

A New Semi-analytical Streamline Simulator and Its Applications to Modelling Waterflooding Experiments

by

©Nan Zhang

A thesis 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

June 2017

St. John's

Newfoundland and Labrador

Abstract

Reservoir simulation is a tool to model the fluid flow in a reservoir over time. Streamline simulation has been proven to be an efficient approach for fine-scale geology models. With the development of engineering applications of streamline methods, researchers are now facing more challenges, for example, 1) tracing streamlines in structurally complex reservoirs; 2) improving the computational accuracy and efficiency for modeling transport problems. This research offers significant potential to meet these challenges. More specifically, this research is mainly focused on the development of a new three-dimensional, two-phase streamline simulator (using Matlab) that can model real physical displacement processes in a fast and accurate manner.

This streamline simulator solves the pressure and saturation equations sequentially. First, streamlines are traced by pressure distribution approximations; and then transport problems are solved along streamlines.

This new streamline simulator applies new semi-analytical methods to trace streamlines, including the Bilinear, Trilinear and Cubic methods. These methods generate streamlines based on pressure distribution approximations using piece-wise polynomials. Then the velocity field, streamline trajectory functions, and time-of-flight (the time a particle takes to travel along a streamline) are derived accordingly. The new streamline method and Pollock's method are systemically compared via pressure and velocity approximations, plus streamline determinations. Through these comparisons, the new methods are proven to be more accurate than Pollock's method, especially in heterogeneous problems and/or when grid resolution is low.

When certain initial conditions are imposed, this new streamline simulator applies a Riemann approach to solving transport problems along streamlines. Standard streamline simulators apply the classical Riemann solution under constant total flow rate conditions. However, the boundary conditions can also be specified by constant injection and production pressures. In this case, the flow rate varies with time, and a new semi-analytical Riemann solver presented in this thesis can be applied to map the Riemann solution along streamlines in terms of time-of-flight. Through a series of case studies using different reservoir properties, the abilities of the new streamline simulator to give sufficiently accurate solutions for homogeneous, heterogeneous, and anisotropic problems are demonstrated. Moreover, a large mobility ratio range (0.5 to 50) is tested to evaluate the performance of this streamline simulator. Through comparisons with a standard reservoir simulator (Eclipse100, Schlumberger) in these cases studies, it is demonstrated that this new streamline simulator significantly enhances the calculation speed and improves the accuracy of simulations when the underlying assumptions are valid.

Finally, the ability of the new simulator is validated and demonstrated by modeling physical waterflooding displacements. This is the first time that waterflooding experiments are performed under constant differential pressure boundaries in a two-dimensional heterogeneous macro-model. Two experiments with the same reservoir and fluid properties are performed under different boundary conditions. The new simulator is applied to history match and simulate these two experiments. The predicted and observed results show excellent agreement. The flow behavior of the fluid under a constant pressure boundary is also well understood by using the visual power of the simulator. We conclude that the new streamline simulator is very efficient and accurate in physical waterflooding processes simulations when the viscous force dominates the flow.

Acknowledgements

I feel that it is a great privilege to have pursued a Ph.D degree at Memorial University of Newfoundland. This has been an exceptional learning and personal experience for me. I take this opportunity to thank those who helped me and contributed to making this experience wonderful.

I deeply indebted to my research co-supervisors, Dr. Thormod Johansen and Dr. Lesley James; without their guidance, help and support, this work would never have been completed. Beyond the valuable advice they give, I am grateful for the manner in which we interacted. They work closely with me during difficult times.

I thank Norah for careful revising of my draft paper and manuscripts. I also wish to thank Dr. Serpil Kocabiyik for being in my examination committee. I address my sincere gratitude to all the engineers and researchers in our Hibernia Enhance Oil Recovery Laboratory, for their tremendous help in assisting me with my experiments. I thank the faculty members of the Faculty of Engineering and Applied Science, who taught me so much. I also thank the department staff for assisting me with all the administrative matters.

At a more personal level, I would like to express my most appreciation to my husband Jie Cao, mum Yuxia Duan, and dad Zhenhong Zhang, for always loving me unconditionally. I also wish to thank all my friends from Memorial University and beyond, whose company has made my years here memorable ones.

