration limit S_L,

$$S_i = S^* + (i-1)\Delta S, \tag{C-13}$$

   where $i \in [1,n], S_n = S_L$.

2. Calculate the first derivative $f'(S_i)$, second derivative $f''(S_i)$ of water fractional flow function and the total mobility $\lambda(S_i)$ with respect to the corresponding water saturation S_i.

3. With the known travel distance $x(S^*,t)$, determine the volume of the stream tube $V(x(S^*,t))$ from the injection to x.

4. Calculate the value of $V(x(S^*,t))\frac{\phi f'(S)}{f'(S^*)}$, then use this value to calculate the value of $A^2\left[V^{-1}[V(x(S^*,t))\frac{\phi f'(S)}{f'(S^*)}\right]$.

5. Calculate the value in Equation C-14 for each saturation value S_i:

$$\Delta\mathcal{J}(S_i) = \frac{f''(S)\Delta S}{A^2\left[V^{-1}[V((x(S^*,t))\frac{\phi f'(S)}{f'(S^*)}\right]\lambda(S)}. \tag{C-14}$$

6. Calculate $\mathcal{J}(S^*)$ by summing the $\Delta\mathcal{J}(S_i)$ for individual saturation:

$$\mathcal{J}(S^*) = \sum_{i=1}^{n} \Delta\mathcal{J}(S_i). \tag{C-15}$$

We also use the numerical method to calculate the value of $\frac{1}{\lambda_R}\int_{x(S^*,t)}^{L}\frac{dx}{A(x)}$. The procedure is:

1. Use the distance interval Δx to define n saturation values x_i from lower integration limit x to the upper integration limit L,

$$x_i = x + (i-1)\Delta x, \tag{C-16}$$

   where $i \in [1, n], x_n = L$.

2. Obtain the cross section area $A(x_i)$ for the corresponding distance x_i.

3. Calculate the value of:

$$\frac{\Delta x}{A(x_i)}. \tag{C-17}$$

4. Then, $\frac{1}{\lambda_R}\int_{x(S^*,t)}^{L}\frac{dx}{A(x)}$ can be calculated as:

$$\frac{1}{\lambda_R}\int_{x(S^*,t)}^{L}\frac{dx}{A(x)} = \frac{1}{\lambda_R}\cdot\sum_{i=1}^{n}\frac{\Delta x}{A(x_i)} \tag{C-18}$$

Appendix D: Experimental Data

D. 1 Error Analysis

In the experiment, measured data always have errors. The methods we used to measure the pore volume and porosity depend on the measurement of volume and weight. Fluctuations in the volume measurements also caused the error in flow rate calculation. These differences can be attributed to the precision of the graduated cylinders, and the response time to read volumes.

For many instruments, we assume that the reading error is $\pm$ 1/2 of the smallest division. In our experiments 10 ml graduated cylinders with the 1 ml graduation level are used to measure the fluid volume. The level of water and oil is read to the nearest 1 ml, hence, a reasonable estimate of the uncertainty in this case would be ± 0.5 ml.

The mean value and the standard deviation are used in error analysis. The standard deviation is calculated by the following equation:

$$S_N = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}, \tag{D-19}$$

where $x_1, x_2, \ldots, x_N$ are the observed values of the sample items and $\bar{x}$ is the mean value of these observations; N is the number of measurements.

We performed two replicate heterogeneous experiments. However, the connate water saturation and the residual oil saturation are different and these points cannot be controlled. Since the pressure difference for the two heterogeneous experiments are the same, these two experiments are treated as replicate runs. To measure the error of the heterogeneous experiments, the standard deviations corresponding to different parameters are calculated. Table D-1 shows the detail of standard deviation calculation for different parameters measurements based on replicate runs.

Table D-1: Standard Deviations for Heterogeneous Experimental Parameters

Parameters	**Replicate 1**	**Replicate 2**	**Average**	**Standard Deviation**
Porosity	0.426	0.417	0.422	0.006
Connate Water Saturation	0.360	0.364	0.362	0.003
Residual Oil Saturation	0.193	0.208	0.201	0.011
Breakthrough Time (*min*)	61	63	62	1.414

Table D-2: Summary of Heterogeneous Experimental Parameters

Parameters	**Value**
Porosity	0.422 ± 0.006
Connate Water Saturation	0.362 ± 0.006
Residual Oil Saturation	0.201 ± 0.011
Breakthrough Time (*min*)	62 ± 1.414

The Corey model parameters obtained by history matching method for the heterogeneous experiments are shown in Table D-3.

Hence, the Corey model for the heterogeneous experiments can be written as:

$$K_{rw} = (0.232 \pm 0.005) \left(\frac{S_w - 0.362 \pm 0.006}{0.437 \pm 0.012} \right)^4, \tag{D-20}$$

$$K_{ro} = (0.879 \pm 0.017) \left(\frac{0.799 \pm 0.006 - S_w}{0.437 \pm 0.012} \right)^2. \tag{D-21}$$

Table D-3: Standard Deviations for Heterogeneous Experimental Corey Model Parameters

Parameters	**Replicate 1**	**Replicate 2**	**Average**	**Standard Deviation**
a_w	0.235	0.228	0.232	0.005
a_o	0.891	0.867	0.879	0.017
n_w	4	4	4	0
n_o	2	2	2	0

Next, we use the homogeneous porous media to show the properties characterization.

D. 2 Absolute Permeability Measurement for Homogeneous Porous Media

Table D-4: Absolute Permeability Measurement Data

Test #	**Flow Rate** (ml/min)	P_{in} (psi)	P_{out} (psi)
1	5	14.312	14.000
2	10	14.593	14.003
3	15	14.989	14.004

For test 1, using Equation 5.1:

$$q = 5\ ml/min = 8.33 \times 10^{-8}\ m^3/s,$$

$$\mu_w = 1 \times 10^{-3}\ Pa \cdot s,$$

$$r_e = 0.1524\ m,$$

$$r_w = 0.0076\ m,$$

$$h = 0.0119\ m,$$

$$\Delta p = 0.309\ psi = 2130.48\ pa$$

$$K = \frac{8.33 \times 10^{-8} \times 1 \times 10^{-3} ln\left(\frac{0.1524}{0.0076}\right)}{2\pi \times 0.0119 \times 2130.48} = 1.54 \times 10^{-12}\ m^2.$$

Three tests are performed at three different flow rates. The average and the standard deviation are shown in Table D-5.

Table D-5: Standard Deviation of Permeability Measurements Based on Replicate Runs

Test #	**Permeability (m^2)**	**Average**	**Standard Deviation**
1	1.54×10^{-12}		
2	1.62×10^{-12}	1.58×10^{-12}	3.41×10^{-14}
3	1.58×10^{-12}		

Hence, $K = 1.58 \pm 0.03 \times 10^{-12}\ m^2$.

D. 3 Porosity Measurement for Homogeneous Porous Media

In the imbibition process, we add the read uncertainty in the calculation.

1. The total water injection volume:

   $V_1 = 1752 \pm 0.5\ ml.$

2. The total water production volume:

   $V_2 = 1359 \pm 0.5\ ml.$

Using Equation 5.2:

$$r_e = 0.1524\ m,$$

$$r_w = 0.0076\ m,$$

$$h = 0.0119\ m,$$

$$\phi = \frac{((1752 \pm 0.5) - (1359 \pm 0.5)) \times 10^{-6}}{\pi \times 0.0119 \times (0.1524^2 - 0.0079^2)} = 0.454 \pm 0.001.$$

D. 4 Connate Water Saturation Measurement for Homogeneous Porous Media

1. The volume of injected oil in the primary drainage process:

   $V_{oil} = 2032 \pm 0.5\ ml.$

2. The total volume of oil produced in the primary drainage process:

   $V_{op} = 1748 \pm 0.5\ ml.$

3. Initial oil in the porous media is:

   $V_{oil} = V_{oi} - V_{op} = (2032 \pm 0.5) - (1748 \pm 0.5) = 284 \pm 0.7\ ml.$

4. The volume of connate water that remains in the porous media is:

   $V_{wc} = V_1 - V_2 - V_{oil} = (1752 \pm 0.5) - (1359 \pm 0.5) - 284 \pm 1 = 109 \pm 1.0\ ml.$

5. The connate water saturation is:

$$S_{wc} = \frac{109 \pm 1}{(1752 \pm 0.5) - (1359 \pm 0.5)} = 0.277 \pm 0.008.$$

D. 5 Residual Oil Saturation Measurement for Homogeneous Porous Media

1. The total volume of water injected in the waterflooding process:

   $V_{wi} = 602 \pm 0.5\ ml.$

2. The total volume of oil produced is:

   $V_{op2} = 219 \pm 0.5\ ml.$

3. The volume of oil remaining in the system is:

   $V_{or} = V_{oil} - V_{op2} = 65 \pm 0.7\ ml.$

4. The residual oil saturation then can be calculated as:

$$S_{or} = \frac{65 \pm 1}{(1752 \pm 0.5) - (1359 \pm 0.5)} = 0.166 \pm 0.008.$$

D. 6 Experimental Pressure Profile

Figure D-1 and D-2 are the pressure profile for the experiments.

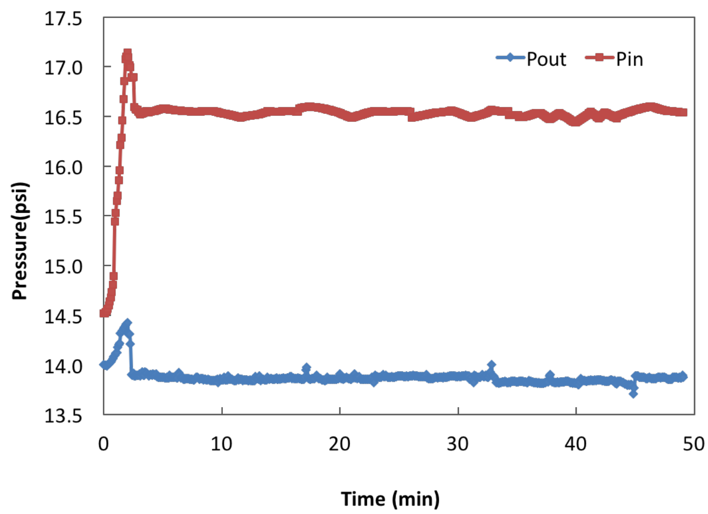


Figure D-1: Pressure Profile for Homogeneous Experiment

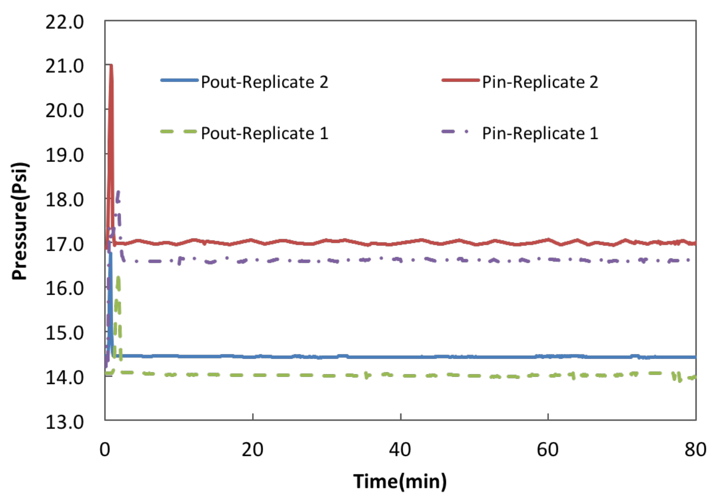


Figure D-2: Pressure Profile for Heterogeneous Experiments

D. 7 History Match Procedure

The trial and error solution seeking is applied in the homogeneous experiment (described in **5.5.2**) to obtain the relative permeabilities. The detailed history match process is:

1. In the trial and error process, we find the parameters a_w and a_o in Corey model (Equations 5.9 and 5.10) change more frequently than the exponents n_w and n_o. Hence, in the history match process, we first determine the exponents and keep them constant ($n_w = 4$ and $n_o = 2$) based on the trial and error results.

2. From experiment measurements and material balance calculations , we input the reservoir dimensions $r_w = 0.0079\ m,\ r_e = 0.1524\ m,\ h = 0.0119\ m$, porosity $\phi = 0.454$, differential pressure between the inlet and outlet $\Delta p = 17249.9\ Pa$, permeability $K = 1.58 \times 10^{-12}\ m^2$, fluid viscosities $\mu_o = 1.19 \times 10^{-3}\ P \cdot s,\ \mu_w = 1.19 \times 10^{-3}\ P \cdot s$, connate water saturation $S_{wc} = 0.277$, residual oil saturation $S_{or} = 0.166$.

3. Based on the trial and error values obtained, we input initial $n_w = 4$, $n_o = 2$, $a_{wi} = 0.8$ and $a_{oi} = 0.2$.

4. We change a_w and a_o with interval of 0.001 and simulate breakthrough times for each a_w and a_o. The breakthrough time is determined by integration of Equation 4.7 between $x = 0$ and $x = L$ using Equation 4.5 for $q(t)$, where L is the length of stream tubes.

5. Since the homogeneous experimental breakthrough time is 33 min, we define a range of $[32.5, 33.4]$ to find the two simulated boundary times. These two simulated boundary times must the in the range and closest to the boundaries,

respectively. The two boundary times are used to limit the a_w and a_o values in 6-11 below to make sure that the simulated breakthrough time has a error less than 10% compare to the experimental breakthrough time.

6. Find the corresponding a_w and a_o for the two simulated boundary times. The boundary values for the homogeneous experiment are shown in D-6.

Table D-6: History Match Boundary Times and Corresponding a_w and a_o

Breakthrough Time (min)	a_w	a_o
33.30	0.260	0.860
32.45	0.254	0.854

7. Input a fixed value of $a_o = 0.857$ in Equation 5.10, which is the average value of simulated boundary values (Table D-6), and lower boundary value of $a_w = 0.254$ in Equation 5.9. Increase a_w with an incremental of 0.001 and simulate the breakthrough time and the flow rate. The breakthrough time is determined by integration of Equation 4.7 and the flow rate is calculated by using Equation 4.5.

8. Calculate the flow rate error between the simulate results and the experimental results by using Equation D-22. Find the minimum error is 0.0177.

$$E_q = \frac{1}{N} \sum \frac{|q_s - q_e|}{q_e}, \tag{D-22}$$

where q_s and q_e are the simulated and experimental flow rate, respectively, N is the number of measurement.

Table D-7 shows the values of a_w and a_o and the corresponding flow rate error.

Table D-7: Value of a_w and a_o and Simulated Values

a_o	a_w	**Breakthrough Time** (min)	**Flow Rate Error** (%)
0.857	0.254	32.95	0.0196
0.857	0.255	32.82	0.0191
0.857	0.256	32.70	0.0186
0.857	0.257	32.58	0.0182
0.857	0.258	32.45	0.0177

9. Obtain the corresponding value of $a_w = 0.258$ which provide the smallest flow rate value with fixed value of a_o.

10. Input the fixed value of $a_w = 0.258$ and the minimum boundary value of $a_o = 0.854$. Increase a_o with interval of 0.001, simulate the breakthrough time and the flow rate. Calculate the flow rate error between the simulate results and the experimental results. Find the minimum error 0.0172 and the corresponding a_o is 0.855. Table D-8 shows the values and the corresponding flow rate error.

Table D-8: Value of a_w and a_o Values and Corresponding Flow Rate Error

a_w	a_o	**Flow Rate Error** (%)
0.258	0.854	0.0174
0.258	0.855	0.0172

11. Output history matched Corey Model parameters:

    $a_w = 0.258,\ a_o = 0.855,\ n_w = 4$, and $n_o = 2$.

Appendix E: Source Code

In this appendix, the MATLAB ® code files used in this research thesis are demonstrated.