Table of Contents

Abstract	ii
Acknowledgments	iv
Table of Contents	viii
List of Tables	x
List of Figures	xvii
1 Introduction and Overview	1
1.1 Streamline Simulation	1
1.1.1 Streamline tracing methods	2
1.1.2 Solving transport problems along streamlines	5
1.1.3 Engineering applications	7
1.2 Research Objectives and Motivations	8
1.3 Thesis Outline	10
2 Literature Review	13
2.1 Streamline Tracing Methods	13
2.2 The Riemann approach	19
2.3 Waterflooding Experiments and Simulations	24

3	Tracing Streamlines Using Continuous Pressure Approximations	28
3.1	Introduction	28
3.2	Streamlines, Stream Function and Time-of-flight	29
3.3	Streamline Tracing Methodology	32
3.3.1	General steps to trace streamlines	32
3.3.2	Pressure and velocity approximation using polynomial functions	33
3.4	The Streamline Tracing Methods Based on Continuous Pressure Approximations	35
3.4.1	The Bilinear Streamline Tracing Method	36
3.4.2	The Trilinear Streamline Tracing Method	44
4	Tracing Streamlines Using Cubic Pressure Approximation Functions	54
4.1	The High-degree Polynomials for Pressure and Velocity Approximations	55
4.2	The Cubic Streamline Tracing Method	59
4.2.1	The Cubic method in two-dimensional grid blocks	60
4.2.2	The Cubic method in three-dimensional grid blocks	68
5	Comparisons of Different Streamline Tracing Methods	76
5.1	Comparisons of Pressure and Velocity Approximations with Analytical Solutions	76
5.1.1	Analytical solution to the pressure and velocity field distribution	77
5.1.2	Pressure distribution approximations	79
5.1.3	Velocity field distribution approximations	85
5.2	Comparisons of Streamline Tracing Results with Accurate Numerical Solutions	91
5.2.1	Tracing streamlines in the internal reservoir domain	92
5.2.2	Streamline distributions in the entire reservoir	103
5.3	Demonstrations of the Bilinear, Trilinear and Cubic Streamline Tracing Methods	108
5.3.1	Layered reservoir	108

5.3.2	Heterogeneous reservoir	111
5.3.3	Coarse grid block in a large reservoir	112
5.4	Discussion and Conclusion of the Chapter	113
6	Solving Transport Problems along Streamlines Using A Semi-analytical Riemann Solver	118
6.1	Immiscible Two Phase Flow Problems in a Reservoir	120
6.2	A Semi-analytical Riemann Solver under Constant Pressure Boundary	124
6.2.1	Derivation of the total flow rate as a function of time	126
6.2.2	Total flow resistance before and after the water breakthrough	129
6.2.3	The shock front location, water breakthrough time, and saturation profile	131
6.2.4	The relationship between flow rate and time	133
6.3	The New Streamline Simulator	136
6.4	Calculation Examples and Comparisons	137
6.4.1	Homogeneous problems	137
6.4.2	Heterogeneous problems	149
6.4.3	Anisotropic problems	153
6.4.4	Varying mobility ratio problems	155
6.5	Chapter Summary	160
7	Experimental Studies and Simulations for Waterflooding under Constant Differential Pressure Boundaries	162
7.1	Introduction	162
7.2	Waterflooding Experiments	163
7.2.1	Experimental setup and apparatus	163
7.2.2	The macro-model and fluid properties	165
7.2.3	Experimental procedures	167
7.2.4	The experimental results	172

7.3	Streamline Simulation Results	181
7.3.1	History matching waterflooding (first experiment), $\Delta P = 4.4 \text{ psi}$. . .	181
7.3.2	Waterflooding simulation (second experiment), $\Delta P = 5.2 \text{ psi}$	186
7.3.3	Analyzing the visualization results	189
7.4	Chapter Summary	197
8	Conclusions and Future Work	198
8.1	Conclusions	198
8.1.1	The New Streamline Tracing Methods	198
8.1.2	The Semi-analytical Riemann Solver	201
8.1.3	The Three-dimensional New Streamline Simulator	201
8.1.4	Experimental and Numerical Studies of Waterflooding Experiments under Constant Differential Pressure Boundaries	202
8.2	Future Work	203
	Bibliography	205
	Appendixes	214
	Appendix A Finite-Difference Method	214
	Appendix B Buckley-Leverett Theory	217
	Appendix C Runge-Kutta 4 th Method	226
	Appendix D Numerical Integration Method	227
	Appendix E Pollock's Method	228
	Appendix F Macro-model Fabrication	237
	Appendix G Absolute Permeability Measurement	238
	Appendix H Porosity Measurement	240
	Appendix I Capillary Number Determination	243
	Appendix J Matlab Code	243

List of Tables

2.1	Streamline tracing method literature summary	14
2.2	Riemann approach literature summary	20
2.3	Experimental and numerical studies on waterflooding literature summary	25
5.1	The reservoir and fluid properties, and boundary conditions for the homogeneous quarter five-spot pattern	78
5.2	The relative error in pressure approximation using different streamline tracing methods	84
5.3	The relative error in velocity approximations using different streamline tracing methods	91
5.4	The reservoir, fluid properties and well controls for the homogeneous case	93
5.5	The <i>Relative Distance</i> to analytical streamlines in the homogeneous case under low grid resolution (10×10)	96
5.6	The CPU time usage for the Pollock, Bilinear and Cubic streamline tracings in different grid resolutions	97
5.7	The <i>Relative Distance</i> to the true streamlines in the heterogeneous case under the low grid resolution (10×10)	100
5.8	The <i>Relative Distance</i> to the true streamlines in the anisotropic case under the low grid resolution (10×10)	102
5.9	Reservoir, fluid properties, and well controls for the 3D homogeneous case	112
5.10	Advantages and limitations of Pollock, Bilinear, and Cubic methods	117

6.1	Reservoir properties and well location for the two-dimensional homogeneous problem	138
6.2	Reservoir properties and well locations for the three-dimensional homogeneous problem	143
6.3	A summary of fluid properties and boundary conditions	156
7.1	The equipment list	165
7.2	Properties for BT-3 and BT-4 glass beads	166
7.3	Fluid properties in the waterflooding experiments	167
7.4	The connate water saturation and residual oil saturation for the two experiment	174
7.5	Input parameters for the history match	183
7.6	The relative permeability parameters in Corey's model for a successful history match result	184
7.7	The average relative errors for matching the cumulative injection and production profiles in the first experiment, $\Delta P = 4.4 \text{ psi}$	184
7.8	Input parameters to directly simulate for waterflooding at $\Delta P = 5.2 \text{ psi}$	186
7.9	Average relative error in simulated and observed cumulative rates in the second experiment, $\Delta P = 5.2 \text{ psi}$	187
G.1	Absolute permeability measurement results for BT-3 glass beads	240
G.2	Absolute permeability measurement results for BT-4 glass beads	240
H.1	Porosity measurement results for BT-3 glass beads	242
H.2	Porosity measurement results for BT-4 glass beads	242
I.1	Experimental results for capillary number at waterfront (2.5 min , $\Delta P = 5.2 \text{ psi}$)	243

List of Figures

1.1	Streamlines in a given velocity field	2
1.2	Illustration of Pollock’s tracing method through a 2D Cartesian grid block . .	3
1.3	The relationship between streamtube and its central streamline	6
1.4	An illustration of the visual power of streamline simulators (Thiele <i>et al.</i> 2010)	8
3.1	Streamline and the velocity vector	30
3.2	An example of the primary pressure nodes and grid blocks in the 2D reservoir	37
3.3	Illustration of the dual-cell system in a 2D reservoir	37
3.4	The normal fluxes across the primary grid block interfaces (w, e, n, s)	38
3.5	The normal fluxes at the dual-cell boundaries	39
3.6	Dual-cell system in different grid structures	44
3.7	An example of the primary pressure nodes and grid blocks in a 3D reservoir .	44
3.8	Illustration of a dual-cell and the sub-cells	45
3.9	The normal fluxes across the primary grid block interfaces (a, b, \dots, l)	46
3.10	The infinitely small control volumes around the primary grid block edge mid- points $(x_1, x_2, y_1, y_2, z_1, z_2)$	47
3.11	The normal fluxes at the vertex z_1 in different grid blocks	48
3.12	The normal fluxes at the 3D dual-cell boundaries	50
4.1	Illustration of Pollock’s tracing method through a 2D Cartesian grid block (same with Figure 1.2)	57
4.2	An example of pressure nodes and grid blocks in a 2D reservoir	61