E. 1 Two-Dimensional Open Hole Well Heterogeneous Case Streamline Simulator

```
clear all;
% Define block no. for R and theta direction
N=20;J=20;M=N*J;
% Define wellbore radius=50 m, reservoir radius=0.05m
Re=50;Rw=0.05;
% Boundary Pressures: wellbore pressure=280*10^5 pa ↩
    reservoir pressure=300*10^5 pa
Pw=280*10^5; Pe=300*10^5;
% Define the block peremability K_block=1e-12
K=1e-12.*ones(N,J,3);
uo=0.8e-3;% Define fluid viscosity cp
% Define heterogeneity block no. and permeability
HENR1=17;HENR2=14;HENT1=16;HENT2=15;
K(HENR2:HENR1-1,HENT2:HENT1-1,1)=0.25e-13;
K(HENR2:HENR1-1,HENT2:HENT1-1,2)=0.25e-13;
R=0:1:N-1;
ro=Rw*(Re/Rw).^(R./(N-1));% Calculate node radii
```

```
rb=ones(1,N+1);
rb(1,2:N)=(ro(:,2:N).*ro(:,1:N-1)).^0.5;
rb(1,[1 N+1])=[Rw^2/rb(1,2) Re^2/rb(1,N)];% Calculate boundary radii
Ro=repmat(ro',1,J);Rb=repmat(rb',1,J); % Ro, Rb Radius for nodes for all grid block
Tn=linspace(2*pi/(2*J),2*pi-2*pi/(2*J),J); % Theta Nodes angle
ttn=repmat(Tn',1,N)';% Theta Nodes angle for all grid block
Dn=360/J*pi/180;% Define theta angle
Kr=(K(:,:,1).*(cos(ttn)).^2+K(:,:,2).*(sin(ttn)).^2);% Calculate Kr from the principle permeability
Kt=(K(:,:,2).*(cos(ttn)).^2+K(:,:,1).*(sin(ttn)).^2);% Calculate Kt from the principle permeability
Mblock=Kt/uo;Mblockr=Kr/uo;% Block Mobility
Mbr=ones(N+1,J); Mbr([1 N+1],:)=Mblockr([1 N],:);% Upscaled mobility in r direction
Mbr(2:N,:)=log(Ro(2:N,:)./Ro(1:N-1,:))./((1./Mblockr(1:N-1,:).*log(Rb(2:N,:)./Ro(1:N-1,:)))+(1./Mblockr(2:N,:).*log(Ro(2:N,:)./Rb(2:N,:))));
Mbt=ones(N,J);% Upscaled mobility in the angular direction
Mbt(:,1:J-1)=2.*Mblock(:,1:J-1).*Mblock(:,2:J)./(Mblock(:,1:J-1)+Mblock(:,2:J));
Mbt(:,J)=2.*Mblock(:,1).*Mblock(:,J)./(Mblock(:,1)+Mblock(:,J));% Last column mobility in the angular direction
```

```
Tr1=Rb(1,:).*Mbr(1,:)./(Ro(1,:).*(Rb(2,:)-Rb(1,:)).*(Ro(1,:)-Rb(1,:)));
Trb=Rb(2:N,:).*Mbr(2:N,:)./(Ro(2:N,:).*(Rb(3:N+1,:)-Rb(2:N,:)).*(Ro(2:N,:)-Ro(1:N-1,:))); %Transmisibility in the radial direction
% Calculate the transmisibility coeffiecints (a,b,c,d,e)in different directions
e(1,:)=Rb(2,:).*Mbr(2,:)./(Ro(1,:).*(Rb(2,:)-Rw).*(Ro(2,:)-Ro(1,:)));
e(2:N-1,:)=Rb(3:N,:).*Mbr(3:N,:)./(Ro(2:N-1,:).*(Rb(3:N,:)-Rb(2:N-1,:)).*(Ro(3:N,:)-Ro(2:N-1,:)));
e(N,:)=Rb(N+1,:).*Mbr(1+N,:)./(Ro(N,:).*(Rb(N+1,:)-Rb(N,:)).*(Rb(N+1,:)-Ro(N,:)));
T=zeros(N,J-1);t=zeros(N,1);
b=Mbt./((Ro.^2).*(Dn^2));
bo=[t b(:,1:J-1)]; oob=[b(:,J) T];
c=[Mbt(:,J) Mbt(:,1:J-1)]./((Ro.^2).*(Dn^2));
coo=[T c(:,1)];oc=[c(:,2:J) t];
d=[Tr1;Trb];
Tre=[zeros(1,J); e(1:N-1,:)];
a=-b-c-d-e;
am=[zeros(N,1) a(:,2:J)];
x1=reshape([Trb; zeros(1,J)]',M,1);x2=reshape(Tre',M,1);
y1=reshape(oc',M,1);y2=reshape(bo',M,1);
y10=reshape(oob',M,1);y20=reshape(coo',M,1);
AA=reshape(a',M,1);
```

```
% Define Wellbore and Outter boundary pressure
DiagVecs=[x1,y10,y1,AA,y2,y20,x2];
DiagIndx =[-J,-J+1,-1,0,1,J-1,J];
D=zeros(M,1);
D(1:J)=Pw;
D(M-J+1:M)=Pe;
% Coefficient matrix
A = spdiags(DiagVecs,DiagIndx,M,M);
for i=1:J
    A(i,:)=0;
    A(i,i)=1;
    A(M-i+1,:)=0;
    A(M-i+1,M-i+1)=1;
end
u = A\D;
% Calculate node pressures
p=reshape(u,J,N);P=p';
% Find the permeability from node pressures to PO
RDo=log(Ro/Rw);RDb=log(Rb/Rw);d1=RDo(2:N,:)-RDb(2:N,:);
Kr2=Kr(2:N,:);Kr1=[Kr(2:N,J) Kr(2:N,1:J-1)];Kr4=[Kr(1:N-1,J) Kr(1:N-1,1:J-1)];Kr3=Kr(1:N-1,:);
Kt2=Kt(2:N,:);Kt1=[Kt(2:N,J) Kt(2:N,1:J-1)];Kt4=[Kt(1:N-1,J) Kt(1:N-1,1:J-1)];Kt3=Kt(1:N-1,:);
% Calculate pressure for half logarithmic point in the radial direction
```

```
Ptij=(Kr(2:N,:).*P(2:N,:)+Kr(1:N-1,:).*P(1:N-1,:))./(Kr(2:N,:)+Kr(1:N-1,:));
% Calculate pressure for half distance point in the angular direction
KRP=P.*Kt;
Prij=(KRP+[KRP(:,J) KRP(:,1:J-1)])./(Kt+[Kt(:,J) Kt(:,1:J-1)]);
P2=P(2:N,:);P1=[P(2:N,J) P(2:N,1:J-1)];P4=[P(1:N-1,J) P(1:N-1,1:J-1)];P3=P(1:N-1,:);
P12=Prij(2:N,:);P14=[Ptij(:,J) Ptij(:,1:J-1)];P34=Prij(1:N-1,:);P23=Ptij;
for i=1:N-1
    for j=1:J
            PP=[P1(i,j);P2(i,j);P3(i,j);P4(i,j);P12(i,j);P12(i,j);...
              P23(i,j);P23(i,j);P34(i,j);P34(i,j);P14(i,j);P14(i,j);0];
            mm=[Tn(1)*d1(1,1) -Tn(1) -d1(1,1) 0 0 0 0 0 0 0 0 0 1 ;...
              0 0 0 -Tn(1)*d1(1,1) Tn(1) -d1(1,1) 0 0 0 0 0 0 1; ...
              0 0 0 0 0 0 Tn(1)*d1(1,1) Tn(1) d1(1,1) 0 0 0 1;...
              0 0 0 0 0 0 0 0 0 -Tn(1)*d1(1,1) -Tn(1) d1(1,1) 1;...
              0 0 -d1(1,1) 0 0 0 0 0 0 0 0 0 1;...
```

```
                    0 0 0 0 0 -d1(1,1) 0 0 0 0 0 0 1;...
                    0 0 0 0 Tn(1) 0 0 0 0 0 0 0 1;...
                    0 0 0 0 0 0 0 Tn(1) 0 0 0 0 1;...
                    0 0 0 0 0 0 0 0 d1(1,1) 0 0 0 1;...
                    0 0 0 0 0 0 0 0 0 0 0 d1(1,1) 1;...
                    0 -Tn(1) 0 0 0 0 0 0 0 0 0 0 1;...
                    0 0 0 0 0 0 0 0 0 0 -Tn(1) 0 1;...
                    -Kr1(i,j)*Tn(1)^2/2+Kt1(i,j)*(-d1(1,1))^2/2 Kt1(i,j)*(-d1(1,1)) Kr1(i,j)*Tn(1) ...
                    Kr2(i,j)*Tn(1)^2/2-Kt2(i,j)*(-d1(1,1))^2/2 -Kt2(i,j)*(-d1(1,1)) Kr2(i,j)*Tn(1) ...
                    -Kr3(i,j)*Tn(1)^2/2+Kt3(i,j)*(-d1(1,1))^2/2 Kt3(i,j)*(-d1(1,1)) -Kr3(i,j)*Tn(1) ...
                    Kr4(i,j)*Tn(1)^2/2-Kt4(i,j)*(-d1(1,1))^2/2 -Kt4(i,j)*(-d1(1,1)) -Kr4(i,j)*Tn(1) 0];
                   Vec=mm\PP;
                   po(i,j)=Vec(13);
     end
     for j=J
end
end
% Rearrange the corner pressure to the corner point coordinates
TO=linspace(0,2*pi-2*pi/(J),J);
tn=[TO 2*pi];
po=[Prij(1,:); po; Prij(N,:)];
```

```
po=[po po(:,1)];
LRD=log(rb./Rw);
LRD(N+1)=LRD(N+1)+LRD(1);LRD(1)=0;
RD=exp(LRD);
% Calculate the coeffiecients for the log-lin pressure assumpution
for i=1:N
    for j=1:J
            PP=[po(i,j); po(i,j+1); po(i+1,j+1); po(i+1,j);]; %P=pressure
            mm=[tn(j)*LRD(i) tn(j) LRD(i) 1 ;...
               tn(j+1)*LRD(i) tn(j+1) LRD(i) 1; ...
               tn(j+1)*LRD(i+1) tn(j+1) LRD(i+1) 1;...
               tn(j)*LRD(i+1) tn(j) LRD(i+1) 1;];
               Vec=mm\PP;
               aa(i,j)=Vec(1);bb(i,j)=Vec(2);cc(i,j)=Vec(3);dd(i,j)=Vec(4);
    end
end
hold on
% Plot presure distribution and permeability
AN=linspace(0,2*pi-2*pi/J,J);
xp=Ro.*cos(repmat(AN',1,N)');
yp=Ro.*sin(repmat(AN',1,N)');
surf([xp xp(:,1)],[yp yp(:,1)],[Kr Kr(:,1)]*10^15); % surface plot for permeability
```

```
surf([xp xp(:,1)],[yp yp(:,1)],[p p(:,1)]);% surface plot for pressure
a_=aa.*(Kr.^0.5);b_=bb.*(Kr.^0.5);c_=aa.*(Kt.^0.5);d_=cc.*(Kt.^0.5);
sita=0:2*pi/200:2*pi;
plot(Re*cos(sita),Re*sin(sita),'black');
hold on
ut=-Kt/uo.*(aa.*log(Ro/Rw)./Ro+bb./Ro); % ut+ flow conterclockwise ut- flow clockwise
tn=[TO 2*pi];
T=ones(N+10,J);
for j=1:J
  w=1;
% Define the launching point coordinate: RDin, thin
  RDin=Re/Rw;
  thin=tn(j+1)+tn(2)/2;
  XX=N+1;
  YY=ceil(thin/tn(2));
  if YY==J+1;
      YY=1;
  end
% Check if the streamline reaches to the boundary
while  XX>1&& YY<=J;
    a11=a_(XX-1,YY);b11=b_(XX-1,YY);c11=c_(XX-1,YY);d11=d_(XX-1,YY);
```

```
% Check if the reservoir is homogenous, if yes, use homogeneous streamline tracing method, if not,calculate C
if abs(a11)<=10^(-10)&& abs(b11)<=10^(-10)
      RDout=RD(XX-1);
      Rd=RDout:(RDin-RDout)/10:RDin;
      ur=Kr(XX-1,YY)/uo*(cc(XX-1,YY)./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
      TT=ones(1,11)*thin;
      dt=((RDin-RDout)*Rw/10)./ur;
      Tof=sum(dt) ;
else
     C=(a11*log(RDin)+b11)^2-(c11*thin+d11)^2;
% Use r as parameterization to trace streamline
 if C<0
 n=(c11*thin+d11)/((a11*log(RDin)+b11)^2-C)^0.5;
 syms RDD
 theta=-d11/c11+(n/c11)*((a11*log(RDD)+b11)^2-C)^0.5;
 RD1=subs(solve(theta-tn(YY+1),RDD));
 RD2=subs(solve(theta-tn(YY),RDD));
 RDD1=RD1(RD1>=RD(XX-1)& RD1<=RDin);
 RDD2=RD2(RD2>=RD(XX-1)& RD2<=RDin);
    if isempty(RDD1)==0 && isreal(RDD1)==1&& size(RDD1,1)==1 %% RDD1==RDin
        if abs(RDD1-RDin)<RDin/10000
          RDD1=[];
```

```
        end
    end
    if isempty(RDD2)==0 && isreal(RDD2)==1&& size(RDD2,1)==1 %% RDD1==RDin
        if abs(RDD2-RDin)<RDin/10000
          RDD2=[];
        end
    end
    if (isempty(RDD1)==1 ||isreal(RDD1)==0) && (isempty(RDD2)==1 || isreal(RDD2)==0 )
        Rd=(RD(XX-1):(RDin-RD(XX-1))/10:RDin);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
            if min(TT)<tn(YY)-1e-5||max(TT)>tn(YY+1)+1e-5
                TT=ones(1,11)*thin;
            end
        dt=((RDin-RD(XX-1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt);
    end
    if isempty(RDD1)==0 && isreal(RDD1)==1&& size(RDD1,1)==1 && (isempty(RDD2)==1 || isreal(RDD2)==0 ) % exit from tn(YY+1)
        rr=sort([RDin RDD1]);
        Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
```

```
        dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt2);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
    end
    if isempty(RDD2)==0 && isreal(RDD2)==1&& size(RDD2,1)==1 && (isempty(RDD1)==1 || isreal(RDD1)==0 ) % exit from tn(YY)
      % Exit from RDD1
      rr=sort([RDin RDD2]);
      Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
      dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
      Tof=sum(dt2);
      TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
    end
    % Entry and exit at same eadge T1 or T2
    if isempty(RDD1)==0 && isreal(RDD1)==1&& size(RDD1,1)==2 && (isempty(RDD2)==1 || isreal(RDD2)==0 ) % entry and exit from T1
        if abs(max(RDD1)-RDin)<RDin/10000
        rr=sort(RDD1);
        Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
```

```
        dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt2);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
        else
        rr=sort([max(RDD1) RDin]);
        Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
        dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt2);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
        end
        if  max(TT)>tn(YY+1)+10^-5||min(TT)<tn(YY)-10^-5
                TT=ones(1,11)*thin;
        end
    end
    if isempty(RDD2)==0 && isreal(RDD2)==1&& size(RDD2,1)==2 && (isempty(RDD1)==1 || isreal(RDD1)==0 ) % entry and exit from T2
        if abs(max(RDD2)-RDin)<RDin/10000
        rr=sort(RDD2);
        Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
        dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((
```

```
            Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt2);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
        else
        rr=sort([max(RDD2) RDin]);
        Rd=rr(1):(rr(2)-rr(1))/10:rr(2);
        dt2=((rr(2)-rr(1))*Rw/10)./(((a11*log((Rd(1:10)+Rd(2:11))/2)+b11).^2-C).^0.5*Kr(XX-1,YY)^0.5/uo./((Rd(1:10)*Rw+Rd(2:11)*Rw)/2));
        Tof=sum(dt2);
        TT=-d11/c11+(n/c11)*((a11*log(Rd)+b11).^2-C).^0.5;
        end
        if  max(TT)>tn(YY+1)+10^-5||min(TT)<tn(YY)-10^-5
                TT=ones(1,11)*thin;
         end
    end
  end
% Use thete as parameterization to trace streamline
 if C>0
  n=(a11*log(RDin)+b11)/((c11*thin+d11)^2+C)^0.5;
  syms theta
  T1=subs(solve(R-RD(XX-1),theta));
  T2=subs(solve(R-RD(XX),theta));
  TT1=round((T1(T1>=tn(YY)&T1<=tn(YY+1))).*1000)./1000;
  TT2=round((T2(T2>=tn(YY)&T2<=tn(YY+1))).*1000)./1000;
  if isempty(TT1)==0 && isreal(TT1)==1&& size(TT1,1)==1
```

```
        if abs(TT1-thin)<thin/10000
            TT1=[];
        end
    end
    if isempty(TT2)==0 && isreal(TT2)==1&& size(TT2,1)==1
        if abs(TT2-thin)<thin/10000
            TT2=[];
        end
    end
    if (isempty(TT1)==1 || isreal(TT1)==0) && (isempty(TT2)==1 || isreal(TT2)==0)% Only exit for the theta edges
        % Enter point is not at tn(YY+1) AND tn(YY) edge
          if abs(thin-tn(YY+1))>1e-3 && abs(thin-tn(YY))>1e-3
              t1=thin:(tn(YY+1)-thin)/10:tn(YY+1);
              t2=tn(YY):(thin-tn(YY))/10:thin;
              tm1=(t1(1:10)+t1(2:11))/2;
              tm2=(t2(1:10)+t2(2:11))/2;
              Rd1=exp(-b11/a11+(n/a11)*((c11*t1+d11).^2+C).^0.5);
              Rd2=exp(-b11/a11+(n/a11)*((c11*t2+d11).^2+C).^0.5);
              Rdm1=exp(-b11/a11+(n/a11)*((c11*tm1+d11).^2+C).^0.5);
              Rdm2=exp(-b11/a11+(n/a11)*((c11*tm2+d11).^2+C).^0.5);
```

```
            dt1=(tn(YY+1)-thin)/10*uo*Rdm1./(((c11*tm1+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
            tof1=sum(dt1);
            dt2=(thin-tn(YY))/10*uo*Rdm2./(((c11*tm2+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
            tof2=sum(dt2);
        end
        % Enter point is at tn(YY+1)edge
        if abs(thin-tn(YY+1))<=1e-3&& abs(thin-tn(YY))>1e-3
            Rd1=RD(XX-1):(RDin-RD(XX-1))/10:RDin;
            t1=ones(1,11)*thin;
            ur=Kr(XX-1,YY)/uo*(cc(XX-1,YY)./((Rd1(1:10)*Rw+Rd1(2:11)*Rw)/2));
            dt1=((RDin-RD(XX-1))*Rw/10)./ur;
            tof1=-1;
            t2=tn(YY):(thin-tn(YY))/10:thin;
            tm2=(t2(1:10)+t2(2:11))/2;
            Rd2=exp(-b11/a11+(n/a11)*((c11*t2+d11).^2+C).^0.5);
            Rdm2=exp(-b11/a11+(n/a11)*((c11*tm2+d11).^2+C).^0.5);
            dt2=(thin-tn(YY))/10*uo*Rdm2./(((c11*tm2+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
            tof2=sum(dt2);
        end
         % Enter point is tn(YY) edge
```

```
        if  abs(thin-tn(YY+1))>1e-3 && abs(thin-tn(YY))<=1e-3
            t1=thin:(tn(YY+1)-thin)/10:tn(YY+1);
            tm1=(t1(1:10)+t1(2:11))/2;
            Rd1=exp(-b11/a11+(n/a11)*((c11*t1+d11).^2+C).^0.5);
            Rdm1=exp(-b11/a11+(n/a11)*((c11*tm1+d11).^2+C).^0.5);
            dt1=(tn(YY+1)-thin)/10*uo*Rdm1./(((c11*tm1+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
            tof1=sum(dt1);
            t2=ones(1,11)*thin;
            Rd2=(RD(XX-1):(RDin-RD(XX-1))/10:RDin);
            ur=abs(Kr(XX-1,YY)/uo*(cc(XX-1,YY)./((Rd2(1:10)*Rw+Rd2(2:11)*Rw)/2)));
            dt2=((RDin-RD(XX-1))*Rw/10)./ur;
            tof2=-1;
        end
        if max(Rd1)>RD(XX)+10^-5||min(Rd1)<RD(XX-1)-10^-5
              tof1=-1;
        end
        if max(Rd2)>RD(XX)+10^-5||min(Rd2)<RD(XX-1)-10^-5
              tof2=-1;
        end
        dt=[tof1 tof2];
        Tof=min(dt(dt>0));
```

```
        if Tof ==tof1
            Rd=Rd1;
            TT=t1;
        else
            Rd=Rd2;
            TT=t2;
        end
end
% Find real streamline between r and theta
if isempty(TT1)==0 && isreal(TT1)==1&& size(TT1,1)==1  && (isempty(TT2)==1 || isreal(TT2)==0 ) % exit from RD(XX-1)
        ttt=sort([thin TT1]);
        t3=ttt(2):-(ttt(2)-ttt(1))/10:ttt(1);
        tm3=(t3(1:10)+t3(2:11))/2;
        Rd3=exp(-b11/a11+(n/a11)*((c11*t3+d11).^2+C).^0.5);
        Rdm3=exp(-b11/a11+(n/a11)*((c11*tm3+d11).^2+C).^0.5);
        dt3=-(ttt(1)-ttt(2))/10*uo*Rdm3./(((c11*tm3+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
        Tof=sum(dt3);
        TT=t3;Rd=Rd3;
end
if isempty(TT2)==0 && isreal(TT2)==1&& size(TT2,1)==1  && (isempty(TT1)==1 || isreal(TT1)==0 ) % exit from RD(XX)
```

```
        ttt=sort([thin TT2]);
        t3=ttt(2):-(ttt(2)-ttt(1))/10:ttt(1);
        tm3=(t3(1:10)+t3(2:11))/2;
        Rd3=exp(-b11/a11+(n/a11)*((c11*t3+d11).^2+C).^0.5);
        Rdm3=exp(-b11/a11+(n/a11)*((c11*tm3+d11).^2+C)↩
            .^0.5);
        dt3=-(ttt(1)-ttt(2))/10*uo*Rdm3./(((c11*tm3+d11)↩
            .^2+C).^0.5*Kt(XX-1,YY)^0.5);
        Tof=sum(dt3);
        TT=t3;Rd=Rd3;
    end
    if isempty(TT1)==0 && isreal(TT1)==1&& size(TT1,1)==2  &&↩
        (isempty(TT2)==1 || isreal(TT2)==0 ) % exit from RD(↩
        XX-1)
        ttt=sort(TT1);
        t3=ttt(2):-(ttt(2)-ttt(1))/10:ttt(1);
        tm3=(t3(1:10)+t3(2:11))/2;
        Rd3=exp(-b11/a11+(n/a11)*((c11*t3+d11).^2+C).^0.5);
        Rdm3=exp(-b11/a11+(n/a11)*((c11*tm3+d11).^2+C)↩
            .^0.5);
        dt3=-(ttt(1)-ttt(2))/10*uo*Rdm3./(((c11*tm3+d11)↩
            .^2+C).^0.5*Kt(XX-1,YY)^0.5);
        Tof=sum(dt3);
        TT=t3;Rd=Rd3;
    end
```

```
    if isempty(TT2)==0 && isreal(TT2)==1&& size(TT2,1)==2  && (isempty(TT1)==1 || isreal(TT1)==0 ) % exit from RD(XX)
          ttt=sort(TT2);
          t3=ttt(2):-(ttt(2)-ttt(1))/10:ttt(1);
          tm3=(t3(1:10)+t3(2:11))/2;
          Rd3=exp(-b11/a11+(n/a11)*((c11*t3+d11).^2+C).^0.5);
          Rdm3=exp(-b11/a11+(n/a11)*((c11*tm3+d11).^2+C).^0.5);
          dt3=-(ttt(1)-ttt(2))/10*uo*Rdm3./(((c11*tm3+d11).^2+C).^0.5*Kt(XX-1,YY)^0.5);
          Tof=sum(dt3);
          TT=t3;Rd=Rd3;
    end
  end
end
% Calculate new grid block coordinates
RDout=min([Rd(1) Rd(11)]);
    if RDout==Rd(1)
        tout=TT(1);
    else
        tout=TT(11);
    end
if abs(thin-tn(YY))<0.01&&abs(thin-tn(YY+1))<0.01
    if abs(tout-tn(YY))<0.01
        YY=YY-1;
```

```
    end
    if abs(tout-tn(YY+1))<0.01
        YY=YY+1;
    end
    if abs(RDout-RD(XX-1))<0.01
        XX=XX-1;
    end
    if abs(RDout-RD(XX))<0.01
        XX=XX+1;
    end
else
    if RDout==RD(XX-1)
        XX=XX-1;
    else
        if thin> tout&& abs(tout-tn(YY))<0.01 % entry at tn(↩
            YY+1) and exit at tn(YY)
            YY=YY-1;
        end
        if thin < tout && abs(tout-tn(YY+1))<0.01 % entry at ↩
            t< tn(YY+1) and exit at tn(YY+1)9
            YY=YY+1;
        end
    end
end
if YY==J+1
    YY=1;
```

```
end
 if YY==0
    YY=J;
end
RDin=RDout;
thin=tout;
   if XX>1
     if abs(thin-tn(YY))<10^-5 && abs(RDin-RD(XX))<10^-5
          if ut(XX-1,YY)<0 && ut(XX-1,YY-1)<0 &&ut(XX,YY)*ut(XX,YY-1)>0
              YY=YY-1;
          end
     end
     if abs(thin-tn(YY+1))<10^-5 && abs(RDin-RD(XX))<10^-5
          if ut(XX-1,YY)>0 && ut(XX-1,YY+1)>0&&ut(XX,YY)*ut(XX,YY+1)>0
              YY=YY+1;
          end
     end
   end
% Plot streamline
 x=Rd.*Rw.*cos(TT);
 y=Rd.*Rw.*sin(TT);
 figure (5)
 plot(x,y,'B-')
 hold on
```

```
 % Save TOF value
 T(w,j)=Tof;
 w=w+1;
end
end
save('streamline')
```

E. 2 Three-Dimensional Homogeneous and Anisotropic Case Streamline Simulator

```
clear all
% Define block no. for R and theta directions
Nr=30;Nt=10;nz=10;Nz=nz+1;N=Nr*Nt*Nz;
% Permeability in X,Y,Z direction for ORIGNAL blocks
kx=1e-13.*ones(Nr,Nt,nz);ky=1e-13.*ones(Nr,Nt,nz);kz=1e↩
    -14.*ones(Nr,Nt,nz);
% Define wellbore radius=50 m, reservoir radius=0.05m, ↩
    reservoir hight 10*Nz
Re=50;Rw=0.05;Dz=5;dz=Dz*(0:1:nz);
% Boundary Pressures: wellbore pressure=280*10^5 pa ↩
    reservoir pressure=300*10^5 pa
Pw=280*10^5; Pe=300*10^5;
zz=10^-10;%Define digital truncation ZERO
% Permeability in X,Y,Z direction for block centered blocks
```

```
Kx(:,:,[1 Nz])=kx(:,:,[1 nz]);Kx(:,:,2:nz)=(kx(:,:,1:nz-1)+kx(:,:,2:nz))/2;
Ky(:,:,[1 Nz])=ky(:,:,[1 nz]);Ky(:,:,2:nz)=(ky(:,:,1:nz-1)+ky(:,:,2:nz))/2;
Kz(:,:,[1 Nz])=kz(:,:,[1 nz]);Kz(:,:,2:nz)=2*kz(:,:,1:nz-1).*kz(:,:,2:nz)./(kz(:,:,1:nz-1)+kz(:,:,2:nz));% Permeability at X,Y,Z direction used in point distributed
uo=0.8e-3;%Define fluid viscosity cp
R=0:1:Nr-1;
ro=Rw*(Re/Rw).^(R./(Nr-1));% Calculate node radii
rb=ones(1,Nr+1);
rb(1,2:Nr)=(ro(:,2:Nr).*ro(:,1:Nr-1)).^0.5;
rb(1,[1 Nr+1])=[Rw^2/rb(1,2) Re^2/rb(1,Nr)];% Calculate boundary radii
Ro=repmat(ro',[1,Nt,Nz]); Rb=repmat(rb',[1,Nt,Nz]); % Ro, Rb Radius for nodes for all grid blocks
Tn=linspace(2*pi/(2*Nt),2*pi-2*pi/(2*Nt),Nt); % Theta Nodes
ttn=repmat(Tn',1,Nr)';ttn=repmat(ttn,[1,1,Nz]);
Dn=360/Nt*pi/180; % Delat theta between nodes
Kr=Kx.*(cos(ttn)).^2+Ky.*(sin(ttn)).^2; % Permeability in r direction
Kt=Ky.*(cos(ttn)).^2+Kx.*(sin(ttn)).^2; % Permeability in tangent direction
Mblock=Kt./uo; Mblockr=Kr./uo; Mblockz=Kz./uo;% Block Mobility for r, tangent and Z direction
```

```
Mbr=ones(Nr+1,Nt,Nz);Mbz=zeros(Nr,Nt,Nz+1);% Upscaled mobility at r, z direction
Mbr([1 Nr+1],:,:)=Mblockr([1 Nr],:,:);% Upscaled mobility at r direction
Mbr(2:Nr,:,:)=log(Ro(2:Nr,:,:)./Ro(1:Nr-1,:,:))./((1./Mblockr(1:Nr-1,:,:).*log(Rb(2:Nr,:,:)./Ro(1:Nr-1,:,:)))+(1./Mblockr(2:Nr,:,:).*log(Ro(2:Nr,:,:)./Rb(2:Nr,:,:))));
Mbt=ones(Nr,Nt,Nz);% Upscaled mobility at theta direction
Mbt(:,1:Nt-1,:)=2.*Mblock(:,1:Nt-1,:).*Mblock(:,2:Nt,:)./(Mblock(:,1:Nt-1,:)+Mblock(:,2:Nt,:));
Mbt(:,Nt,:)=2.*Mblock(:,1,:).*Mblock(:,Nt,:)./(Mblock(:,1,:)+Mblock(:,Nt,:));% Last column is Mobility from the last to 1
Trb=zeros(Nr,Nt,Nz);
Mbz(:,:,2:Nz)=2.*Mblockz(:,:,1:Nz-1).*Mblockz(:,:,2:Nz)./(Mblockz(:,:,1:Nz-1)+Mblockz(:,:,2:Nz));
Mbr(:,:,[1 Nz])=0.5*Mbr(:,:,[1 Nz]);Mbt(:,:,[1 Nz])=0.5*Mbt(:,:,[1 Nz]);
Tr1=Rb(1,:,:).*Mbr(1,:,:)./(Ro(1,:,:).*(Rb(2,:,:)-Rb(1,:,:)).*(Ro(1,:,:)-Rb(1,:,:))); %First layer upscaled mobility at r direction is the block mobility
Trb(1:Nr-1,:,:)=Rb(2:Nr,:,:).*Mbr(2:Nr,:,:)./(Ro(2:Nr,:,:).*(Rb(3:Nr+1,:,:)-Rb(2:Nr,:,:)).*(Ro(2:Nr,:,:)-Ro(1:Nr-1,:,:)));
% Calculate the transmisibility coeffiecints a,b,c,d,e,f,g
```

```
e([1 Nr],:,:)=Rb([2 Nr+1],:,:).*Mbr([2 Nr+1],:,:)./(Ro([1 Nr],:,:).*(Rb([2 Nr+1],:,:)-Rb([1 Nr],:,:)).*([Ro(2,:,:);Rb(Nr+1,:,:)]-Ro([1 Nr],:,:)));
e(2:Nr-1,:,:)=Rb(3:Nr,:,:).*Mbr(3:Nr,:,:)./(Ro(2:Nr-1,:,:).*(Rb(3:Nr,:,:)-Rb(2:Nr-1,:,:)).*(Ro(3:Nr,:,:)-Ro(2:Nr-1,:,:)));
oob=zeros(Nr,Nt,Nz);bo=zeros(Nr,Nt,Nz);oc=zeros(Nr,Nt,Nz);coo=zeros(Nr,Nt,Nz);
b=Mbt./((Ro.^2).*(Dn^2));bo(:,2:Nt,:)=b(:,1:Nt-1,:); oob(:,1,:)=b(:,Nt,:);
c=[Mbt(:,Nt,:) Mbt(:,1:Nt-1,:)]./((Ro.^2).*(Dn^2));coo(:,Nt,:)=c(:,1,:);oc(:,1:Nt-1,:)=c(:,2:Nt,:);
d=[Tr1;Trb(1:Nr-1,:,:)];Tre=zeros(Nr,Nt,Nz);Tre(2:Nr,:,:)=e(1:Nr-1,:,:);
f=Mbz(:,:,2:Nz+1)./((Dz^2));g=Mbz(:,:,1:Nz)./((Dz^2));
e(Nr,:,1:Nz-1)=0;d(1,:,2:Nz)=0;
a=b+c+d+e+f+g;
x1=reshape(permute(Trb,[2,1,3]),N,1);x2=reshape(permute(Tre,[2,1,3]),N,1);
y1=reshape(permute(oc,[2,1,3]),N,1);y2=reshape(permute(bo,[2,1,3]),N,1);
y10=reshape(permute(oob,[2,1,3]),N,1);y20=reshape(permute(coo,[2,1,3]),N,1);
z1=reshape(permute(g,[2,1,3]),N,1);z2=reshape(permute(f,[2,1,3]),N,1);
AA=reshape(permute(a,[2,1,3]),N,1);
```

```
DiagVecs=[-z2,-x1,-y10,-y1,AA,-y2,-y20,-x2,-z1];
DiagIndx =[-Nr*Nt,-Nt,-Nt+1,-1,0,1,Nt-1,Nt,Nr*Nt];
% Coefficient matrix A for the pressure
A = spdiags(DiagVecs,DiagIndx,N,N);
% Define the boundary condition vector D
D=zeros(N,1);
t=linspace(-pi,pi,100);
D(1:Nt)=Tr1(:,:,1)*Pw;
D(N-Nt+1:N)=e(Nr,:,Nz)*Pe;
u = A\D;
for i=1
plot3(Re*cos(t),Re*sin(t),(i-1)*Dz*ones(1,100))
hold on
plot3(Rw*cos(t),Rw*sin(t),(i-1)*Dz*ones(1,100))
hold on
end
% Calculate the pressure solution
p=permute(reshape(u,Nt,Nr,Nz),[2,1,3]);
Rb1=Ro(2:Nr,:,:);Rb2=Ro(1:Nr-1,:,:);RB=Rb(2:Nr,:,:);RD1=Rb1./RB;RD2=Rb2./RB;
Kr2=Kr(2:Nr,:,:);Kr1=[Kr(2:Nr,Nt,:) Kr(2:Nr,1:Nt-1,:)];
Kr4=[Kr(1:Nr-1,Nt,:) Kr(1:Nr-1,1:Nt-1,:)];Kr3=Kr(1:Nr-1,:,:);
Kt2=Kt(2:Nr,:,:);Kt1=[Kt(2:Nr,Nt,:) Kt(2:Nr,1:Nt-1,:)];
Kt4=[Kt(1:Nr-1,Nt,:) Kt(1:Nr-1,1:Nt-1,:)];Kt3=Kt(1:Nr-1,:,:);
```

```
d1=log(RD1);d2=log(RD2);
% Calculate pressure at R boundaries
Ptij=(-d2.*Kr(2:Nr,:,:).*p(2:Nr,:,:)+d1.*Kr(1:Nr-1,:,:).*p(1:Nr-1,:,:))./(-d2.*Kr(2:Nr,:,:)+d1.*Kr(1:Nr-1,:,:));
KTP=p.*Kt; Prij=(KTP+[KTP(:,Nt,:) KTP(:,1:Nt-1,:)])./(Kt+[Kt(:,Nt,:) Kt(:,1:Nt-1,:)]);% p12,p34
P2=p(2:Nr,:,:);P1=[p(2:Nr,Nt,:) p(2:Nr,1:Nt-1,:)];P4=[p(1:Nr-1,Nt,:) p(1:Nr-1,1:Nt-1,:)];P3=p(1:Nr-1,:,:);
P12=Prij(2:Nr,:,:);P14=[Ptij(:,Nt,:) Ptij(:,1:Nt-1,:)];P34=Prij(1:Nr-1,:,:);P23=Ptij;
% Calculate corner pressures
for k=1:Nz
  for i=1:Nr-1
     for j=1:Nt
            PP=[P1(i,j,k);P2(i,j,k);P3(i,j,k);P4(i,j,k);P12(i,j,k);P12(i,j,k);...
                P23(i,j,k);P23(i,j,k);P34(i,j,k);P34(i,j,k);P14(i,j,k);P14(i,j,k);0]; %P=pressure
            mm=[Tn(1)*d1(1,1,1) -Tn(1) -d1(1,1,1) 0 0 0 0 0 0 0 0 0 1 ;...
                0 0 0 -Tn(1)*d1(1,1,1) Tn(1) -d1(1,1,1) 0 0 0 0 0 0 1; ...
                0 0 0 0 0 0 Tn(1)*d1(1,1,1) Tn(1) d1(1,1,1) 0 0 0 1;...
                0 0 0 0 0 0 0 0 0 -Tn(1)*d1(1,1,1) -Tn(1) d1(1,1,1) 1;...
```

```
                    0 0 -d1(1,1,1) 0 0 0 0 0 0 0 0 0 1;...
                    0 0 0 0 0 -d1(1,1,1) 0 0 0 0 0 0 1;...
                    0 0 0 0 Tn(1) 0 0 0 0 0 0 0 1;...
                    0 0 0 0 0 0 0 Tn(1) 0 0 0 0 1;...
                    0 0 0 0 0 0 0 0 d1(1,1,1) 0 0 0 1;...
                    0 0 0 0 0 0 0 0 0 0 0 d1(1,1,1) 1;...
                    0 -Tn(1) 0 0 0 0 0 0 0 0 0 0 1;...
                    0 0 0 0 0 0 0 0 0 0 -Tn(1) 0 1;...
                    -Kr1(i,j)*Tn(1)^2/2+Kt1(i,j)*(-d1(1,1,1))^2/2 Kt1(i,j)*(-d1(1,1,1)) Kr1(i,j)*Tn(1) ...
                    Kr2(i,j)*Tn(1)^2/2-Kt2(i,j)*(-d1(1,1,1))^2/2 -Kt2(i,j)*(-d1(1,1,1)) Kr2(i,j)*Tn(1) ...
                    -Kr3(i,j)*Tn(1)^2/2+Kt3(i,j)*(-d1(1,1,1))^2/2 Kt3(i,j)*(-d1(1,1,1)) -Kr3(i,j)*Tn(1) ...
                    Kr4(i,j)*Tn(1)^2/2-Kt4(i,j)*(-d1(1,1,1))^2/2 -Kt4(i,j)*(-d1(1,1,1)) -Kr4(i,j)*Tn(1) 0];
                 Vec=mm\PP;
                 po(i,j,k)=Vec(13);
        end
    end
end
PO=[Prij(1,:,:);po;Prij(Nr,:,:)];
```

```
PO=[PO PO(:,1,:)];
Roo=Rb./Rw;LRD=log(Roo(:,1)');LRD(Nr+1)=LRD(Nr+1)+LRD(1);↩
    LRD(1)=0;
RD=exp(LRD);
for i=1:Nz
PRZ(:,i)=PO(:,1,i);
end
xro=repmat((RD*Rw)',1,Nz);
yz=0:Dz:nz*Dz;
yz=repmat(yz',1,Nr+1);yz=yz';
tn=linspace(0,2*pi,Nt+1);
aa=ones(Nr,Nt,Nz-1);bb=ones(Nr,Nt,Nz-1);cc=ones(Nr,Nt,Nz-1)↩
    ;dd=ones(Nr,Nt,Nz-1);
ee=ones(Nr,Nt,Nz-1);ff=ones(Nr,Nt,Nz-1);gg=ones(Nr,Nt,Nz-1)↩
    ;hh=ones(Nr,Nt,Nz-1);
kr=kx.*(cos(ttn(:,:,1:nz))).^2+ky.*(sin(ttn(:,:,1:nz))).^2;↩
     % Permeability in r direction
kt=ky.*(cos(ttn(:,:,1:nz))).^2+kx.*(sin(ttn(:,:,1:nz))).^2;↩
     % Permeability in tangent direction
% Calculate the coeffiecients for the bilin-log pressure ↩
    assumption
for i=1:Nr
     for j=1:Nt
          for k=1:Nz-1
               P=[PO(i,j,k); PO(i,j+1,k); PO(i+1,j+1,k); PO(i↩
                   +1,j,k);...
```

```
        PO(i,j,k+1); PO(i,j+1,k+1); PO(i+1,j+1,k+1); PO(i+1,j,k+1)]; %P=pressure
    A=[tn(j)*LRD(i)*Dz*(k-1) tn(j)*LRD(i) tn(j)*Dz*(k-1) LRD(i)*Dz*(k-1) tn(j) LRD(i) Dz*(k-1) 1;...
       tn(j+1)*LRD(i)*Dz*(k-1) tn(j+1)*LRD(i) tn(j+1)*Dz*(k-1) LRD(i)*Dz*(k-1) tn(j+1) LRD(i) Dz*(k-1) 1; ...
       tn(j+1)*LRD(i+1)*Dz*(k-1) tn(j+1)*LRD(i+1) tn(j+1)*Dz*(k-1) LRD(i+1)*Dz*(k-1) tn(j+1) LRD(i+1) Dz*(k-1) 1;...
       tn(j)*LRD(i+1)*Dz*(k-1) tn(j)*LRD(i+1) tn(j)*Dz*(k-1) LRD(i+1)*Dz*(k-1) tn(j) LRD(i+1) Dz*(k-1) 1;...
       tn(j)*LRD(i)*Dz*k tn(j)*LRD(i) tn(j)*Dz*k LRD(i)*Dz*k tn(j) LRD(i) Dz*k 1;...
       tn(j+1)*LRD(i)*Dz*k tn(j+1)*LRD(i) tn(j+1)*Dz*k LRD(i)*Dz*k tn(j+1) LRD(i) Dz*k 1; ...
       tn(j+1)*LRD(i+1)*Dz*k tn(j+1)*LRD(i+1) tn(j+1)*Dz*k LRD(i+1)*Dz*k tn(j+1) LRD(i+1) Dz*k 1;...
       tn(j)*LRD(i+1)*Dz*k tn(j)*LRD(i+1) tn(j)*Dz*k LRD(i+1)*Dz*k tn(j) LRD(i+1) Dz*k 1;];
       Vec=A\P;
```

```
                    aa(i,j,k)=Vec(1);bb(i,j,k)=Vec(2);cc(i,j,k)=Vec(3);dd(i,j,k)=Vec(4);
                    ee(i,j,k)=Vec(5);ff(i,j,k)=Vec(6);gg(i,j,k)=Vec(7);hh(i,j,k)=Vec(8);
          end
     end
end
TOF=0;
for m=nz
for k=3
for j=1:Nt
 % Define the launching point coordinate: RDin, thin,zin
  RDin=Re/Rw;
  thin=tn(j)+(k-1)*tn(2)/5+tn(2)/10;
  zin=(m-1)*Dz+0.5*Dz;
 % Determine the grid block coordinates
  XX=Nr+1;
  YY=ceil(thin/tn(2));
  ZZ=ceil(zin/Dz);
  if YY==Nt+1;
      YY=1;
  end
  if YY==0
      YY=Nt;
  end
 % Check if the streamline reaches to the boundary
```

```
while RDin>1&&ZZ>0
  % Use r as parameterization to trace the streamline
  if abs(aa(XX-1,YY,ZZ)*thin*zin+bb(XX-1,YY,ZZ)*thin+dd(XX-1,YY,ZZ)*zin+ff(XX-1,YY,ZZ))>zz;
       step=-(RDin*Rw-RD(XX-1)*Rw)/19;
       r_r=RDin*Rw:step:RD(XX-1)*Rw;% upper and lower limitation for r
       t_r=zeros(1,length(r_r));
       z_r=zeros(1,length(r_r));
       t_r(1)=thin; z_r(1)=zin;% initial condition
       for i=1:(length(r_r)-1)% calculation loop
           F_tr=@(lrd,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_r(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_r(i)+ee(XX-1,YY,ZZ))/...
             (kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_r(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_r(i)+ff(XX-1,YY,ZZ)));
           F_zr=@(lrd,z) kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_r(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_r(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ))/...
             (kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_r(i)+bb(XX-1,YY,ZZ)*t_r(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ)));
           k_1=F_tr(r_r(i),t_r(i));
           k_2=F_tr(r_r(i)+0.5*step,t_r(i)+0.5*step*k_1);
```

```
            k_3=F_tr((r_r(i)+0.5*step),(t_r(i)+0.5*step*k_2));
            k_4=F_tr((r_r(i)+step),(t_r(i)+k_3*step));
            t_r(i+1)=t_r(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
            kk_1=F_zr(r_r(i),z_r(i));
            kk_2=F_zr(r_r(i)+0.5*step,z_r(i)+0.5*step*kk_1);
            kk_3=F_zr((r_r(i)+0.5*step),(z_r(i)+0.5*step*kk_2));
            kk_4=F_zr((r_r(i)+step),(z_r(i)+kk_3*step));
            z_r(i+1)=z_r(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
        end
        % Determine if the solution is possible, if possible calculate the
        % TOF for this streamline
        Rd=r_r/Rw;
        if any(t_r-0<=zz)
            t_r=t_r+2*pi;
            if any(t_r<tn(Nt)-zz)||any(t_r>tn(Nt+1)+zz)||any(z_r<dz(ZZ)-zz)||any(z_r>dz(ZZ+1)+zz)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if any(t_r-2*pi>=1e-4)
```

```
            t_r=t_r-2*pi;
            if any(t_r<tn(1)-zz)||any(t_r>tn(2)+zz)||any(z_r<dz(ZZ)-zz)||any(z_r>dz(ZZ+1)+zz)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if all(t_r-2*pi<1e-4)&&all(t_r-0>=1e-4)
            if any(t_r<tn(YY)-1e-4)||any(t_r>tn(YY+1)+1E-4)||any(z_r<dz(ZZ)-zz)||any(z_r>dz(ZZ+1)+zz)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if isempty(r_r)==0
            x=Rd*Rw.*cos(t_r);
            y=Rd*Rw.*sin(t_r);
            length_r=0;
            for i=1:19
                length_r=length_r+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_r(i+1)-z_r(i))^2)^0.5;
            end
        dt_r=((RDin*Rw-RD(XX-1)*Rw)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kr(XX-1,YY,ZZ)/uo...
        .*(aa(XX-1,YY,ZZ).*((t_r(1:19)+t_r(2:20))/2).*((z_r(1:19)+z_r(2:20))/2)+bb(XX-1,YY,ZZ).*((t_r(1:19)+t_r(2:20))/2)...
```

```
        +dd(XX-1,YY,ZZ).*((z_r(1:19)+z_r(2:20))/2)+ff(XX-1,YY,ZZ)));
        T_r=sum(dt_r);
        end
    else
        r_r=[];t_r=[];z_r=[];
    end
    % Use theta as parameterization to trace the streamline
    if abs(aa(XX-1,YY,ZZ)*log(RDin)*zin+bb(XX-1,YY,ZZ)*log(RDin)+cc(XX-1,YY,ZZ)*zin+ee(XX-1,YY,ZZ))>zz;
       if abs(thin-tn(YY))>zz;
        step=-(thin-tn(YY))/19;% Exit at tn(YY)
        t_t1=thin:step:tn(YY);% upper and lower limitation for theta
        r_t1=zeros(1,length(t_t1));
        z_t1=zeros(1,length(t_t1));
        r_t1(1)=RDin*Rw; z_t1(1)=zin;% initial condition
        for i=1:(length(t_t1)-1)% calculation loop
          F_tr=@(t,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t1(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_t1(i)+ff(XX-1,YY,ZZ))/...
              (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t1(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_t1(i)+ee(XX-1,YY,ZZ)));
```

```
            F_tz=@(t,z) kz(XX-1,YY,ZZ)*r_t1(i)*(aa(XX-1,YY,ZZ)*t*log(r_t1(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_t1(i)/Rw)+gg(XX-1,YY,ZZ))/...
                (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_t1(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_t1(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ)));
            k_1=F_tr(t_t1(i),r_t1(i));
            k_2=F_tr(t_t1(i)+0.5*step,r_t1(i)+0.5*step*k_1);
            k_3=F_tr((t_t1(i)+0.5*step),(r_t1(i)+0.5*step*k_2));
            k_4=F_tr((t_t1(i)+step),(r_t1(i)+k_3*step));
            r_t1(i+1)=r_t1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
            kk_1=F_tz(t_t1(i),z_t1(i));
            kk_2=F_tz(t_t1(i)+0.5*step,z_t1(i)+0.5*step*kk_1);
            kk_3=F_tz((t_t1(i)+0.5*step),(z_t1(i)+0.5*step*kk_2));
            kk_4=F_tz((t_t1(i)+step),(z_t1(i)+kk_3*step));
            z_t1(i+1)=z_t1(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
        end
        Rd=r_t1/Rw;
        % Check if Rd or Z out of grid block, if out, ignore this streamline
```

```
        % if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-1E-4)||any(Rd>RD(XX)+1E-4)||any(z_t1<dz(ZZ)-zz)||any(z_t1>dz(ZZ+1)+zz)
            r_t1=[];t_t1=[];z_t1=[];
        else
            x=Rd*Rw.*cos(t_t1);
            y=Rd*Rw.*sin(t_t1);
            length_t1=0;
            for i=1:19
                length_t1=length_t1+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_t1(i+1)-z_t1(i))^2)^0.5;
            end
            dt_t1=((thin-tn(YY))/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kt(XX-1,YY,ZZ)/uo.*(aa(XX-1,YY,ZZ).*...
         ((log(r_t1(1:19)/Rw)+log(r_t1(2:20)/Rw))/2).*((z_t1(1:19)+z_t1(2:20))/2)+bb(XX-1,YY,ZZ).*((log(r_t1(1:19)/Rw)+log(r_t1(2:20)/Rw))/2)...
         +cc(XX-1,YY,ZZ).*((z_r(1:19)+z_r(2:20))/2)+ee(XX-1,YY,ZZ)));
         T_t1=sum(dt_t1);
       end
      else
         r_t1=[];t_t1=[];z_t1=[];
      end
```

```
    if abs(tn(YY+1)-thin)>zz;
      step=(tn(YY+1)-thin)/19;% Exit at tn(YY+1)
      t_t2=thin:step:tn(YY+1);% upper and lower limitation for lrd
      r_t2=zeros(1,length(t_t2));
      z_t2=zeros(1,length(t_t2));
      r_t2(1)=RDin*Rw; z_t2(1)=zin;% initial condition
      for i=1:(length(t_t2)-1)% calculation loop
        F_tr=@(t,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t2(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_t2(i)+ff(XX-1,YY,ZZ))/...
          (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t2(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_t2(i)+ee(XX-1,YY,ZZ)));
        F_tz=@(t,z) kz(XX-1,YY,ZZ)*r_t2(i)*(aa(XX-1,YY,ZZ)*t*log(r_t2(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_t2(i)/Rw)+gg(XX-1,YY,ZZ))/...
          (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_t2(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_t2(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ)));
        k_1=F_tr(t_t2(i),r_t2(i));
        k_2=F_tr(t_t2(i)+0.5*step,r_t2(i)+0.5*step*k_1);
        k_3=F_tr((t_t2(i)+0.5*step),(r_t2(i)+0.5*step*k_2));
        k_4=F_tr((t_t2(i)+step),(r_t2(i)+k_3*step));
```

```
            r_t2(i+1)=r_t2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
            kk_1=F_tz(t_t2(i),z_t2(i));
            kk_2=F_tz(t_t2(i)+0.5*step,z_t2(i)+0.5*step*kk_1);
            kk_3=F_tz((t_t2(i)+0.5*step),(z_t2(i)+0.5*step*kk_2));
            kk_4=F_tz((t_t2(i)+step),(z_t2(i)+kk_3*step));
            z_t2(i+1)=z_t2(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
        end
        Rd=r_t2/Rw;
        % Check if Rd or Z out of grid block, if out, ignore this streamline
        % if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-1E-4)||any(Rd>RD(XX)+1E-4)||any(z_t2<dz(ZZ)-1E-4)||any(z_t2>dz(ZZ+1)+1E-4)
            r_t2=[];t_t2=[];z_t2=[];
        else
            x=Rd.*Rw.*cos(t_t2);
            y=Rd.*Rw.*sin(t_t2);
            length_t2=0;
            for i=1:19
                length_t2=length_t2+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_t2(i+1)-z_t2(i))^2)^0.5;
```

```
              end
            dt_t2=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kt(XX-1,YY,ZZ)/uo.*(aa(XX-1,YY,ZZ).*...
          ((log(r_t2(1:19)/Rw)+log(r_t2(2:20)/Rw))/2).*((z_t2(1:19)+z_t2(2:20))/2)+bb(XX-1,YY,ZZ).*((log(r_t2(1:19)/Rw)+log(r_t2(2:20)/Rw))/2)...
          +cc(XX-1,YY,ZZ).*((z_r(1:19)+z_r(2:20))/2)+ee(XX-1,YY,ZZ)));
          T_t2=sum(dt_t2);
        end
      else
          r_t2=[];t_t2=[];z_t2=[];
      end
        if isempty(r_t1)==1&& isempty(r_t2)==1 % both empty
           lrd_t=[];t_t=[];z_t=[];
        end
        if isempty(r_t1)==1&& isempty(r_t2)==0 % t1 empty
           lrd_t=r_t2;t_t=t_t2;z_t=z_t2;T_t=T_t2;
        end
        if isempty(r_t2)==1&& isempty(r_t1)==0 % t2 empty
            lrd_t=r_t1;t_t=t_t1;z_t=z_t1;T_t=T_t1;
        end
        if isempty(r_t1)==0&& isempty(r_t2)==0 % both not empty
```

```
            if T_t1<T_t2
               lrd_t=r_t1;t_t=t_t1;z_t=z_t1;T_t=T_t1;
            else
               lrd_t=r_t2;t_t=t_t2;z_t=z_t2;T_t=T_t2;
            end
        end
    else
      lrd_t=[];t_t=[];z_t=[];
    end
    % Use z as parameterization to trace streamline
    if abs(aa(XX-1,YY,ZZ)*log(RDin)*thin+cc(XX-1,YY,ZZ)*↩
       thin+dd(XX-1,YY,ZZ)*log(RDin)+gg(XX-1,YY,ZZ))>zz;
       if abs(dz(ZZ+1)-zin)>=zz
         step=(dz(ZZ+1)-zin)/19; % out from dz(ZZ+1)
         z_z1=zin:step:dz(ZZ+1);% upper and lower ↩
            limitation for lrd
         t_z1=zeros(1,length(z_z1));
         r_z1=zeros(1,length(z_z1));
         t_z1(1)=thin; r_z1(1)=RDin*Rw; % initial condition
         for i=1:(length(z_z1)-1)% calculation loop
            F_tz=@(z,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*↩
               log(r_z1(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_z1(i)/↩
               Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ))/...
                (kz(XX-1,YY,ZZ)*r_z1(i)*(aa(XX-1,YY,ZZ)*t*↩
                   log(r_z1(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX↩
                   -1,YY,ZZ)*log(r_z1(i)/Rw)+gg(XX-1,YY,ZZ)↩
```

```
));
        F_zr=@(z,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_z1(i)+bb(XX-1,YY,ZZ)*t_z1(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ))/...
            (kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_z1(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_z1(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ)));
        k_1=F_tz(z_z1(i),t_z1(i));
        k_2=F_tz(z_z1(i)+0.5*step,t_z1(i)+0.5*step*k_1);
        k_3=F_tz((z_z1(i)+0.5*step),(t_z1(i)+0.5*step*k_2));
        k_4=F_tz((z_z1(i)+step),(t_z1(i)+k_3*step));
        t_z1(i+1)=t_z1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
        kk_1=F_zr(z_z1(i),r_z1(i));
        kk_2=F_zr(z_z1(i)+0.5*step,r_z1(i)+0.5*step*kk_1);
        kk_3=F_zr((z_z1(i)+0.5*step),(r_z1(i)+0.5*step*kk_2));
        kk_4=F_zr((z_z1(i)+step),(r_z1(i)+kk_3*step));
        r_z1(i+1)=r_z1(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
    end
    Rd=r_z1/Rw;
```

```
        % Check if theta or rd out of grid block, if out, ignore this streamline
        % if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-zz)||any(Rd>RD(XX)+zz)||any(t_z1<tn(YY)-zz)||any(t_z1>tn(YY+1)+zz)
            r_z1=[];t_z1=[];z_z1=[];
        else
            x=Rd.*Rw.*cos(t_z1);
            y=Rd.*Rw.*sin(t_z1);
            length_z1=0;
            for i=1:19
                length_z1=length_z1+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_z1(i+1)-z_z1(i))^2)^0.5;
            end
            dt_z1=((dz(ZZ+1)-zin)/19)./(Kz(XX-1,YY,ZZ)/uo.*(aa(XX-1,YY,ZZ).*((t_z1(1:19)+t_z1(2:20))/2)...
        .*((log(r_z1(1:19)/Rw)+log(r_z1(2:20)/Rw))/2)+cc(XX-1,YY,ZZ).*((t_z1(1:19)+t_z1(2:20))/2)...
          +dd(XX-1,YY,ZZ).*((log(r_z1(1:19)/Rw)+log(r_z1(2:20)/Rw))/2)+gg(XX-1,YY,ZZ)));
          T_z1=sum(dt_z1);
        end
      else
          r_z1=[];t_z1=[];z_z1=[];
```

```
    end
    if abs(zin-dz(ZZ))>=zz
      step=-(zin-dz(ZZ))/19;% out from dz(ZZ)
      z_z2=zin:step:dz(ZZ);% upper and lower limitation for lrd
      t_z2=zeros(1,length(z_z2));
      r_z2=zeros(1,length(z_z2));
      t_z2(1)=thin; r_z2(1)=RDin*Rw;% initial condition
      for i=1:(length(z_z2)-1)% calculation loop
        F_tz=@(z,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_z2(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_z2(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ))/...
            (kz(XX-1,YY,ZZ)*r_z2(i)*(aa(XX-1,YY,ZZ)*t*log(r_z2(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_z2(i)/Rw)+gg(XX-1,YY,ZZ)));
        F_zr=@(z,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_z2(i)+bb(XX-1,YY,ZZ)*t_z2(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ))/...
            (kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_z2(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_z2(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ)));
        k_1=F_tz(z_z2(i),t_z2(i));
        k_2=F_tz(z_z2(i)+0.5*step,t_z2(i)+0.5*step*k_1);
```

```
            k_3=F_tz((z_z2(i)+0.5*step),(t_z2(i)+0.5*step*k_2));
            k_4=F_tz((z_z2(i)+step),(t_z2(i)+k_3*step));
            t_z2(i+1)=t_z2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
            kk_1=F_zr(z_z2(i),r_z2(i));
            kk_2=F_zr(z_z2(i)+0.5*step,r_z2(i)+0.5*step*kk_1);
            kk_3=F_zr((z_z2(i)+0.5*step),(r_z2(i)+0.5*step*kk_2));
            kk_4=F_zr((z_z2(i)+step),(r_z2(i)+kk_3*step));
            r_z2(i+1)=r_z2(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
        end
        Rd=r_z2/Rw;
        % Check if theta or rd out of grid block,if out,ignore this streamline
        % if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-zz)||any(Rd>RD(XX)+zz)||any(t_z2<tn(YY)-zz)||any(t_z2>tn(YY+1)+zz)
            r_z2=[];t_z2=[];z_z2=[];
        else
            x=Rd.*Rw.*cos(t_z2);
            y=Rd.*Rw.*sin(t_z2);
            length_z2=0;
```

```
                for i=1:19
                  length_z2=length_z2+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_z2(i+1)-z_z2(i))^2)^0.5;
                end
                dt_z2=((zin-dz(ZZ))/19)./(Kz(XX-1,YY,ZZ)/uo.*(aa(XX-1,YY,ZZ).*((t_z2(1:19)+t_z2(2:20))/2)...
            .*((log(r_z2(1:19)/Rw)+log(r_z2(2:20)/Rw))/2)+cc(XX-1,YY,ZZ).*((t_z2(1:19)+t_z2(2:20))/2)...
              +dd(XX-1,YY,ZZ).*((log(r_z2(1:19)/Rw)+log(r_z2(2:20)/Rw))/2)+gg(XX-1,YY,ZZ)));
              T_z2=sum(dt_z2);
            end
          else
              r_z2=[];t_z2=[];z_z2=[];
          end
          if isempty(r_z1)==1&& isempty(r_z2)==1 % both empty
               lrd_z=[];t_z=[];z_z=[];
          end
          if isempty(r_z1)==1&& isempty(r_z2)==0 % t1 empty
              lrd_z=r_z2;t_z=t_z2;z_z=z_z2;T_z=T_z2;
          end
          if isempty(r_z2)==1&& isempty(r_z1)==0 % t2 empty
              lrd_z=r_z1;t_z=t_z1;z_z=z_z1;T_z=T_z1;
          end
```

```
        if isempty(r_z1)==0&& isempty(r_z2)==0 % both not empty
            if T_z1<T_z2
                lrd_z=r_z1;t_z=t_z1;z_z=z_z1;T_z=T_z1;
            else
                lrd_z=r_z2;t_z=t_z2;z_z=z_z2;T_z=T_z2;
            end
        end
    else
        lrd_z=[];t_z=[];z_z=[];
    end
% Find real streamline among r,t,z
    if isempty(r_r)==1&& isempty(lrd_t)==1&& isempty(lrd_z)==0
        lrd=lrd_z;t=t_z;z=z_z;tof=T_z;
    end
    if isempty(r_r)==1&& isempty(lrd_t)==0&& isempty(lrd_z)==1
        lrd=lrd_t;t=t_t;z=z_t;tof=T_t;
    end
    if isempty(r_r)==0&& isempty(lrd_t)==1&& isempty(lrd_z)==1
        lrd=r_r;t=t_r;z=z_r;tof=T_r;
    end
    if isempty(r_r)==1&& isempty(lrd_t)==0&& isempty(lrd_z)==0
```

```
        if T_t<T_z
                lrd=lrd_t;t=t_t;z=z_t;tof=T_t;
        else
                lrd=lrd_z;t=t_z;z=z_z;tof=T_z;
        end
    end
    if isempty(r_r)==0&& isempty(lrd_t)==0&& isempty(lrd_z)==1
        if T_t<T_r
                lrd=lrd_t;t=t_t;z=z_t;tof=T_t;
        else
                lrd=r_r;t=t_r;z=z_r;tof=T_r;
        end
    end
    if isempty(r_r)==0&& isempty(lrd_t)==1&& isempty(lrd_z)==0
        if T_z<T_r
                lrd=lrd_z;t=t_z;z=z_z;tof=T_z;
        else
                lrd=r_r;t=t_r;z=z_r;tof=T_r;
        end
    end
    if isempty(r_r)==0&& isempty(lrd_t)==0&& isempty(lrd_z)==0
       l=[T_r T_t T_z];
       x=find(l==min(l));
```

```
        if x==1
            lrd=r_r;t=t_r;z=z_r;tof=T_r;
        end
        if x==2
            lrd=lrd_t;t=t_t;z=z_t;tof=T_t;
        end
        if x==3
            lrd=lrd_z;t=t_z;z=z_z;tof=T_z;
        end
    end
    if isempty(r_r)==1&& isempty(lrd_t)==1&& isempty(lrd_z)↩
        ==1
         input('no streamline found')
    end
    Rd=lrd/Rw;
    if abs(RDin-Rd(20))<=zz
        RDout=Rd(1);tout=t(1);zout=z(1);
    end
    if abs(RDin-Rd(1))<=zz
         RDout=Rd(20);tout=t(20);zout=z(20);
    end
    % Plot streamline
    x=Rd.*Rw.*cos(t);
    y=Rd.*Rw.*sin(t);
    plot3(x,y,z,'black')
    % Save TOF to total TOF value
```

```
    TOF=TOF+tof;
    hold on
    % Calculate new grid block coordinates
    if RDout==1;
          XX=1;
    end
    for i=1:Nr
        if RDout>RD(i)+zz&&RDout<=RD(i+1)+zz
            XX_out=i+1;
        end
    end
    YY_out=ceil(tout/tn(2));
    ZZ_out=ceil(zout/Dz);
    if XX_out==XX&&ZZ_out==ZZ
      if YY_out==YY
         if abs(tout-tn(YY))<0.01
            YY=YY-1;
         end
         if abs(tout-tn(YY+1))<0.01
           YY=YY+1;
         end
      else
         YY=YY_out;
      end
    end
    if YY==Nt+1;
```

```
      YY=1;
    end
    if YY==0
      YY=Nt;
    end
    thin=tout;
    zin=zout;
    RDin=RDout;
    XX=XX_out;
    ZZ=ZZ_out;
    if abs(thin-0)<=zz
        thin=2*pi;
        YY=Nt;
    end
  end
end
end
end
```

E. 3 Two-Dimensional Perforated Well Streamline Simulator

```
clear all
N=100;J=150;M=N*J;% Block no. for R and theta directions
```

```
Re=20;Rw=0.15; % Define wellbore radius=50 m, reservoir radius=0.05m
Pw=250*10^5; Pe=300*10^5;% Boundary Pressures: wellbore pressure=280*10^5 pa reservoir pressure=300*10^5 pa
K_block=1e-12;K=K_block.*ones(N,J,3);% Define the block peremability K_block=1e-12 m^2;
K_D=0.5*1e-12;K(1:7,:,:)=K_D;% Define the damaged zone peremability K_D=0.5*1e-12 m^2;
Fluid.swc=0.2;Fluid.sor=0.15;% critical saturation point
uw=1e-3;uo=10e-3;%Define fluid viscosity cp
R=0:1:N-1;
ro=Rw*(Re/Rw).^(R./(N-1));% Calculate node radii
rb=ones(1,N+1);
rb(1,2:N)=(ro(:,2:N).*ro(:,1:N-1)).^0.5;
rb(1,[1 N+1])=[Rw^2/rb(1,2) Re^2/rb(1,N)];% Calculate boundary radii
Ro=repmat(ro',1,J);Rb=repmat(rb',1,J); % Ro, Rb Radius for nodes for all grid blocks
Tn=linspace(2*pi/(2*J),2*pi-2*pi/(2*J),J); % Theta Nodes angle
ttn=repmat(Tn',1,N)';% Theta Nodes angle for all grid blocks
Dn=360/J*pi/180;% Define theta angle
Kr=(K(:,:,1).*(cos(ttn)).^2+K(:,:,2).*(sin(ttn)).^2);% Calculate Kr from the principle permeability
```

```
Kt=(K(:,:,2).*(cos(ttn)).^2+K(:,:,1).*(sin(ttn)).^2);% Calculate Kt from the principle permeability
Mblock=Kt/uo;Mblockr=Kr/uo;% Block Mobility
Mbr=ones(N+1,J); Mbr([1 N+1],:)=Mblockr([1 N],:);% Upscaled mobility in r direction
Mbr(2:N,:)=log(Ro(2:N,:)./Ro(1:N-1,:))./((1./Mblockr(1:N-1,:).*log(Rb(2:N,:)./Ro(1:N-1,:)))+(1./Mblockr(2:N,:).*log(Ro(2:N,:)./Rb(2:N,:))));
Mbt=ones(N,J);% Upscaled mobility in the angular direction
Mbt(:,1:J-1)=2.*Mblock(:,1:J-1).*Mblock(:,2:J)./(Mblock(:,1:J-1)+Mblock(:,2:J));
Mbt(:,J)=2.*Mblock(:,1).*Mblock(:,J)./(Mblock(:,1)+Mblock(:,J));% Last column mobility in the angular direction
Tr1=Rb(1,:).*Mbr(1,:)./(Ro(1,:).*(Rb(2,:)-Rb(1,:)).*(Ro(1,:)-Rb(1,:)));
Trb=Rb(2:N,:).*Mbr(2:N,:)./(Ro(2:N,:).*(Rb(3:N+1,:)-Rb(2:N,:)).*(Ro(2:N,:)-Ro(1:N-1,:))); %Transmisibility in the radial direction
% Calculate the transmisibility coeffiecints (a,b,c,d,e)in different directions
e(1,:)=Rb(2,:).*Mbr(2,:)./(Ro(1,:).*(Rb(2,:)-Rw).*(Ro(2,:)-Ro(1,:)));
e(2:N-1,:)=Rb(3:N,:).*Mbr(3:N,:)./(Ro(2:N-1,:).*(Rb(3:N,:)-Rb(2:N-1,:)).*(Ro(3:N,:)-Ro(2:N-1,:)));
e(N,:)=Rb(N+1,:).*Mbr(1+N,:)./(Ro(N,:).*(Rb(N+1,:)-Rb(N,:)).*(Rb(N+1,:)-Ro(N,:)));
```

```
T=zeros(N,J-1);t=zeros(N,1);
b=Mbt./((Ro.^2).*(Dn^2));
bo=[t b(:,1:J-1)]; oob=[b(:,J) T];
c=[Mbt(:,J) Mbt(:,1:J-1)]./((Ro.^2).*(Dn^2));
coo=[T c(:,1)];oc=[c(:,2:J) t];
% Define the casing permeability to 0
d=[Tr1;Trb];d(1,7:150)=0;
Tre=[zeros(1,J); e(1:N-1,:)];
a=-b-c-d-e;
x1=reshape([Trb; zeros(1,J)]',M,1);x2=reshape(Tre',M,1);
y1=reshape(oc',M,1);y2=reshape(bo',M,1);
y10=reshape(oob',M,1);y20=reshape(coo',M,1);
AA=reshape(a',M,1);
DiagVecs=[x1,y10,y1,AA,y2,y20,x2];
DiagIndx =[-J,-J+1,-1,0,1,J-1,J];
A=spdiags(DiagVecs,DiagIndx,M,M);% Coefficient matrix A for the pressure calculation
% Assign A inside the perforation 1 to ensure the node pressure inside the
% perforation equals to the wellbore pressure.
A(1:6,:)=0;
for i=1:6
    A(i,i)=1;
end
A(J+1:J+6,:)=0;
for i=1:6
```

```
        A(J+i,J+i)=1;
end
A(2*J+1:2*J+6,:)=0;
for i=1:6
        A(2*J+i,2*J+i)=1;
end
A(3*J+1:3*J+6,:)=0;
for i=1:6
        A(3*J+i,3*J+i)=1;
end
A(4*J+2:4*J+5,:)=0;
for i=2:5
        A(4*J+i,4*J+i)=1;
end
A(5*J+2:5*J+5,:)=0;
for i=2:5
        A(5*J+i,5*J+i)=1;
end
A(6*J+2:6*J+5,:)=0;
for i=2:5
        A(6*J+i,6*J+i)=1;
end
A(7*J+2:7*J+5,:)=0;
for i=2:5
        A(7*J+i,7*J+i)=1;
end
```

```
A(8*J+3:8*J+4,:)=0;
for i=3:4
    A(8*J+i,8*J+i)=1;
end
A(9*J+3:9*J+4,:)=0;
for i=3:4
    A(9*J+i,9*J+i)=1;
end
A(10*J+3:10*J+4,:)=0;
for i=3:4
    A(10*J+i,10*J+i)=1;
end
A(11*J+3:11*J+4,:)=0;
for i=3:4
    A(11*J+i,11*J+i)=1;
end
for i=1:J
    A(M-i+1,:)=0;
    A(M-i+1,M-i+1)=1;
end
% Define the boundary condition vector D
D=zeros(M,1);
% Assign node pressure inside the peforation to wellbore pressure:Pw
D(1:6)=Pw;D(J+1:J+6)=Pw;D(2*J+1:2*J+6)=Pw;D(3*J+1:3*J+6)=Pw;
```

```
D(4*J+2:4*J+5)=Pw;D(5*J+2:5*J+5)=Pw;D(6*J+2:6*J+5)=Pw;D(7*J+2:7*J+5)=Pw;
D(8*J+3:8*J+4)=Pw;D(9*J+3:9*J+4)=Pw;D(10*J+3:10*J+4)=Pw;
D(11*J+3:11*J+4)=Pw;
D(M-J+1:M)=Pe;
% Calculate the pressure solution
u = A\D;
p=reshape(u,J,N);P=p';
% Find the permeability from pressure node to  PO
RDo=log(Ro/Rw);RDb=log(Rb/Rw);d1=RDo(2:N,:)-RDb(2:N,:);
Kr2=Kr(2:N,:);Kr1=[Kr(2:N,J) Kr(2:N,1:J-1)];
Kr4=[Kr(1:N-1,J) Kr(1:N-1,1:J-1)];Kr3=Kr(1:N-1,:);
Kt2=Kt(2:N,:);Kt1=[Kt(2:N,J) Kt(2:N,1:J-1)];
Kt4=[Kt(1:N-1,J) Kt(1:N-1,1:J-1)];Kt3=Kt(1:N-1,:);
% Calculate pressure for half logarithmic point in the radial direction
Ptij=(Kr(2:N,:).*P(2:N,:)+Kr(1:N-1,:).*P(1:N-1,:))./(Kr(2:N,:)+Kr(1:N-1,:));
% Calculate pressure for half distance point in the angular direction
KRP=P.*Kt;
Prij=(KRP+[KRP(:,J) KRP(:,1:J-1)])./(Kt+[Kt(:,J) Kt(:,1:J-1)]);
% Determination of the corner pressure
P2=P(2:N,:);P1=[P(2:N,J) P(2:N,1:J-1)];P4=[P(1:N-1,J) P(1:N-1,1:J-1)];P3=P(1:N-1,:);
```

```
P12=Prij(2:N,:);P14=[Ptij(:,J) Ptij(:,1:J-1)];P34=Prij(1:N-1,:);P23=Ptij;
for i=1:N-1
    for j=1:J
        PP=[P1(i,j);P2(i,j);P3(i,j);P4(i,j);P12(i,j);P12(i,j);...
            P23(i,j);P23(i,j);P34(i,j);P34(i,j);P14(i,j);P14(i,j);0]; %P=pressure
        mm=[Tn(1)*d1(1,1) -Tn(1) -d1(1,1) 0 0 0 0 0 0 0 0 0 1 ;...
            0 0 0 -Tn(1)*d1(1,1) Tn(1) -d1(1,1) 0 0 0 0 0 0 1; ...
            0 0 0 0 0 0 Tn(1)*d1(1,1) Tn(1) d1(1,1) 0 0 0 1;...
            0 0 0 0 0 0 0 0 0 -Tn(1)*d1(1,1) -Tn(1) d1(1,1) 1;...
            0 0 -d1(1,1) 0 0 0 0 0 0 0 0 0 1;...
            0 0 0 0 0 -d1(1,1) 0 0 0 0 0 0 1;...
            0 0 0 0 Tn(1) 0 0 0 0 0 0 0 1;...
            0 0 0 0 0 0 0 Tn(1) 0 0 0 0 1;...
            0 0 0 0 0 0 0 0 d1(1,1) 0 0 0 1;...
            0 0 0 0 0 0 0 0 0 0 0 d1(1,1) 1;...
            0 -Tn(1) 0 0 0 0 0 0 0 0 0 0 1;...
            0 0 0 0 0 0 0 0 0 0 -Tn(1) 0 1;...
            -Kr1(i,j)*Tn(1)^2/2+Kt1(i,j)*(-d1(1,1))^2/2 Kt1(i,j)*(-d1(1,1)) Kr1(i,j)*Tn(1) ...
```

```
                Kr2(i,j)*Tn(1)^2/2-Kt2(i,j)*(-d1(1,1))^2/2 -Kt2(i,j)*(-d1(1,1)) Kr2(i,j)*Tn(1) ...
                -Kr3(i,j)*Tn(1)^2/2+Kt3(i,j)*(-d1(1,1))^2/2 Kt3(i,j)*(-d1(1,1)) -Kr3(i,j)*Tn(1) ...
                Kr4(i,j)*Tn(1)^2/2-Kt4(i,j)*(-d1(1,1))^2/2 -Kt4(i,j)*(-d1(1,1)) -Kr4(i,j)*Tn(1) 0];
            Vec=mm\PP;
            po(i,j)=Vec(13);
    end
    for j=J
    end
end
% Rearrange the corner pressure to the corner point coordinates
TO=linspace(0,2*pi-2*pi/(J),J);
tn=[TO 2*pi];
po=[Prij(1,:); po; Prij(N,:)];
po=[po po(:,1)];
LRD=log(rb./Rw);
LRD(N+1)=LRD(N+1)+LRD(1);LRD(1)=0;
RD=exp(LRD);
T=ones(N+10,J);
% Calculate the coeffiecients for the log-lin pressure assumption
for i=1:N
    for j=1:J
```

```
        PP=[po(i,j); po(i,j+1); po(i+1,j+1); po(i+1,j);];
        mm=[tn(j)*LRD(i) tn(j) LRD(i) 1 ;...
            tn(j+1)*LRD(i) tn(j+1) LRD(i) 1; ...
            tn(j+1)*LRD(i+1) tn(j+1) LRD(i+1) 1;...
            tn(j)*LRD(i+1) tn(j) LRD(i+1) 1;];
        Vec=mm\PP;
        aa(i,j)=Vec(1);bb(i,j)=Vec(2);cc(i,j)=Vec(3);dd(i,j)=Vec(4);
    end
end
% Calculate the flow rate for each grid block (q1)
q1=K_block.*((aa(N,:).*tn(2:151).^2/2+cc(N,:).*tn(2:151))-...
    (aa(N,:).*tn(1:150).^2/2+cc(N,:).*tn(1:150)));
% Calculate the total flow rate
q_per=sum(q1);
% Calculate the TOTAL Skin
S=2*pi*(Pe-Pw)*K_block/q_per-log(Re/Rw);
hold on
for k=2
    for j=1:J;
        w=1;
        % Define the launching point coordinate: RDin, thin
        RDin=Re/Rw;
        thin=tn(j)+tn(2)/2;
        % Determine the grid block coordinates
```

```
        XX=N+1;
        YY=ceil(thin/tn(2));
        % If the theta coordinate is J+1, change into frist grid block
        if YY==J+1;
            YY=1;
        end
        % If the theta coordinate is 0, change into last grid block
        if YY==0
            YY=J;
        end
        % Check if the streamline reaches to the boundary
        while XX>=2
            % Check if alnrD+B equals to 0, if it is use the heterogenous method to
            % tracing the streamline
            if abs(aa(XX-1,YY)*thin+cc(XX-1,YY))>10E-7;
                step=-(log(RDin)-log(RD(XX-1)))/19;
                lrd_r=log(RDin):step:log(RD(XX-1)); % upper and lower limitation for lrd
                t_r=zeros(1,length(lrd_r));
                t_r(1)=thin;  % initial condition
                for i=1:(length(lrd_r)-1)% calculation loop to find the potential exit point
```

```
                F_tr=@(lrd,t) Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY))/...
                    (Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY)));
                k_1=F_tr(lrd_r(i),t_r(i));
                k_2=F_tr(lrd_r(i)+0.5*step,t_r(i)+0.5*step*k_1);
                k_3=F_tr((lrd_r(i)+0.5*step),(t_r(i)+0.5*step*k_2));
                k_4=F_tr((lrd_r(i)+step),(t_r(i)+k_3*step));
                t_r(i+1)=t_r(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step; % main equation
            end
            Rd=exp(lrd_r);
            % Determine if the solution is possible, if possible
            % calculate TOF
            if any(t_r<0)
                t_r=t_r+2*pi;
                if any(t_r<tn(J))||any(t_r>tn(J+1))
                    lrd_r=[];t_r=[];
                end
            end
            if any(t_r>2*pi)
                t_r=t_r-2*pi;
```

```
                if any(t_r<tn(1))||any(t_r>tn(2))
                    lrd_r=[];t_r=[];
                end
            end
            if all(t_r<2*pi)&&all(t_r>=0)
                if any(t_r<tn(YY))||any(t_r>tn(YY+1))
                    lrd_r=[];t_r=[];
                end
            end
            if isempty(lrd_r)==0
                dt_r=((RDin-RD(XX-1))*Rw/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)...
                    ./(Kr(XX-1,YY)*(aa(XX-1,YY)*(t_r(1:19)+t_r(2:20))/2+cc(XX-1,YY)));
                T_r=sum(dt_r);
            end
        else
            lrd_r=[];t_r=[];
        end
        % Use theta as the parameteriztion to trace the streamline
        if abs(aa(XX-1,YY)*log(RDin)+bb(XX-1,YY))>1E-7;
            if abs(thin-tn(YY))>1E-7;
                step=-(thin-tn(YY))/19;% out at tn(YY)
```

```
        t_t1=thin:step:tn(YY);% upper and lower limitation for lrd
        lrd_t1=zeros(1,length(t_t1));
        lrd_t1(1)=log(RDin);% initial condition
        for i=1:(length(t_t1)-1)% calculation loop
            F_tr=@(t,lrd) Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY))/...
                (Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY)));
            k_1=F_tr(t_t1(i),lrd_t1(i));
            k_2=F_tr(t_t1(i)+0.5*step,lrd_t1(i)+0.5*step*k_1);
            k_3=F_tr((t_t1(i)+0.5*step),(lrd_t1(i)+0.5*step*k_2));
            k_4=F_tr((t_t1(i)+step),(lrd_t1(i)+k_3*step));
            lrd_t1(i+1)=lrd_t1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
        end
        Rd=exp(lrd_t1);
        % Check if Rd out of grid block, if out, ignore this streamline
        % if not out calculate the TOF of this streamline
```

```
                    if any(Rd<RD(XX-1)-1E-7)||any(Rd>RD(XX)+1E-7)
                        lrd_t1=[];t_t1=[];z_t1=[];
                    else
                        dt_t1=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)...
                            ./(Kt(XX-1,YY)*(aa(XX-1,YY)*(lrd_t1(1:19)+lrd_t1(2:20))/2+bb(XX-1,YY)));
                        T_t1=sum(dt_t1);
                    end
                else
                    lrd_t1=[];t_t1=[];
                end
                if abs(tn(YY+1)-thin)>1E-7;% out at tn(YY+1)
                    step=(tn(YY+1)-thin)/19;
                    t_t2=thin:step:tn(YY+1); % upper and lower limitation for theta
                else
                    t_t2=ones(1,20)*thin;
                end
                lrd_t2=zeros(1,length(t_t2));
                lrd_t2(1)=log(RDin);% initial condition
                for i=1:(length(t_t2)-1)% calculation loop
```

```
            F_tr=@(t,lrd) Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY))/...
                (Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY)));
            k_1=F_tr(t_t2(i),lrd_t2(i));
            k_2=F_tr(t_t2(i)+0.5*step,lrd_t2(i)+0.5*step*k_1);
            k_3=F_tr((t_t2(i)+0.5*step),(lrd_t2(i)+0.5*step*k_2));
            k_4=F_tr((t_t2(i)+step),(lrd_t2(i)+k_3*step));
            lrd_t2(i+1)=lrd_t2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
        end
        Rd=exp(lrd_t2);
        % Check if Rd out of grid block,if out,ignore this streamline
        % if not out calculate the TOF of this streamline
        if any(Rd<RD(XX-1)-1E-7)||any(Rd>RD(XX)+1E-7)
            lrd_t2=[];t_t2=[];
        else
            dt_t2=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)...
```

```
                    ./(Kt(XX-1,YY)*(aa(XX-1,YY)*(lrd_t2(1:19)+lrd_t2(2:20))/2+bb(XX-1,YY)));
                T_t2=sum(dt_t2);
            end
            if abs(tn(YY+1)-thin)<=1E-7;
                lrd_t2=[];t_t2=[];
            end
            if isempty(lrd_t1)==1&& isempty(lrd_t2)==1 % both empty
                lrd_t=[];t_t=[];
            end
            if isempty(lrd_t1)==1&& isempty(lrd_t2)==0 % t1 empty
                lrd_t=lrd_t2;t_t=t_t2;T_t=T_t2;
            end
            if isempty(lrd_t2)==1&& isempty(lrd_t1)==0 % t2 empty
                lrd_t=lrd_t1;t_t=t_t1;T_t=T_t1;
            end
            if isempty(lrd_t1)==0&& isempty(lrd_t2)==0 % both not empty
                if T_t1<T_t2
                    lrd_t=lrd_t1;t_t=t_t1;T_t=T_t1;
                else
                    lrd_t=lrd_t2;t_t=t_t2;T_t=T_t2;
```

```
                    end
                end
            else
                lrd_t=[];t_t=[];
            end
            % Find real streamline between r and t
            if isempty(lrd_r)==1&& isempty(lrd_t)==0
                lrd=lrd_t;t=t_t;Tof=T_t;
            end
            if isempty(lrd_r)==0&& isempty(lrd_t)==1
                lrd=lrd_r;t=t_r;Tof=T_r;
            end
            if isempty(lrd_r)==0&& isempty(lrd_t)==0
                if T_t<T_r
                    lrd=lrd_t;t=t_t;Tof=T_t;
                else
                    lrd=lrd_r;t=t_r;Tof=T_r;
                end
            end
            if isempty(lrd_r)==1&& isempty(lrd_t)==1
                lrd=log(RDin):-(log(RDin)-log(RD(XX-1)))↩
                   /19:log(RD(XX-1));t=ones(1,20)*thin;
            end
            Rd=exp(lrd);
            if abs(RDin-Rd(20))<=1E-7
                RDout=Rd(1);tout=t(1);
```

```
        end
        if abs(RDin-Rd(1))<=1E-7
            RDout=Rd(20);tout=t(20);
        end
        % Plot streamline
        x=Rd.*Rw.*cos(t);
        y=Rd.*Rw.*sin(t);
        plot(x,y,'-r')
        % Calculate new grid block coordinates
        if RDout==1;
            XX=1;
        end
        if XX>=2&& abs(RDout-RD(XX-1))<1e-7
            XX_out=XX-1;
        else
            XX_out=XX;
        end
        YY_out=ceil(tout/tn(2));
        if XX_out==XX
            if YY_out==YY
                if tout<thin
                    YY=YY-1;
                end
                if tout>thin
                    YY=YY+1;
                end
```

```
            else
                YY=YY_out;
            end
        end
        if YY==J+1;
            YY=1;
        end
        if YY==0
            YY=J;
        end
        RDin=RDout;
        thin=tout;
        XX=XX_out;
        if thin==2*pi&&YY==1
            thin=0;
        end
        %Stop tracing once streamline researches to the perforation
        if (XX<=2&& thin>=tn(1)&&thin<=tn(7))||(XX<=3&& thin>=tn(1)&&thin<=tn(7))...
                ||(XX<=4&& thin>=tn(1)&&thin<=tn(7))||(XX<=5&& thin>=tn(1)&&thin<=tn(7))...
                ||(XX<=6&& thin>=tn(2)&&thin<=tn(6))||(XX<=7&& thin>=tn(2)&&thin<=tn(6))...
                ||(XX<=8&& thin>=tn(2)&&thin<=tn(6))||(XX<=9&& thin>=tn(2)&&thin<=tn(6))...
```

```
                        ||(XX<=10&& thin>=tn(3)&&thin<=tn(5))||(XX<=11&& thin>=tn(3)&&thin<=tn(5))...
                        ||(XX<=12&& thin>=tn(3)&&thin<=tn(5))||(XX<=13&& thin>=tn(3)&&thin<=tn(5))

                    XX=1;
               end
               % Save TOF value
               T(w,j)=Tof;
               w=w+1;
          end
     end
end
```

E. 4 Three-Dimensional Perforated Well Streamline Simulator

```
clear all
% Define block no. for R, theta and z directions
Nr=50;Nt=20;nz=10;Nz=nz+1;N=Nr*Nt*Nz; % block no. for R and theta direction
kx=1e-13.*ones(Nr,Nt,nz);ky=1e-13.*ones(Nr,Nt,nz);kz=1e-14.*ones(Nr,Nt,nz);% Permeability at X,Y,Z direction 
```

```
for ORIGNAL blocks
% Define wellbore radius=20 m, reservoir radius=0.15 m, reservoir hight 10*Nz
Re=2;Rw=0.15;Dz=0.02;dz=Dz*(0:1:nz);
% Boundary Pressures: wellbore pressure=280*10^5 pa reservoir pressure=300*10^5 pa
Pw=280*10^5; Pe=300*10^5;
zz=10^-4;%Define digital truncation ZERO
% Permeability in X,Y,Z direction for block centered blocks
Kx(:,:,[1 Nz])=kx(:,:,[1 nz]);Kx(:,:,2:nz)=(kx(:,:,1:nz-1)+kx(:,:,2:nz))/2;
Ky(:,:,[1 Nz])=ky(:,:,[1 nz]);Ky(:,:,2:nz)=(ky(:,:,1:nz-1)+ky(:,:,2:nz))/2;
Kz(:,:,[1 Nz])=kz(:,:,[1 nz]);Kz(:,:,2:nz)=2*kz(:,:,1:nz-1).*kz(:,:,2:nz)./(kz(:,:,1:nz-1)+kz(:,:,2:nz));% Permeability at X,Y,Z direction used in point distributed
% Define fluid viscosity cp
uo=0.8e-3;
R=0:1:Nr-1;
ro=Rw*(Re/Rw).^(R./(Nr-1));% Calculate node radii
rb=ones(1,Nr+1);
rb(1,2:Nr)=(ro(:,2:Nr).*ro(:,1:Nr-1)).^0.5;% Calculate boundary radii
rb(1,[1 Nr+1])=[Rw^2/rb(1,2) Re^2/rb(1,Nr)];
```

```
Ro=repmat(ro',[1,Nt,Nz]); Rb=repmat(rb',[1,Nt,Nz]); % Ro, Rb Radius for nodes for all grid blocks
Tn=linspace(2*pi/(2*Nt),2*pi-2*pi/(2*Nt),Nt); % Theta Nodes
ttn=repmat(Tn',1,Nr)';ttn=repmat(ttn,[1,1,Nz]);
Dn=360/Nt*pi/180; %Delat theta between nodes
Kr=Kx.*(cos(ttn)).^2+Ky.*(sin(ttn)).^2; % Permeability in r direction
Kt=Ky.*(cos(ttn)).^2+Kx.*(sin(ttn)).^2; % Permeability in tangent direction
Mblock=Kt./uo; Mblockr=Kr./uo; Mblockz=Kz./uo;%Block Mobility for r, tangent and Z direction
Mbr=ones(Nr+1,Nt,Nz);Mbz=zeros(Nr,Nt,Nz+1);%Upscaled mobility at r, z direction
Mbr([1 Nr+1],:,:)=Mblockr([1 Nr],:,:);% Upscaled mobility at r direction
Mbr(2:Nr,:,:)=log(Ro(2:Nr,:,:)./Ro(1:Nr-1,:,:))./((1./Mblockr(1:Nr-1,:,:).*log(Rb(2:Nr,:,:)./Ro(1:Nr-1,:,:)))+(1./Mblockr(2:Nr,:,:).*log(Ro(2:Nr,:,:)./Rb(2:Nr,:,:))));
Mbt=ones(Nr,Nt,Nz);% Upscaled mobility at theta direction
Mbt(:,1:Nt-1,:)=2.*Mblock(:,1:Nt-1,:).*Mblock(:,2:Nt,:)./(Mblock(:,1:Nt-1,:)+Mblock(:,2:Nt,:));
Mbt(:,Nt,:)=2.*Mblock(:,1,:).*Mblock(:,Nt,:)./(Mblock(:,1,:)+Mblock(:,Nt,:));% Last column is Mobility from the last to 1
Trb=zeros(Nr,Nt,Nz);
```

```
Mbz(:,:,2:Nz)=2.*Mblockz(:,:,1:Nz-1).*Mblockz(:,:,2:Nz)./(Mblockz(:,:,1:Nz-1)+Mblockz(:,:,2:Nz));
Mbr(:,:,[1 Nz])=0.5*Mbr(:,:,[1 Nz]);Mbt(:,:,[1 Nz])=0.5*Mbt(:,:,[1 Nz]);
% Calculate the transmisibility coeffiecints a,b,c,d,e,g,g
Tr1=Rb(1,:,:).*Mbr(1,:,:)./(Ro(1,:,:).*(Rb(2,:,:)-Rb(1,:,:)).*(Ro(1,:,:)-Rb(1,:,:))); %first layer upscaled mobility at r direction is the block mobility
Trb(1:Nr-1,:,:)=Rb(2:Nr,:,:).*Mbr(2:Nr,:,:)./(Ro(2:Nr,:,:).*(Rb(3:Nr+1,:,:)-Rb(2:Nr,:,:)).*(Ro(2:Nr,:,:)-Ro(1:Nr-1,:,:)));
e([1 Nr],:,:)=Rb([2 Nr+1],:,:).*Mbr([2 Nr+1],:,:)./(Ro([1 Nr],:,:).*(Rb([2 Nr+1],:,:)-Rb([1 Nr],:,:)).*([Ro(2,:,:);Rb(Nr+1,:,:)]-Ro([1 Nr],:,:)));
e(2:Nr-1,:,:)=Rb(3:Nr,:,:).*Mbr(3:Nr,:,:)./(Ro(2:Nr-1,:,:).*(Rb(3:Nr,:,:)-Rb(2:Nr-1,:,:)).*(Ro(3:Nr,:,:)-Ro(2:Nr-1,:,:)));
oob=zeros(Nr,Nt,Nz);bo=zeros(Nr,Nt,Nz);oc=zeros(Nr,Nt,Nz);coo=zeros(Nr,Nt,Nz);
b=Mbt./((Ro.^2).*(Dn^2));bo(:,2:Nt,:)=b(:,1:Nt-1,:); oob(:,1,:)=b(:,Nt,:);
c=[Mbt(:,Nt,:) Mbt(:,1:Nt-1,:)]./((Ro.^2).*(Dn^2));coo(:,Nt,:)=c(:,1,:);oc(:,1:Nt-1,:)=c(:,2:Nt,:);
d=[Tr1;Trb(1:Nr-1,:,:)];Tre=zeros(Nr,Nt,Nz);Tre(2:Nr,:,:)=e(1:Nr-1,:,:);
f=Mbz(:,:,2:Nz+1)./((Dz^2));g=Mbz(:,:,1:Nz)./((Dz^2));
```

```
d(1,:,1)=0;d(1,2:20,2)=0;d(1,:,3:9)=0;d(1,1:10,10)=0;d(1,12:20,10)=0;
d(1,:,11)=0;
a=b+c+d+e+f+g;
x1=reshape(permute(Trb,[2,1,3]),N,1);x2=reshape(permute(Tre,[2,1,3]),N,1);
y1=reshape(permute(oc,[2,1,3]),N,1);y2=reshape(permute(bo,[2,1,3]),N,1);
y10=reshape(permute(oob,[2,1,3]),N,1);y20=reshape(permute(coo,[2,1,3]),N,1);
z1=reshape(permute(g,[2,1,3]),N,1);z2=reshape(permute(f,[2,1,3]),N,1);
AA=reshape(permute(a,[2,1,3]),N,1);
DiagVecs=[-z2,-x1,-y10,-y1,AA,-y2,-y20,-x2,-z1];
DiagIndx =[-Nr*Nt,-Nt,-Nt+1,-1,0,1,Nt-1,Nt,Nr*Nt];
% Coefficient matrix A for the pressure
A = spdiags(DiagVecs,DiagIndx,N,N);
% Define the boundary condition ensure pressure for grid block inside perforation equals to Pw
A(Nt*Nr+1,:)=0;
A(9*Nt*Nr+11,:)=0;
for i=1
A(Nt*Nr+i,Nt*Nr+i)=1;
A(9*Nt*Nr+10+i,9*Nt*Nr+10+i)=1;
end
% Define the boundary condition
```

```
D=zeros(N,1);
D(Nt*Nr+1)=Pw;
D(9*Nt*Nr+11)=Pw;
for i=1:Nz
D(i*Nr*Nt-Nt+1:i*Nr*Nt)=Pe;
A((i-1)*Nr*Nt+(Nr-1)*Nt+1:(i-1)*Nr*Nt+Nr*Nt,:)=0;
end
for i=1:Nz
    for k=1:Nt
        A((i-1)*Nr*Nt+(Nr-1)*Nt+k,(i-1)*Nr*Nt+(Nr-1)*Nt+k)↩
            =1;
    end
end
t=linspace(-pi,pi,100);
% Calculate the pressure solution
u = A\D;
for i=1
plot3(Re*cos(t),Re*sin(t),(i-1)*Dz*ones(1,100))
hold on
plot3(Rw*cos(t),Rw*sin(t),(i-1)*Dz*ones(1,100))
hold on
end
p=permute(reshape(u,Nt,Nr,Nz),[2,1,3]);
% Calculate pressure at R. theta and z boundaries
Rb1=Ro(2:Nr,:,:);Rb2=Ro(1:Nr-1,:,:);RB=Rb(2:Nr,:,:);RD1=Rb1↩
    ./RB;RD2=Rb2./RB;
```

```
Kr2=Kr(2:Nr,:,:);Kr1=[Kr(2:Nr,Nt,:) Kr(2:Nr,1:Nt-1,:)];
Kr4=[Kr(1:Nr-1,Nt,:) Kr(1:Nr-1,1:Nt-1,:)];Kr3=Kr(1:Nr-1,:,:);
Kt2=Kt(2:Nr,:,:);Kt1=[Kt(2:Nr,Nt,:) Kt(2:Nr,1:Nt-1,:)];
Kt4=[Kt(1:Nr-1,Nt,:) Kt(1:Nr-1,1:Nt-1,:)];Kt3=Kt(1:Nr-1,:,:);
d1=log(RD1);d2=log(RD2);
% Calculate pressure at R boundaries
Ptij=(-d2.*Kr(2:Nr,:,:).*p(2:Nr,:,:)+d1.*Kr(1:Nr-1,:,:).*p(1:Nr-1,:,:))./(-d2.*Kr(2:Nr,:,:)+d1.*Kr(1:Nr-1,:,:));
KTP=p.*Kt; Prij=(KTP+[KTP(:,Nt,:) KTP(:,1:Nt-1,:)])./(Kt+[Kt(:,Nt,:) Kt(:,1:Nt-1,:)]);% p12,p34
P2=p(2:Nr,:,:);P1=[p(2:Nr,Nt,:) p(2:Nr,1:Nt-1,:)];P4=[p(1:Nr-1,Nt,:) p(1:Nr-1,1:Nt-1,:)];P3=p(1:Nr-1,:,:);
P12=Prij(2:Nr,:,:);P14=[Ptij(:,Nt,:) Ptij(:,1:Nt-1,:)];P34=Prij(1:Nr-1,:,:);P23=Ptij;
% Calculate corner pressures
for k=1:Nz
  for i=1:Nr-1
     for j=1:Nt
            PP=[P1(i,j,k);P2(i,j,k);P3(i,j,k);P4(i,j,k);P12(i,j,k);P12(i,j,k);...
                P23(i,j,k);P23(i,j,k);P34(i,j,k);P34(i,j,k);P14(i,j,k);P14(i,j,k);0]; %P=pressure
            mm=[Tn(1)*d1(1,1,1) -Tn(1) -d1(1,1,1) 0 0 0 0 0 0 0 0 0 1 ;...
```

```
0 0 0 -Tn(1)*d1(1,1,1) Tn(1) -d1(1,1,1) 0 0 0 0 0 0 1; ...
0 0 0 0 0 0 Tn(1)*d1(1,1,1) Tn(1) d1(1,1,1) 0 0 0 1;...
0 0 0 0 0 0 0 0 0 -Tn(1)*d1(1,1,1) -Tn(1) d1(1,1,1) 1;...
0 0 -d1(1,1,1) 0 0 0 0 0 0 0 0 0 1;...
0 0 0 0 0 -d1(1,1,1) 0 0 0 0 0 0 1;...
0 0 0 0 Tn(1) 0 0 0 0 0 0 0 1;...
0 0 0 0 0 0 0 Tn(1) 0 0 0 0 1;...
0 0 0 0 0 0 0 0 d1(1,1,1) 0 0 0 1;...
0 0 0 0 0 0 0 0 0 0 0 d1(1,1,1) 1;...
0 -Tn(1) 0 0 0 0 0 0 0 0 0 0 1;...
0 0 0 0 0 0 0 0 0 0 -Tn(1) 0 1;...
-Kr1(i,j)*Tn(1)^2/2+Kt1(i,j)*(-d1(1,1,1))^2/2 Kt1(i,j)*(-d1(1,1,1)) Kr1(i,j)*Tn(1) ...
Kr2(i,j)*Tn(1)^2/2-Kt2(i,j)*(-d1(1,1,1))^2/2 -Kt2(i,j)*(-d1(1,1,1)) Kr2(i,j)*Tn(1) ...
-Kr3(i,j)*Tn(1)^2/2+Kt3(i,j)*(-d1(1,1,1))^2/2 Kt3(i,j)*(-d1(1,1,1)) -Kr3(i,j)*Tn(1) ...
Kr4(i,j)*Tn(1)^2/2-Kt4(i,j)*(-d1(1,1,1))^2/2 -Kt4(i,j)*(-d1(1,1,1)) -Kr4(i,j)*Tn(1) 0];
```

```
                Vec=mm\PP;
                po(i,j,k)=Vec(13);
        end
    end
end
PO=[Prij(1,:,:);po;Prij(Nr,:,:)];
PO(1,1:2,2:3)=Pw;
PO=[PO PO(:,1,:)];
Roo=Rb./Rw;LRD=log(Roo(:,1)');LRD(Nr+1)=LRD(Nr+1)+LRD(1);LRD(1)=0;
RD=exp(LRD);
for i=1:Nz
PRZ(:,i)=PO(:,1,i);
end
xro=repmat((RD*Rw)',1,Nz);
yz=0:Dz:nz*Dz;
yz=repmat(yz',1,Nr+1);yz=yz';
tn=linspace(0,2*pi,Nt+1);
aa=ones(Nr,Nt,Nz-1);bb=ones(Nr,Nt,Nz-1);cc=ones(Nr,Nt,Nz-1);dd=ones(Nr,Nt,Nz-1);
ee=ones(Nr,Nt,Nz-1);ff=ones(Nr,Nt,Nz-1);gg=ones(Nr,Nt,Nz-1);hh=ones(Nr,Nt,Nz-1);
kr=kx.*(cos(ttn(:,:,1:nz))).^2+ky.*(sin(ttn(:,:,1:nz))).^2; % Permeability in r direction
kt=ky.*(cos(ttn(:,:,1:nz))).^2+kx.*(sin(ttn(:,:,1:nz))).^2; % Permeability in tangent direction
```

```
% Calculate the coeffiecients for the trilin-log pressure assumpution
for i=1:Nr
    for j=1:Nt
        for k=1:Nz-1
            P=[PO(i,j,k); PO(i,j+1,k); PO(i+1,j+1,k); PO(i+1,j,k);...
                PO(i,j,k+1); PO(i,j+1,k+1); PO(i+1,j+1,k+1); PO(i+1,j,k+1)]; %P=pressure
            A=[tn(j)*LRD(i)*Dz*(k-1) tn(j)*LRD(i) tn(j)*Dz*(k-1) LRD(i)*Dz*(k-1) tn(j) LRD(i) Dz*(k-1) 1;...
               tn(j+1)*LRD(i)*Dz*(k-1) tn(j+1)*LRD(i) tn(j+1)*Dz*(k-1) LRD(i)*Dz*(k-1) tn(j+1) LRD(i) Dz*(k-1) 1; ...
               tn(j+1)*LRD(i+1)*Dz*(k-1) tn(j+1)*LRD(i+1) tn(j+1)*Dz*(k-1) LRD(i+1)*Dz*(k-1) tn(j+1) LRD(i+1) Dz*(k-1) 1;...
               tn(j)*LRD(i+1)*Dz*(k-1) tn(j)*LRD(i+1) tn(j)*Dz*(k-1) LRD(i+1)*Dz*(k-1) tn(j) LRD(i+1) Dz*(k-1) 1;...
               tn(j)*LRD(i)*Dz*k tn(j)*LRD(i) tn(j)*Dz*k LRD(i)*Dz*k tn(j) LRD(i) Dz*k 1;...
               tn(j+1)*LRD(i)*Dz*k tn(j+1)*LRD(i) tn(j+1)*Dz*k LRD(i)*Dz*k tn(j+1) LRD(i) Dz*k 1; ...
```

```
                    tn(j+1)*LRD(i+1)*Dz*k tn(j+1)*LRD(i+1) tn(j+1)*Dz*k LRD(i+1)*Dz*k tn(j+1) LRD(i+1) Dz*k 1;...
                    tn(j)*LRD(i+1)*Dz*k tn(j)*LRD(i+1) tn(j)*Dz*k LRD(i+1)*Dz*k tn(j) LRD(i+1) Dz*k 1;];
                    Vec=A\P;
                    aa(i,j,k)=Vec(1);bb(i,j,k)=Vec(2);cc(i,j,k)=Vec(3);dd(i,j,k)=Vec(4);
                    ee(i,j,k)=Vec(5);ff(i,j,k)=Vec(6);gg(i,j,k)=Vec(7);hh(i,j,k)=Vec(8);
          end
     end
end
TTO=repmat(Tn',1,nz);
TTB=repmat(tn',1,nz);
dzz=0.5*Dz:Dz:nz*Dz;
Z=repmat(dzz',1,Nt)';
ZB=repmat(dz',1,Nt)';
q=reshape(Kr(Nr-1,1:Nt,1:nz),Nt,nz)/uo.*(reshape(aa(Nr-1,:,:),Nt,nz).*(TTB(2:Nt+1,:).^2-TTB(1:Nt,:).^2).* ...
(ZB(:,2:nz+1).^2-ZB(:,1:nz).^2)./4+reshape(bb(Nr-1,:,:),Nt,nz).*(TTB(2:Nt+1,:).^2-TTB(1:Nt,:).^2) ...
 .*(ZB(:,2:nz+1)-ZB(:,1:nz))./2+reshape(dd(Nr-1,:,:),Nt,nz).*(ZB(:,2:nz+1).^2-ZB(:,1:nz).^2).* ...
 (TTB(2:Nt+1,:)-TTB(1:Nt,:))./2+reshape(ff(Nr-1,:,:),Nt,nz).* ...
```

```
 (ZB(:,2:nz+1)-ZB(:,1:nz)).*(TTB(2:Nt+1,:)-TTB(1:Nt,:)));
q_per=sum(q);q_per=sum(q_per);
% Calculate total skin
S=2*pi*Kr(Nr,1,1)*(Pe-Pw)*nz*Dz/q_per/uo-log(Re/Rw);
[x,y,z]=cylinder(Rw,Nt);
surf(x,y,nz*Dz*z);
t=[];
for m=4
for k=5
for j=1:10
  % Define the launching point coordinate: RDin, thin,zin
  RDin=Re/Rw;
  thin=tn(j)+(k-1)*tn(2)/5+tn(2)/10;
  zin=(m-1)*Dz+0.5*Dz;
  % Determine the grid block coordinates
  XX=Nr+1;
  YY=ceil(thin/tn(2));
  ZZ=ceil(zin/Dz);
  if YY==Nt+1;
      YY=1;
  end
  if YY==0
      YY=Nt;
  end
% Check if the streamline reaches to the boundary
while XX>1&&ZZ>0
```

```
    % Use r as parameterization to trace the streamline
    if abs(aa(XX-1,YY,ZZ)*thin*zin+bb(XX-1,YY,ZZ)*thin+dd(XX-1,YY,ZZ)*zin+ff(XX-1,YY,ZZ))>zz;
        step=-(RDin*Rw-RD(XX-1)*Rw)/19;
        r_r=RDin*Rw:step:RD(XX-1)*Rw;% upper and lower limitation for lrd
        t_r=zeros(1,length(r_r));
        z_r=zeros(1,length(r_r));
        t_r(1)=thin; z_r(1)=zin;% initial condition
        for i=1:(length(r_r)-1)% calculation loop
            F_tr=@(lrd,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_r(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_r(i)+ee(XX-1,YY,ZZ))/...
              (kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_r(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_r(i)+ff(XX-1,YY,ZZ)));    % change the function as you desire
            F_zr=@(lrd,z) kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_r(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_r(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ))/...
              (kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_r(i)+bb(XX-1,YY,ZZ)*t_r(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ)));
            k_1=F_tr(r_r(i),t_r(i));
            k_2=F_tr(r_r(i)+0.5*step,t_r(i)+0.5*step*k_1);
```

```
            k_3=F_tr((r_r(i)+0.5*step),(t_r(i)+0.5*step*k_2));
            k_4=F_tr((r_r(i)+step),(t_r(i)+k_3*step));
            t_r(i+1)=t_r(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
            kk_1=F_zr(r_r(i),z_r(i));
            kk_2=F_zr(r_r(i)+0.5*step,z_r(i)+0.5*step*kk_1);
            kk_3=F_zr((r_r(i)+0.5*step),(z_r(i)+0.5*step*kk_2));
            kk_4=F_zr((r_r(i)+step),(z_r(i)+kk_3*step));
            z_r(i+1)=z_r(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;% main equation
        end
        % Determine if the solution is possible, if possible calculate the
        % TOF for this streamline
        Rd=r_r/Rw;
        if any(t_r-0<=zz)
            t_r=t_r+2*pi;
            if any(t_r<tn(Nt)-zz)||any(t_r>tn(Nt+1)+zz)||any(z_r<dz(ZZ)-1E-4)||any(z_r>dz(ZZ+1)+1E-4)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if any(t_r-2*pi>=zz)
```

```
            t_r=t_r-2*pi;
            if any(t_r<tn(1)-zz)||any(t_r>tn(2)+zz)||any(z_r<dz(ZZ)-1E-4)||any(z_r>dz(ZZ+1)+1E-4)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if all(t_r-2*pi<1e-4)&&all(t_r-0>=1e-4)
            if any(t_r<tn(YY)-1e-4)||any(t_r>tn(YY+1)+1E-4)||any(z_r<dz(ZZ)-1E-4)||any(z_r>dz(ZZ+1)+1E-4)
                r_r=[];t_r=[];z_r=[];
            end
        end
        if isempty(r_r)==0&&isreal(r_r)==1
            x=Rd*Rw.*cos(t_r);
            y=Rd*Rw.*sin(t_r);
            length_r=0;
            for i=1:19
                length_r=length_r+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_r(i+1)-z_r(i))^2)^0.5;
            end
        dt_r=((RDin*Rw-RD(XX-1)*Rw)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kr(XX-1,YY,ZZ)...
        .*(aa(XX-1,YY,ZZ).*((t_r(1:19)+t_r(2:20))/2).*((z_r(1:19)+z_r(2:20))/2)+bb(XX-1,YY,ZZ).*((t_r(1:19)+t_r(2:20))/2)...
```

```
            +dd(XX-1,YY,ZZ).*((z_r(1:19)+z_r(2:20))/2)+ff(XX-1,YY,ZZ)));
        T_r=sum(dt_r);XXR_out=XX-1;YYR_out=YY;ZZR_out=ZZ;
        end
    else
        r_r=[];t_r=[];z_r=[];
    end
    % Use theta as parameterization to trace the streamline
    if abs(aa(XX-1,YY,ZZ)*log(RDin)*zin+bb(XX-1,YY,ZZ)*log(RDin)+cc(XX-1,YY,ZZ)*zin+ee(XX-1,YY,ZZ))>zz;
        if abs(thin-tn(YY))>10e-5;
          step=-(thin-tn(YY))/19; % Exit at tn(YY)
          t_t1=thin:step:tn(YY);  % upper and lower limitation for lrd
          r_t1=zeros(1,length(t_t1));
          z_t1=zeros(1,length(t_t1));
          r_t1(1)=RDin*Rw; z_t1(1)=zin; % initial condition
          for i=1:(length(t_t1)-1) % calculation loop
            F_tr=@(t,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t1(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_t1(i)+ff(XX-1,YY,ZZ))/...
                (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t1(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_t1(i)+ee(XX-1,YY,ZZ)));     % change the function as you desire
```

```
            F_tz=@(t,z) kz(XX-1,YY,ZZ)*r_t1(i)*(aa(XX-1,YY,ZZ)*t*log(r_t1(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_t1(i)/Rw)+gg(XX-1,YY,ZZ))/...
                (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_t1(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_t1(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ)));
            k_1=F_tr(t_t1(i),r_t1(i));
            k_2=F_tr(t_t1(i)+0.5*step,r_t1(i)+0.5*step*k_1);
            k_3=F_tr((t_t1(i)+0.5*step),(r_t1(i)+0.5*step*k_2));
            k_4=F_tr((t_t1(i)+step),(r_t1(i)+k_3*step));
            r_t1(i+1)=r_t1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;  % main equation
            kk_1=F_tz(t_t1(i),z_t1(i));
            kk_2=F_tz(t_t1(i)+0.5*step,z_t1(i)+0.5*step*kk_1);
            kk_3=F_tz((t_t1(i)+0.5*step),(z_t1(i)+0.5*step*kk_2));
            kk_4=F_tz((t_t1(i)+step),(z_t1(i)+kk_3*step));
            z_t1(i+1)=z_t1(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;  % main equation
          end
        Rd=r_t1/Rw;
        %check if Rd or Z out of grid block,if out,ignore this streamline
        %if not out calculate the length of this streamline
```

```
            if any(Rd<RD(XX-1)-1E-4)||any(Rd>RD(XX)+1E-4)||any(z_t1<dz(ZZ)-zz)||any(z_t1>dz(ZZ+1)+zz)||isreal(Rd)==0||isreal(z_t1)==0
                r_t1=[];t_t1=[];z_t1=[];
            else
                x=Rd*Rw.*cos(t_t1);
                y=Rd*Rw.*sin(t_t1);
                length_t1=0;
                for i=1:19
                    length_t1=length_t1+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_t1(i+1)-z_t1(i))^2)^0.5;
                end
                dt_t1=((thin-tn(YY))/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kt(XX-1,YY,ZZ).*(aa(XX-1,YY,ZZ).*...
             (((r_t1(1:19))+(r_t1(2:20)))/2).*((z_t1(1:19)+z_t1(2:20))/2)+bb(XX-1,YY,ZZ).*(((r_t1(1:19))+(r_t1(2:20)))/2)...
             +cc(XX-1,YY,ZZ).*((z_t1(1:19)+z_t1(2:20))/2)+ee(XX-1,YY,ZZ)));
             T_t1=sum(dt_t1);
            end
          else
             r_t1=[];t_t1=[];z_t1=[];
          end
          if abs(tn(YY+1)-thin)>10e-5;
```

```
        step=(tn(YY+1)-thin)/19; % Exit at tn(YY+1)
        t_t2=thin:step:tn(YY+1); % upper and lower limitation for theta
        r_t2=zeros(1,length(t_t2));
        z_t2=zeros(1,length(t_t2));
        r_t2(1)=RDin*Rw; z_t2(1)=zin; % initial condition
        for i=1:(length(t_t2)-1) % calculation loop
          F_tr=@(t,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t2(i)*t+bb(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*z_t2(i)+ff(XX-1,YY,ZZ))/...
              (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z_t2(i)*log(lrd/Rw)+bb(XX-1,YY,ZZ)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*z_t2(i)+ee(XX-1,YY,ZZ))); % change the function as you desire
          F_tz=@(t,z) kz(XX-1,YY,ZZ)*r_t2(i)*(aa(XX-1,YY,ZZ)*t*log(r_t2(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_t2(i)/Rw)+gg(XX-1,YY,ZZ))/...
              (kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_t2(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_t2(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ)));
          k_1=F_tr(t_t2(i),r_t2(i));
          k_2=F_tr(t_t2(i)+0.5*step,r_t2(i)+0.5*step*k_1);
          k_3=F_tr((t_t2(i)+0.5*step),(r_t2(i)+0.5*step*k_2));
          k_4=F_tr((t_t2(i)+step),(r_t2(i)+k_3*step));
```

```
            r_t2(i+1)=r_t2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;  % main equation
            kk_1=F_tz(t_t2(i),z_t2(i));
            kk_2=F_tz(t_t2(i)+0.5*step,z_t2(i)+0.5*step*kk_1);
            kk_3=F_tz((t_t2(i)+0.5*step),(z_t2(i)+0.5*step*kk_2));
            kk_4=F_tz((t_t2(i)+step),(z_t2(i)+kk_3*step));
            z_t2(i+1)=z_t2(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step;  % main equation
        end
        Rd=r_t2/Rw;
        %check if Rd or Z out of grid block,if out,ignore this streamline
        %if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-1E-4)||any(Rd>RD(XX)+1E-4)||any(z_t2<dz(ZZ)-1E-4)||any(z_t2>dz(ZZ+1)+1E-4)||isreal(Rd)==0||isreal(z_t2)==0
            r_t2=[];t_t2=[];z_t2=[];
        else
            x=Rd.*Rw.*cos(t_t2);
            y=Rd.*Rw.*sin(t_t2);
            length_t2=0;
            for i=1:19
```

```
                length_t2=length_t2+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_t2(i+1)-z_t2(i))^2)^0.5;
            end
            dt_t2=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)./(Kt(XX-1,YY,ZZ).*(aa(XX-1,YY,ZZ).*...
        (((r_t2(1:19))+(r_t2(2:20)))/2).*((z_t2(1:19)+z_t2(2:20))/2)+bb(XX-1,YY,ZZ).*(((r_t2(1:19))+(r_t2(2:20)))/2)...
         +cc(XX-1,YY,ZZ).*((z_t2(1:19)+z_t2(2:20))/2)+ee(XX-1,YY,ZZ)));
         T_t2=sum(dt_t2);
        end
    else
        r_t2=[];t_t2=[];z_t2=[];
    end
        if isempty(r_t1)==1&& isempty(r_t2)==1 % both empty
            lrd_t=[];t_t=[];z_t=[];
        end
        if isempty(r_t1)==1&& isempty(r_t2)==0 % t1 empty
            lrd_t=r_t2;t_t=t_t2;z_t=z_t2;T_t=T_t2;XXT_out=XX;YYT_out=YY+1;ZZT_out=ZZ;
        end
        if isempty(r_t2)==1&& isempty(r_t1)==0 % t2 empty
```

```
                lrd_t=r_t1;t_t=t_t1;z_t=z_t1;T_t=T_t1;XXT_out=XX;YYT_out=YY-1;ZZT_out=ZZ;
            end
            if isempty(r_t1)==0&& isempty(r_t2)==0 % both not empty
                if T_t1<T_t2
                    lrd_t=r_t1;t_t=t_t1;z_t=z_t1;T_t=T_t1;XXT_out=XX;YYT_out=YY-1;ZZT_out=ZZ;
                else
                    lrd_t=r_t2;t_t=t_t2;z_t=z_t2;T_t=T_t2;XXT_out=XX;YYT_out=YY+1;ZZT_out=ZZ;
                end
            end
    else
      lrd_t=[];t_t=[];z_t=[];
    end
    % Use z as parameterization to trace streamline
    if abs(aa(XX-1,YY,ZZ)*log(RDin)*thin+cc(XX-1,YY,ZZ)*thin+dd(XX-1,YY,ZZ)*log(RDin)+gg(XX-1,YY,ZZ))>zz;
        if abs(dz(ZZ+1)-zin)>=10e-6
            step=(dz(ZZ+1)-zin)/19; % out from dz(ZZ+1)
            z_z1=zin:step:dz(ZZ+1); % upper and lower limitation for lrd
            t_z1=zeros(1,length(z_z1));
            r_z1=zeros(1,length(z_z1));
            t_z1(1)=thin; r_z1(1)=RDin*Rw; % initial condition
```

```
    for i=1:(length(z_z1)-1) % calculation loop
        F_tz=@(z,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log(r_z1(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_z1(i)/Rw)+cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ))/...
            (kz(XX-1,YY,ZZ)*r_z1(i)*(aa(XX-1,YY,ZZ)*t*log(r_z1(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX-1,YY,ZZ)*log(r_z1(i)/Rw)+gg(XX-1,YY,ZZ)));     % change the function as you desire
        F_zr=@(z,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_z1(i)+bb(XX-1,YY,ZZ)*t_z1(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ))/...
            (kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_z1(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_z1(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ)));
        k_1=F_tz(z_z1(i),t_z1(i));
        k_2=F_tz(z_z1(i)+0.5*step,t_z1(i)+0.5*step*k_1);
        k_3=F_tz((z_z1(i)+0.5*step),(t_z1(i)+0.5*step*k_2));
        k_4=F_tz((z_z1(i)+step),(t_z1(i)+k_3*step));
        t_z1(i+1)=t_z1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step; % main equation
        kk_1=F_zr(z_z1(i),r_z1(i));
```

```
            kk_2=F_zr(z_z1(i)+0.5*step,r_z1(i)+0.5*step*kk_1);
            kk_3=F_zr((z_z1(i)+0.5*step),(r_z1(i)+0.5*step*kk_2));
            kk_4=F_zr((z_z1(i)+step),(r_z1(i)+kk_3*step));
            r_z1(i+1)=r_z1(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step; % main equation
        end
        Rd=r_z1/Rw;
        %check if Rd or Z out of grid block,if out,ignore this streamline
        %if not out calculate the length of this streamline
        if any(Rd<RD(XX-1)-zz)||any(Rd>RD(XX)+zz)||any(t_z1<tn(YY)-zz)||any(t_z1>tn(YY+1)+zz)||isreal(Rd)==0||isreal(t_z1)==0
            r_z1=[];t_z1=[];z_z1=[];
        else
            x=Rd.*Rw.*cos(t_z1);
            y=Rd.*Rw.*sin(t_z1);
            length_z1=0;
            for i=1:19
              length_z1=length_z1+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_z1(i+1)-z_z1(i))^2)^0.5;
            end
```

```
            dt_z1=((dz(ZZ+1)-zin)/19)./(Kz(XX-1,YY,ZZ).*(aa↩
                (XX-1,YY,ZZ).*((t_z1(1:19)+t_z1(2:20))/2)...
        .*(((r_z1(1:19))+(r_z1(2:20)))/2)+cc(XX-1,YY,ZZ)↩
            .*((t_z1(1:19)+t_z1(2:20))/2)...
          +dd(XX-1,YY,ZZ).*(((r_z1(1:19))+(r_z1(2:20)))/2)+↩
            gg(XX-1,YY,ZZ)));
          T_z1=sum(dt_z1);
        end
      else
          r_z1=[];t_z1=[];z_z1=[];
      end
      if abs(zin-dz(ZZ))>=10e-6
        step=-(zin-dz(ZZ))/19; % out from dz(ZZ)
        z_z2=zin:step:dz(ZZ); % upper and lower limitation↩
            for lrd
        t_z2=zeros(1,length(z_z2));
        r_z2=zeros(1,length(z_z2));
        t_z2(1)=thin;r_z2(1)=RDin*Rw; % initial condition
        for i=1:(length(z_z2)-1) % calculation loop
          F_tz=@(z,t) kt(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*log↩
              (r_z2(i)/Rw)+bb(XX-1,YY,ZZ)*log(r_z2(i)/Rw)+↩
              cc(XX-1,YY,ZZ)*z+ee(XX-1,YY,ZZ))/...
                (kz(XX-1,YY,ZZ)*r_z2(i)*(aa(XX-1,YY,ZZ)*t*↩
                    log(r_z2(i)/Rw)+cc(XX-1,YY,ZZ)*t+dd(XX↩
                    -1,YY,ZZ)*log(r_z2(i)/Rw)+gg(XX-1,YY,ZZ)↩
                    ));    % change the function as you ↩
```

```
                desire
            F_zr=@(z,lrd) kr(XX-1,YY,ZZ)*(aa(XX-1,YY,ZZ)*z*t_z2(i)+bb(XX-1,YY,ZZ)*t_z2(i)+dd(XX-1,YY,ZZ)*z+ff(XX-1,YY,ZZ))/...
                (kz(XX-1,YY,ZZ)*lrd*(aa(XX-1,YY,ZZ)*t_z2(i)*log(lrd/Rw)+cc(XX-1,YY,ZZ)*t_z2(i)+dd(XX-1,YY,ZZ)*log(lrd/Rw)+gg(XX-1,YY,ZZ)));
            k_1=F_tz(z_z2(i),t_z2(i));
            k_2=F_tz(z_z2(i)+0.5*step,t_z2(i)+0.5*step*k_1);
            k_3=F_tz((z_z2(i)+0.5*step),(t_z2(i)+0.5*step*k_2));
            k_4=F_tz((z_z2(i)+step),(t_z2(i)+k_3*step));
            t_z2(i+1)=t_z2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step; % main equation
            kk_1=F_zr(z_z2(i),r_z2(i));
            kk_2=F_zr(z_z2(i)+0.5*step,r_z2(i)+0.5*step*kk_1);
            kk_3=F_zr((z_z2(i)+0.5*step),(r_z2(i)+0.5*step*kk_2));
            kk_4=F_zr((z_z2(i)+step),(r_z2(i)+kk_3*step));
            r_z2(i+1)=r_z2(i)+(1/6)*(kk_1+2*kk_2+2*kk_3+kk_4)*step; % main equation
        end
         Rd=r_z2/Rw;
```

```
            %check if Rd or Z out of grid block,if out,ignore this streamline
            %if not out calculate the length of this streamline
            if any(Rd<RD(XX-1)-zz)||any(Rd>RD(XX)+zz)||any(t_z2<tn(YY)-zz)||any(t_z2>tn(YY+1)+zz)||isreal(Rd)==0||isreal(t_z2)==0
                r_z2=[];t_z2=[];z_z2=[];
            else
                x=Rd.*Rw.*cos(t_z2);
                y=Rd.*Rw.*sin(t_z2);
                length_z2=0;
                for i=1:19
                    length_z2=length_z2+((x(i+1)-x(i))^2+(y(i+1)-y(i))^2+(z_z2(i+1)-z_z2(i))^2)^0.5;
                end
                dt_z2=((zin-dz(ZZ))/19)./(Kz(XX-1,YY,ZZ).*(aa(XX-1,YY,ZZ).*((t_z2(1:19)+t_z2(2:20))/2)...
           .*(((r_z2(1:19))+(r_z2(2:20)))/2)+cc(XX-1,YY,ZZ).*((t_z2(1:19)+t_z2(2:20))/2)...
            +dd(XX-1,YY,ZZ).*(((r_z2(1:19))+(r_z2(2:20)))/2)+gg(XX-1,YY,ZZ)));
            T_z2=sum(dt_z2);
            end
        else
            r_z2=[];t_z2=[];z_z2=[];
```

```
        end
        if isempty(r_z1)==1&& isempty(r_z2)==1 % both empty
            lrd_z=[];t_z=[];z_z=[];
        end
        if isempty(r_z1)==1&& isempty(r_z2)==0 % Z1 empty
            lrd_z=r_z2;t_z=t_z2;z_z=z_z2;T_z=T_z2;XXZ_out=XX;YYZ_out=YY;ZZZ_out=ZZ-1;
        end
        if isempty(r_z2)==1&& isempty(r_z1)==0 % Z2 empty
            lrd_z=r_z1;t_z=t_z1;z_z=z_z1;T_z=T_z1;XXZ_out=XX;YYZ_out=YY;ZZZ_out=ZZ+1;
        end
        if isempty(r_z1)==0&& isempty(r_z2)==0 % both not empty
            if T_z1<T_z2
                lrd_z=r_z1;t_z=t_z1;z_z=z_z1;T_z=T_z1;XXZ_out=XX;YYZ_out=YY;ZZZ_out=ZZ+1;
            else
                lrd_z=r_z2;t_z=t_z2;z_z=z_z2;T_z=T_z2;XXZ_out=XX;YYZ_out=YY;ZZZ_out=ZZ-1;
            end
        end
    else
        lrd_z=[];t_z=[];z_z=[];
    end
    % find real streamline among r,t,z
```

```
    if isempty(r_r)==1&& isempty(lrd_t)==1&& isempty(lrd_z)==0&&isreal(lrd_z)==1
        lrd=lrd_z;t=t_z;z=z_z;XX_out=XXZ_out;YY_out=YYZ_out;ZZ_out=ZZZ_out;
    end
    if isempty(r_r)==1&& isempty(lrd_t)==0&& isempty(lrd_z)==1&&isreal(lrd_t)==1
        lrd=lrd_t;t=t_t;z=z_t;XX_out=XXT_out;YY_out=YYT_out;ZZ_out=ZZT_out;
    end
    if isempty(r_r)==0&& isempty(lrd_t)==1&& isempty(lrd_z)==1&&isreal(r_r)==1
        lrd=r_r;t=t_r;z=z_r;XX_out=XXR_out;YY_out=YYR_out;ZZ_out=ZZR_out;
    end
    if isempty(r_r)==1&& isempty(lrd_t)==0&& isempty(lrd_z)==0
        if T_t<T_z
                lrd=lrd_t;t=t_t;z=z_t;XX_out=XXT_out;YY_out=YYT_out;ZZ_out=ZZT_out;
        else
                lrd=lrd_z;t=t_z;z=z_z;XX_out=XXZ_out;YY_out=YYZ_out;ZZ_out=ZZZ_out;
        end
    end
```

```
    if isempty(r_r)==0&& isempty(lrd_t)==0&& isempty(lrd_z)==1
        if T_t<T_r
                lrd=lrd_t;t=t_t;z=z_t;XX_out=XXT_out;YY_out=YYT_out;ZZ_out=ZZT_out;
        else
                lrd=r_r;t=t_r;z=z_r;XX_out=XXR_out;YY_out=YYR_out;ZZ_out=ZZR_out;
        end
    end
    if isempty(r_r)==0&& isempty(lrd_t)==1&& isempty(lrd_z)==0
        if T_z<T_r
                lrd=lrd_z;t=t_z;z=z_z;XX_out=XXZ_out;YY_out=YYZ_out;ZZ_out=ZZZ_out;
        else
                lrd=r_r;t=t_r;z=z_r;XX_out=XXR_out;YY_out=YYR_out;ZZ_out=ZZR_out;
        end
    end
    if isempty(r_r)==0&& isempty(lrd_t)==0&& isempty(lrd_z)==0
        l=[T_r T_t T_z];
        x=find(l==min(l));
        if x==1
```

```
            lrd=r_r;t=t_r;z=z_r;XX_out=XXR_out;YY_out=YYR_out↩
                ;ZZ_out=ZZR_out;
        end
        if x==2
            lrd=lrd_t;t=t_t;z=z_t;XX_out=XXT_out;YY_out=↩
                YYT_out;ZZ_out=ZZT_out;
        end
        if x==3
            lrd=lrd_z;t=t_z;z=z_z;XX_out=XXZ_out;YY_out=↩
                YYZ_out;ZZ_out=ZZZ_out;
        end
    end
    if isempty(t_r)==1&& isempty(t_t)==1&& isempty(t_z)↩
        ==1&&zin==0.02||zin==0.18
         XX_out=XX-1;YY_out=YY;ZZ_out=ZZ;
    end
    Rd=lrd/Rw;
    if abs(RDin-Rd(20))<=zz
        RDout=Rd(1);tout=t(1);zout=z(1);
    end
    if abs(RDin-Rd(1))<=zz
         RDout=Rd(20);tout=t(20);zout=z(20);
    end
    % Plot streamline
    x=Rd.*Rw.*cos(t);
    y=Rd.*Rw.*sin(t);
```

```
    plot3(x,y,z,'r-')
    t1=2*Tn(1)-t;
    plot3(Rd.*Rw.*cos(t1),Rd.*Rw.*sin(t1),z,'r-')
    hold on
    % Calculate new grid block coordinates
    thin=tout;
    zin=zout;
    RDin=RDout;
    XX=XX_out;
    ZZ=ZZ_out;
    YY=YY_out;
    if YY==Nt+1;
      YY=1;
    end
    if YY==0
      YY=Nt;
    end
    if abs(thin-0)<=zz
        thin=2*pi;
        YY=Nt;
    end
    if (XX<=2&& thin>=tn(1)&&thin<=tn(2)&&ZZ==2)||(XX<=2&& ↩
       thin>=tn(1)&&thin<=tn(2)&&ZZ==3)
    XX=1;
    end
end
```

```
end
end
end
```

E. 5 Two-Dimensional Stream Tube Simulator

```
clear all
% Define block no. for R and theta direction
N=50;J=25;M=N*J;
% Define wellbore radius=0.0078m, reservoir radius=0.15024m
Re=0.3048/2;Rw=0.0157/2;
% Define oundary Pressures
Pw=117210.9; Pe=98595.0;
% Define the block peremability and heterogeneity block peremability
K_block=1.58e-12;K_H=1.08e-12;
K=K_block.*ones(N,J,3);
% Define heterogeneity blick number
HENR1=42;HENR2=37;HENT1=1;HENT2=1;
K(HENR2:HENR1,HENT2:HENT1,1)=K_H;K(HENR2:HENR1,HENT2:HENT1,2)=K_H;
% Define critical saturation point
Fluid.swc=0.3;Fluid.sor=0.22;
uw=1e-3;uo=1.19e-3;%fluid viscosity cp
```

```
%Define Corey Model coefficients
aw=0.21;ao=0.48;
s=Fluid.swc*ones(N,J);
S=(s-Fluid.swc)/(1-Fluid.swc-Fluid.sor); %S
Mw=aw*S.^2/uw;Mo=ao*(1-S).^2/uo;Mt=Mw+Mo;% Total mobility
R=0:1:N-1;
ro=Rw*(Re/Rw).^(R./(N-1));% Calculate node radii
rb=ones(1,N+1);
rb(1,2:N)=(ro(:,2:N).*ro(:,1:N-1)).^0.5;% Calculate boundary radii
rb(1,[1 N+1])=[Rw^2/rb(1,2) Re^2/rb(1,N)];
Ro=repmat(ro',1,J);Rb=repmat(rb',1,J); % Ro, Rb Radius for nodes for all grid blocks
Tn=linspace(2*pi/(2*J),2*pi-2*pi/(2*J),J); % Theta Nodes angle
ttn=repmat(Tn',1,N)';% Theta Nodes angle for all grid blocks
Dn=360/J*pi/180;% Define theta angle
% Calculate Kr, Kt from the principle permeability
Kr=(K(:,:,1).*(cos(ttn)).^2+K(:,:,2).*(sin(ttn)).^2);
Kt=(K(:,:,2).*(cos(ttn)).^2+K(:,:,1).*(sin(ttn)).^2);
Mblock=Kt.*Mt;% Block Mobility
Mblockr=Kr.*Mt;
Mbr=ones(N+1,J);
Mbr([1 N+1],:)=Mblockr([1 N],:);% Upscaled mobility at r direction
```

```
Mbr(2:N,:)=log(Ro(2:N,:)./Ro(1:N-1,:))./((1./Mblockr(1:N-1,:).*log(Rb(2:N,:)./Ro(1:N-1,:)))+(1./Mblockr(2:N,:).*log(Ro(2:N,:)./Rb(2:N,:))));
Mbt=ones(N,J);% Upscaled mobility at theta direction
Mbt(:,1:J-1)=2.*Mblock(:,1:J-1).*Mblock(:,2:J)./(Mblock(:,1:J-1)+Mblock(:,2:J));
Mbt(:,J)=2.*Mblock(:,1).*Mblock(:,J)./(Mblock(:,1)+Mblock(:,J));% Last column is Mobility from the last to 1
Tr1=Rb(1,:).*Mbr(1,:)./(Ro(1,:).*(Rb(2,:)-Rb(1,:)).*(Ro(1,:)-Rb(1,:)));
Trb=Rb(2:N,:).*Mbr(2:N,:)./(Ro(2:N,:).*(Rb(3:N+1,:)-Rb(2:N,:)).*(Ro(2:N,:)-Ro(1:N-1,:)));
% Calculate the transmisibility coeffiecints (a,b,c,d,e)in different directions
e(1,:)=Rb(2,:).*Mbr(2,:)./(Ro(1,:).*(Rb(2,:)-Rw).*(Ro(2,:)-Ro(1,:)));
e(2:N-1,:)=Rb(3:N,:).*Mbr(3:N,:)./(Ro(2:N-1,:).*(Rb(3:N,:)-Rb(2:N-1,:)).*(Ro(3:N,:)-Ro(2:N-1,:)));
e(N,:)=Rb(N+1,:).*Mbr(1+N,:)./(Ro(N,:).*(Rb(N+1,:)-Rb(N,:)).*(Rb(N+1,:)-Ro(N,:)));
T=zeros(N,J-1);t=zeros(N,1);
b=Mbt./((Ro.^2).*(Dn^2));
bo=[t b(:,1:J-1)]; oob=[b(:,J) T];%Ttbo=b(1:N-1)
c=[Mbt(:,J) Mbt(:,1:J-1)]./((Ro.^2).*(Dn^2));
coo=[T c(:,1)];oc=[c(:,2:J) t];
d=[Tr1;Trb];
```

```
Tre=[zeros(1,J); e(1:N-1,:)];
a=-b-c-d-e;
am=[zeros(N,1) a(:,2:J)];
x1=reshape([Trb; zeros(1,J)]',M,1);x2=reshape(Tre',M,1);
y1=reshape(oc',M,1);y2=reshape(bo',M,1);
y10=reshape(oob',M,1);y20=reshape(coo',M,1);
AA=reshape(a',M,1);
DiagVecs=[x1,y10,y1,AA,y2,y20,x2];
DiagIndx =[-J,-J+1,-1,0,1,J-1,J];
% Coefficient matrix A for the pressure calculation
A = spdiags(DiagVecs,DiagIndx,M,M);
% Define boundary conditions
for i=1:J
    A(i,:)=0;
    A(i,i)=1;
    A(M-i+1,:)=0;
    A(M-i+1,M-i+1)=1;
end
D=zeros(M,1);
D(1:J)=Pw;
D(M-J+1:M)=Pe;
u = A\D;
p=reshape(u,J,N);P=p';% Find the permeability from pressure node to PO
RDo=log(Ro/Rw);RDb=log(Rb/Rw);d1=RDo(2:N,:)-RDb(2:N,:);
```

```
Kr2=Kr(2:N,:);Kr1=[Kr(2:N,J) Kr(2:N,1:J-1)];Kr4=[Kr(1:N-1,J) Kr(1:N-1,1:J-1)];Kr3=Kr(1:N-1,:);
Kt2=Kt(2:N,:);Kt1=[Kt(2:N,J) Kt(2:N,1:J-1)];Kt4=[Kt(1:N-1,J) Kt(1:N-1,1:J-1)];Kt3=Kt(1:N-1,:);
% Calculate pressure for half logarithmic point in the radial direction
Ptij=(Kr(2:N,:).*P(2:N,:)+Kr(1:N-1,:).*P(1:N-1,:))./(Kr(2:N,:)+Kr(1:N-1,:));
% Calculate pressure for half distance point in the angular direction
KRP=P.*Kt;
Prij=(KRP+[KRP(:,J) KRP(:,1:J-1)])./(Kt+[Kt(:,J) Kt(:,1:J-1)]);
% Determination of the corner pressure
P2=P(2:N,:);P1=[P(2:N,J) P(2:N,1:J-1)];P4=[P(1:N-1,J) P(1:N-1,1:J-1)];P3=P(1:N-1,:);
P12=Prij(2:N,:);P14=[Ptij(:,J) Ptij(:,1:J-1)];P34=Prij(1:N-1,:);P23=Ptij;
for i=1:N-1
    for j=1:J
            PP=[P1(i,j);P2(i,j);P3(i,j);P4(i,j);P12(i,j);P12(i,j);...
              P23(i,j);P23(i,j);P34(i,j);P34(i,j);P14(i,j);P14(i,j);0]; %P=pressure
            mm=[Tn(1)*d1(1,1) -Tn(1) -d1(1,1) 0 0 0 0 0 0 0 0 0 1 ;...
```

```
                0 0 0 -Tn(1)*d1(1,1) Tn(1) -d1(1,1) 0 0 0 0 0 0 1; ...
                0 0 0 0 0 0 Tn(1)*d1(1,1) Tn(1) d1(1,1) 0 0 0 1;...
                0 0 0 0 0 0 0 0 0 -Tn(1)*d1(1,1) -Tn(1) d1(1,1) 1;...
                0 0 -d1(1,1) 0 0 0 0 0 0 0 0 0 1;...
                0 0 0 0 0 -d1(1,1) 0 0 0 0 0 0 1;...
                0 0 0 0 Tn(1) 0 0 0 0 0 0 0 1;...
                0 0 0 0 0 0 0 Tn(1) 0 0 0 0 1;...
                0 0 0 0 0 0 0 0 d1(1,1) 0 0 0 1;...
                0 0 0 0 0 0 0 0 0 0 0 d1(1,1) 1;...
                0 -Tn(1) 0 0 0 0 0 0 0 0 0 0 1;...
                0 0 0 0 0 0 0 0 0 0 -Tn(1) 0 1;...
                -Kr1(i,j)*Tn(1)^2/2+Kt1(i,j)*(-d1(1,1))^2/2 Kt1(i,j)*(-d1(1,1)) Kr1(i,j)*Tn(1) ...
                Kr2(i,j)*Tn(1)^2/2-Kt2(i,j)*(-d1(1,1))^2/2 -Kt2(i,j)*(-d1(1,1)) Kr2(i,j)*Tn(1) ...
                -Kr3(i,j)*Tn(1)^2/2+Kt3(i,j)*(-d1(1,1))^2/2 Kt3(i,j)*(-d1(1,1)) -Kr3(i,j)*Tn(1) ...
                Kr4(i,j)*Tn(1)^2/2-Kt4(i,j)*(-d1(1,1))^2/2 -Kt4(i,j)*(-d1(1,1)) -Kr4(i,j)*Tn(1) 0];
            Vec=mm\PP;
            po(i,j)=Vec(13);
    end
    for j=J
```

```
end
end
% Rearrange the corner pressure to the corner point ↩
    coordinates
TO=linspace(0,2*pi-2*pi/(J),J);
tn=[TO 2*pi];
po=[Prij(1,:); po; Prij(N,:)];
po=[po po(:,1)];
LRD=log(rb./Rw);
LRD(N+1)=LRD(N+1)+LRD(1);LRD(1)=0;
RD=exp(LRD);
% Calculate the coeffiecients for the log-lin pressure ↩
    assumpution
for i=1:N
    for j=1:J
            PP=[po(i,j); po(i,j+1); po(i+1,j+1); po(i+1,j)↩
                ;]; %P=pressure
            mm=[tn(j)*LRD(i) tn(j) LRD(i) 1 ;...
                tn(j+1)*LRD(i) tn(j+1) LRD(i) 1; ...
                tn(j+1)*LRD(i+1) tn(j+1) LRD(i+1) 1;...
                tn(j)*LRD(i+1) tn(j) LRD(i+1) 1;];
                Vec=mm\PP;
                aa(i,j)=Vec(1);bb(i,j)=Vec(2);cc(i,j)=Vec(3)↩
                    ;dd(i,j)=Vec(4);
    end
end
```

```
hold on
% Plot heterogenity region
rh=rb(HENR2):(rb(HENR1+1)-rb(HENR2))/49:rb(HENR1+1);
RH=[rb(HENR2) rb(HENR1+1)];
sh=TO(HENT2):(TO(HENT1+1)-TO(HENT2))/20:TO(HENT1+1);
SH=[TO(HENT2) TO(HENT1+1)];
for i=1:2;
    plot(RH(1,i)*cos(sh),RH(1,i)*sin(sh),'b');
end
for i=1:2;
    plot(rh*cos(SH(1,i)),rh*sin(SH(1,i)),'b');
end
Xs=[];Ys=[];
for k=1:2
for j=1:J;
% Define the launching point coordinate: RDin, thin
  RDin=Re/Rw;
  thin=tn(j)+(k-1)*tn(2)/2+tn(2)/4;
  XX=N+1;
  YY=ceil(thin/tn(2));
 if YY==J+1;
      YY=1;
end
 if YY==0
      YY=J;
end
```

```
% Check if the streamline reaches to the boundary
while XX>=2
% Check if the reservoir is homogenous, if yes, use homogeneous streamline tracing method, if not,calculate C
if abs(aa(XX-1,YY)*thin+cc(XX-1,YY))>10E-7;
    step=-(log(RDin)-log(RD(XX-1)))/19;
    lrd_r=log(RDin):step:log(RD(XX-1));% upper and lower limitation for lrd
    t_r=zeros(1,length(lrd_r));
    t_r(1)=thin;% initial condition
    for i=1:(length(lrd_r)-1)% calculation loop
        F_tr=@(lrd,t) Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY))/...
                (Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY)));
        k_1=F_tr(lrd_r(i),t_r(i));
        k_2=F_tr(lrd_r(i)+0.5*step,t_r(i)+0.5*step*k_1);
        k_3=F_tr((lrd_r(i)+0.5*step),(t_r(i)+0.5*step*k_2));
        k_4=F_tr((lrd_r(i)+step),(t_r(i)+k_3*step));
        t_r(i+1)=t_r(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
    end
    % Determine if the solution is possible, if possible calculate TOF
      Rd=exp(lrd_r);
```

```
        if any(t_r<0)
            t_r=t_r+2*pi;
            if any(t_r<tn(J))||any(t_r>tn(J+1))
               lrd_r=[];t_r=[];
            end
        end
        if any(t_r>2*pi)
            t_r=t_r-2*pi;
            if any(t_r<tn(1))||any(t_r>tn(2))
               lrd_r=[];t_r=[];
            end
        end
        if all(t_r<2*pi)&&all(t_r>=0)
           if any(t_r<tn(YY))||any(t_r>tn(YY+1))
               lrd_r=[];t_r=[];
           end
        end
        if isempty(lrd_r)==0
        dt_r=((RDin-RD(XX-1))*Rw/19).*((Rd(1:19)*Rw+Rd↩
           (2:20)*Rw)/2)...
        ./(Kr(XX-1,YY)*Mt(XX-1,YY)*(aa(XX-1,YY)*(t_r(1:19)+↩
           t_r(2:20))/2+cc(XX-1,YY)));
        T_r=sum(dt_r);
        end
else
        lrd_r=[];t_r=[];
```

```
end
% Use theta as the parameteriztion to trace the streamline
 if abs(aa(XX-1,YY)*log(RDin)+bb(XX-1,YY))>1E-7;
        if abs(thin-tn(YY))>1E-7;
          step=-(thin-tn(YY))/19; % out at tn(YY)
          t_t1=thin:step:tn(YY); % upper and lower limitation for lrd
          lrd_t1=zeros(1,length(t_t1));
          lrd_t1(1)=log(RDin);% initial condition
          for i=1:(length(t_t1)-1)% calculation loop
            F_tr=@(t,lrd) Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY))/...
                (Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY)));
            k_1=F_tr(t_t1(i),lrd_t1(i));
            k_2=F_tr(t_t1(i)+0.5*step,lrd_t1(i)+0.5*step*k_1);
            k_3=F_tr((t_t1(i)+0.5*step),(lrd_t1(i)+0.5*step*k_2));
            k_4=F_tr((t_t1(i)+step),(lrd_t1(i)+k_3*step));
            lrd_t1(i+1)=lrd_t1(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
          end
         Rd=exp(lrd_t1);
        % Check if Rd out of grid block,if out,ignore this streamline
```

```
    % if not out calculate the TOF of this streamline
     if any(Rd<RD(XX-1)-1E-7)||any(Rd>RD(XX)+1E-7)
          lrd_t1=[];t_t1=[];z_t1=[];
     else
     dt_t1=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)...
     ./(Kt(XX-1,YY)*Mt(XX-1,YY)*(aa(XX-1,YY)*(lrd_t1(1:19)+lrd_t1(2:20))/2+cc(XX-1,YY)));
     T_t1=sum(dt_t1);
     end
    else
      lrd_t1=[];t_t1=[];
    end
    if abs(tn(YY+1)-thin)>1E-7;% Exit at tn(YY)
      step=(tn(YY+1)-thin)/19;
      t_t2=thin:step:tn(YY+1); % upper and lower limitation for theta
    else
        t_t2=ones(1,20)*thin;
    end
      lrd_t2=zeros(1,length(t_t2));
      lrd_t2(1)=log(RDin);% initial condition
      for i=1:(length(t_t2)-1)% calculation loop
        F_tr=@(t,lrd) Kr(XX-1,YY)*(aa(XX-1,YY)*t+cc(XX-1,YY))/...
```

```
                    (Kt(XX-1,YY)*(aa(XX-1,YY)*lrd+bb(XX-1,YY)));
            k_1=F_tr(t_t2(i),lrd_t2(i));
            k_2=F_tr(t_t2(i)+0.5*step,lrd_t2(i)+0.5*step*k_1);
            k_3=F_tr((t_t2(i)+0.5*step),(lrd_t2(i)+0.5*step*k_2));
            k_4=F_tr((t_t2(i)+step),(lrd_t2(i)+k_3*step));
            lrd_t2(i+1)=lrd_t2(i)+(1/6)*(k_1+2*k_2+2*k_3+k_4)*step;% main equation
          end
          Rd=exp(lrd_t2);
          % check if Rd out of grid block,if out,ignore this streamline
          % if not out calculate the TOF of this streamline
          if any(Rd<RD(XX-1)-1E-7)||any(Rd>RD(XX)+1E-7)
              lrd_t2=[];t_t2=[];
          else
          dt_t2=((tn(YY+1)-thin)/19).*((Rd(1:19)*Rw+Rd(2:20)*Rw)/2)...
         ./(Kt(XX-1,YY)*Mt(XX-1,YY)*(aa(XX-1,YY)*(lrd_t2(1:19)+lrd_t2(2:20))/2+cc(XX-1,YY)));
         T_t2=sum(dt_t2);
          end
        if abs(tn(YY+1)-thin)<=1E-7;% Exit at tn(YY+1)
            lrd_t2=[];t_t2=[];
```

```
        end
        if isempty(lrd_t1)==1&& isempty(lrd_t2)==1 % both empty
            lrd_t=[];t_t=[];
        end
        if isempty(lrd_t1)==1&& isempty(lrd_t2)==0 % t1 empty
            lrd_t=lrd_t2;t_t=t_t2;T_t=T_t2;
        end
        if isempty(lrd_t2)==1&& isempty(lrd_t1)==0 % t2 empty
            lrd_t=lrd_t1;t_t=t_t1;T_t=T_t1;
        end
        if isempty(lrd_t1)==0&& isempty(lrd_t2)==0 % both not empty
            if T_t1<T_t2
                lrd_t=lrd_t1;t_t=t_t1;T_t=T_t1;
            else
                lrd_t=lrd_t2;t_t=t_t2;T_t=T_t2;
            end
        end
else
      lrd_t=[];t_t=[];
end
% Find real streamline between r and t
if isempty(lrd_r)==1&& isempty(lrd_t)==0
```

```
        lrd=lrd_t;t=t_t;
end
if isempty(lrd_r)==0&& isempty(lrd_t)==1
        lrd=lrd_r;t=t_r;
end
if isempty(lrd_r)==0&& isempty(lrd_t)==0
    if T_t<T_r
            lrd=lrd_t;t=t_t;
    else
            lrd=lrd_r;t=t_r;
    end
end
if isempty(lrd_r)==1&& isempty(lrd_t)==1
          lrd=log(RDin):-(log(RDin)-log(RD(XX-1)))/19:log(RD(XX-1));t=ones(1,20)*thin;
end
Rd=exp(lrd);
if min([Rd(1) Rd(20)])==Rd(1);% Rearrange Rd from big to small
   Rds=fliplr(Rd);
   TTs=fliplr(t);
else
   Rds=Rd;
   TTs=t;
end
% Plot streamline
```

```
x=Rds.*Rw.*cos(TTs);
y=Rds.*Rw.*sin(TTs);
plot(x,y,'black');
% Calculate new grid block coordinates
if j==1&& Rds(20)==rb(HENR1+1)/Rw
      xm1(j,1)=x(20);ym1(j,1)=y(20);
      Rdh1=Rds(20);Th1=TTs(20);
 end
 if j==1&& abs(Rds(20)-rb(HENR2)/Rw)<10e-6
      xm2(j,1)=x(20);ym2(j,1)=y(20);
      Rdh2=Rds(20);Th2=TTs(20);
 end
Xs=[Xs x(1,1:19)];Ys=[Ys y(1,1:19)];
RDout=min([Rd(1) Rd(20)]);
   if RDout==Rd(1)
      tout=t(1);
   else
      tout=t(20);
   end
if XX>=2&& abs(RDout-RD(XX-1))<1e-7
       XX_out=XX-1;
else
       XX_out=XX;
end
YY_out=ceil(tout/tn(2));
if XX_out==XX
```

```
        if YY_out==YY
            if tout<thin
                YY=YY-1;
            end
            if tout>thin
                YY=YY+1;
            end
        else
            YY=YY_out;
        end
end
if YY==J+1;
    YY=1;
end
if YY==0
    YY=J;
end
RDin=RDout;
thin=tout;
XX=XX_out;
if thin==2*pi&&YY==1
    thin=0;
end
end
% Save the streamline and stream tube coordiante for front calculation
```

```
[xs ys]=size(Xs);
ds=abs((Xs(1,1:ys-1)-Xs(1,2:ys)).^2+(Ys(1,1:ys-1)-Ys(1,2:ys)).^2).^0.5;
sl=[0 cumsum(ds)];
sls=0:sl(ys)/(N*10-1):sl(ys);
Sls(2*(j-1)+k,:)=sls;
AXs(2*(j-1)+k,:)=interp1(sl,fliplr(Xs),sls,'linear');% Points used to calculate stream tube area
AYs(2*(j-1)+k,:)=interp1(sl,fliplr(Ys),sls,'linear');% Points used to calculate stream tube area
if j==1
    % Determine the stream tube length for the heterogeneous sector
    A=polyfit(Xs,Ys,2);%fitted curve
    xsh1=linspace(Xs(1),xm1(j,1),20);
    ysh1=polyval(A,xsh1);
    dsh1=abs((xsh1(1,1:19)-xsh1(1,2:20)).^2+(ysh1(1,1:19)-ysh1(1,2:20)).^2).^0.5;
    SH(1,j)=sum(dsh1);
    xsh=linspace(xm1(j,1),xm2(j,1),20);
    ysh=polyval(A,xsh);
    dsh=abs((xsh(1,1:19)-xsh(1,2:20)).^2+(ysh(1,1:19)-ysh(1,2:20)).^2).^0.5;
    DSH(j,1:19)=dsh;
    SH(2,j)=sum(dsh);
    xsh2=linspace(xm2(j,1),Xs(ys),20);
```

```
        ysh2=polyval(A,xsh2);
        dsh2=abs((xsh2(1,1:19)-xsh2(1,2:20)).^2+(ysh2(1,1:19)-↩
            ysh2(1,2:20)).^2).^0.5;
        SH(3,j)=sum(dsh2);
end
 Xs=[];Ys=[];
end
end
SLLL=(Sls(1:49,:)+Sls(2:50,:))/2;% Streamline length
SLL=(SLLL(:,1:499)+SLLL(:,2:500))/2;
XS=(AXs(1:49,:)+AXs(2:50,:))/2;
YS=(AYs(1:49,:)+AYs(2:50,:))/2;
STA=((AXs(1:50,:)-[AXs(50,:);AXs(1:49,:)]).^2+...
      (AYs(1:50,:)-[AYs(50,:);AYs(1:49,:)]).^2).^0.5;% ↩
          calculate stream tube area
save('stremtube-experiment')
```

E. 6 Two-Phase Flow Simulator

```
clear all
load streamline; %Load streamline and stream tube ↩
    coordinates
Fluid.swc=0.364;Fluid.sor=0.208;% critical saturation point
uw=1e-3;uo=1.19e-3;% fluid viscosity cp
```

```
aw=0.23;ao=0.86;% Corey Model coefficients
fi=0.43;h=0.0119;% Porosity and porous media height
% Calculate volume for each strem tube
SLLL=(Sls(1:49,:)+Sls(2:50,:))/2;
SLL=(SLLL(:,1:499)+SLLL(:,2:500))/2;
XS=(AXs(1:49,:)+AXs(2:50,:))/2; YS=(AYs(1:49,:)+AYs(2:50,:))/2;
STA=((AXs(1:50,:)-[AXs(50,:);AXs(1:49,:)]).^2+...
    (AYs(1:50,:)-[AYs(50,:);AYs(1:49,:)]).^2).^0.5;% calculate area
for i=1:26
Vs(i,:)=h*cumsum(SLLL(i,500)/(N*10-1)*0.5*(STA(i,1:499)+STA(i,2:500)));
end
STA=(STA(:,1:499)+STA(:,2:500))/2;
STA=STA*h;
syms Sw
% Calcaulte pressure for intersection points in heterogeneous stream tube
Ph(1,1)=aa(HENR1,1)*Tn(1)*log(rb(43)/Rw)+bb(HENR1,1)*Tn(1)+cc(HENR1,1)*log(rb(43)/Rw)+dd(HENR1,1);%42
Ph(2,1)=aa(HENR1,1)*Tn(1)*log(rb(37)/Rw)+bb(HENR1,1)*Tn(1)+cc(HENR2,1)*log(rb(37)/Rw)+dd(HENR2,1);%37
% Define relative permeabilities
kro=ao*((1-Sw-Fluid.sor)./(1-Fluid.swc-Fluid.sor)).^2;
krw=aw*((Sw-Fluid.swc)./(1-Fluid.swc-Fluid.sor)).^2;
```

```
% Define fractional flow for water
f=(krw/uw)./(krw/uw+kro/uo);
% Define total mobility ratio
Lamda=K_block*(krw/uw+kro/uo);
pf1=simple(diff(f,1));
pf2=simple(diff(f,2));
delta_s = 1;
f_sr=0;
% Guess the front saturation
Swi=0.7;
f1=(f-f_sr)/(Sw-Fluid.sor)-pf1;
df1=simple(diff(f1,1));
counter = 0;
% Loop to determine front saturation
while delta_s > 10^-12
    Sw=Swi;
    f10=subs(f1);df10=subs(df1);
    Sw_new=Sw-f10/df10;
    delta_s = abs(Sw_new -Sw);
    Swi=Sw_new;
    if  Swi>1||Swi<0
        error('input another value')
    end
end
% Calculate the derivates for the front saturation
SF=Swi;
```

```
pf1_fs=subs(subs(pf1,Sw,SF));
syms Sw
s=SF:(1-Fluid.sor-SF)/99:1-Fluid.sor;
PF1=subs(subs(pf1,Sw,s));PF2=subs(subs(pf2,Sw,s));
% Calculate total mobility for the fluid behind and ahead of front
LAMDA=subs(subs(Lamda,Sw,s));
LR=subs(subs(Lamda,Sw,Fluid.swc));
deltat=1;x=Rw;
sx=zeros(1e5,24);
SAB=SF:(1-Fluid.sor-SF)/99:1-Fluid.sor;
Sw=SAB;
pf1_fsAB=subs(pf1);
% Mapping 3D Riemann solution to homogeneous stream tubes
for j=3:26
    i=1;tb=0;DX=0;x=Rw;
    while x<Re
    v1=pchip(SLL(j,:),Vs(j,:),x);%find v @ x
    V=v1*PF1/pf1_fs*fi;
    A=pchip(Vs(j,:),STA(j,:),V);
    J=sum(PF2*((1-Fluid.sor-SF)/99)./(A.^2.*LAMDA));
    Dxr=x:(max(SLLL(j,:))-x)/99:max(SLL(j,:));
    Axr=pchip(SLL(j,:),STA(j,:),Dxr);
    q=(Pw-Pe)/(-v1/pf1_fs*J+sum((max(SLLL(j,:))-x)/99./(Axr))/LR);
    A_f=pchip(SLL(j,:),STA(j,:),x);
```

```
    % Calculate the front movement in dt
    x=pf1_fs*q*deltat/A_f/fi+x;
    sx(i,j)=x;
    AQ(i,j)=q*1e6*60;
    % Calculate movement in dt for S>S* and S<SL
    DX=pf1_fsAB.*q.*deltat./pchip(SLL(j,:),STA(j,:),DX)./fi+DX;
    i=i+1;
    end
    tb(j)=deltat*i;
    % Calculate flow rate after breakthrough
    for i=1:14
        syms Sw
        s=SAB(i):(1-Fluid.sor-SAB(i))/79:1-Fluid.sor;
        Sw=s;PF1=subs(pf1);PF2=subs(pf2);LAMDA=subs(Lamda);
        VS_AB(i)=pchip(SLL(j,:),Vs(j,:),(DX(i)-Rw));
        V=VS_AB(i)*PF1/pf1_fsAB(i)*fi;
        A=pchip(Vs(j,:),STA(j,:),V);
        J=sum(PF2*((1-Fluid.sor-SF)/79)./(A.^2.*LAMDA));
        Ts=tb(j)-(max(Vs(j,:))^2-VS_AB(i)^2)*J*fi/2/(Pw-Pe)/(pf1_fsAB(i)^2);
        Q_ts(i,j)=1e6*60*(Pw-Pe)*pf1_fsAB(i)/(-VS_AB(i)*J);
    end
end
% Mapping 3D Riemann solution along heterogeneous stream tubes
```

```
an=TO(2);h=0.0119;A=an*h*(x);
for j=1:2
 i=1;
% Calculate fluid movement in first section of the heterogeneous stream tube
 while x<SH(1,1)
  v1=pchip(SLL(j,:),Vs(j,:),x);%find v @ x
  V=v1*PF1/pf1_fs*fi;
  A=pchip(Vs(j,:),STA(j,:),V);
  J=sum(PF2*((1-Fluid.sor-SF)/99)./(A.^2.*LAMDA));
  Dxr=x:(max(SLLL(j,:))-x)/99:max(SLL(j,:));
  Axr=pchip(SLL(j,:),STA(j,:),Dxr);
  q=(Pw-Ph(2,1))/(-v1/pf1_fs*J+sum((max(SLLL(j,:))-x)/99./(Axr))/LR);
  A_f=pchip(SLL(j,:),STA(j,:),x);
  x=pf1_fs*q/A_f/fi*deltat+x;
  sx(i,j)=x;
  AQ(i,j)=q*1e6*60;
  i=i+1;
 end
 % Calculate fluid movement in second section of the heterogeneous stream tube
  Lamda=K_H*(krw/uw+kro/uo);
  LAMDA=subs(subs(Lamda,Sw,s));
 while x>SH(1,1)&&x<SH(2,1)+SH(1,1)
  v1=pchip(SLL(j,:),Vs(j,:),x);%find v @ x
```

```
 V=v1*PF1/pf1_fs*fi;
 A=pchip(Vs(j,:),STA(j,:),V);
 J=sum(PF2*((1-Fluid.sor-SF)/99)./(A.^2.*LAMDA));
 Dxr=x:(max(SLLL(j,:))-x)/99:max(SLL(j,:));
 Axr=pchip(SLL(j,:),STA(j,:),Dxr);
 q=(Ph(2,1)-Ph(1,1))/(-v1/pf1_fs*J+sum((max(SLLL(j,:))-x)/99./(Axr))/LR);
 A_f=pchip(SLL(j,:),STA(j,:),x);
 x=pf1_fs*q/A_f/fi*deltat+x;
 A=an*h*(x);
 sx(i,j)=x;
 AQ(i,j)=q*1e6*60;
 i=i+1;
end
% Calculate fluid movement in thrid section of the heterogeneous stream tube
Lamda=K_block*(krw/uw+kro/uo);
LAMDA=subs(subs(Lamda,Sw,s));
while x<Re&&x>SH(2,1)
 v1=pchip(SLL(j,:),Vs(j,:),x);%find v @ x
 V=v1*PF1/pf1_fs*fi;
 A=pchip(Vs(j,:),STA(j,:),V);
 J=sum(PF2*((1-Fluid.sor-SF)/99)./(A.^2.*LAMDA));
 Dxr=x:(max(SLLL(j,:))-x)/99:max(SLL(j,:));
 Axr=pchip(SLL(j,:),STA(j,:),Dxr);
```

```
    q=(Ph(1,1)-Pe)/(-v1/pf1_fs*J+sum((max(SLLL(j,:))-x)/99./(Axr))/LR);
    A_f=pchip(SLL(j,:),STA(j,:),x);
    x=pf1_fs*q/A_f/fi*deltat+x;
    A=an*h*(x);
    sx(i,j)=x;
    AQ(i,j)=q*1e6*60;
  i=i+1;
  end
  tb(j)=deltat*(i-1);
  x=Rw;
  st(j)=i-1;
  i=1;
end
%Calculate total flow rate
QS=AQ(1:2:min(tb),:);
for i=1:min(tb)/2
    QT(i,1)=2*sum(QS(i,:));
end
```