4.3	An illustration of the assumption that pressure varies continuously in its perpendicular direction	62
4.4	Pressure derivative at the center of grid block interfaces	63
4.5	Determine the pressure derivative distribution along grid block interface	64
4.6	The x -directional permeability of adjacent grid blocks	65
4.7	The pressure derivatives at every grid block vertex	67
4.8	Eight vertex velocities interpreted at one grid block	67
4.9	An example of the pressure nodes and grid blocks in the 3D reservoir (same as Figure 3.7)	69
4.10	y - directional pressure derivative at the center of grid block interfaces	70
4.11	y - directional pressure derivative along x - direction	71
4.12	The twenty four velocities at the grid block edges	73
5.1	Five-spot well pattern	77
5.2	Analytical pressure and velocity field	79
5.3	The pressure approximation domain, primary grid blocks and cells	81
5.4	The analytical pressure distribution of the quarter-five-spot well pattern	82
5.5	Pressure distribution approximation and its relative error obtained by using Pollock's method	82
5.6	Pressure distribution approximation and its relative error obtained by using the Bilinear method	83
5.7	Pressure distribution approximation and its relative error obtained by using the Cubic method	84
5.8	The analytical velocity field of the quarter-five-spot well pattern	87
5.9	Velocity field approximation and its relative error obtained by using the Pollock method	88
5.10	Velocity field approximation and its relative error obtained by using the Bilinear method	89

5.11	Velocity field approximation and its relative error obtained by using the Cubic method	90
5.12	An illustration of internal and boundary domains in a 2D reservoir	92
5.13	The analytical and approximated streamlines in the homogeneous case under the low grid resolution (10×10)	93
5.14	The distance between approximated and analytical streamlines at five equal time-of-flight points	94
5.15	An illustration of errors in streamline location and time-of-flight determinations	95
5.16	The <i>Relative Distance</i> between approximated and analytical streamlines in the homogeneous case under different grid resolution	96
5.17	The true and approximated streamlines in the heterogeneous case under the low grid resolution (10×10)	99
5.18	The <i>Relative Distance</i> between the approximated and analytical streamlines in the homogeneous case under different grid resolution	99
5.19	The true and approximated streamlines in the anisotropic case under the low grid resolution (10×10)	101
5.20	The <i>Relative Distance</i> between the approximated and analytical streamlines in the anisotropic case under different grid resolution	102
5.21	The Bilinear and the true streamlines in a no-flow boundary grid block	103
5.22	The analytical and approximated streamlines in the homogeneous reservoir with low grid resolution (10×10)	104
5.23	Streamline numbers	105
5.24	The <i>Relative Distance</i> for individual streamlines in the entire homogeneous reservoir with low grid resolution (10×10)	105
5.25	The true and approximated streamlines in the entire heterogeneous reservoir under the low grid resolution (10×10)	106
5.26	The <i>Relative Distance</i> for individual streamlines in the entire heterogeneous reservoir at low grid resolution (10×10)	107

5.27	The <i>Relative Distance</i> for the approximated streamline No.1 to the analytical solution in a homogeneous reservoir under different grid resolutions	107
5.28	The analytical and Bilinear streamlines in a homogeneous reservoir under high grid resolution (100×100)	108
5.29	The streamline distribution in the layered reservoir (Cubic method)	109
5.30	The streamline distribution in the 3D layered reservoir (Cubic method)	110
5.31	The streamline distribution in the reservoir has two shale layers (Bilinear method)	111
5.32	The pressure distribution in a rectangular reservoir	113
5.33	The streamline tracing results with and without the coarse grid block	113
6.1	Solve waterflooding problems along streamlines/streamtube	121
6.2	A typical saturation profile before water breakthrough	122
6.3	The relationship between a streamtube and its central streamline	123
6.4	Typical saturation profiles for a Riemann problem at different time; where BT is the water break through time	130
6.5	Flow chart of semi-analytical Riemann approach along streamline	135
6.6	40 streamlines in 2D homogeneous problem	139
6.7	The flow rate asociate with each streamline varies with time	140
6.8	The flow resistance for the streamline No.1 in the Figure 6.7	140
6.9	The sensitivity of the fluid production rates to the number of streamlines	142
6.10	The production profile for 2D homogeneous problem	143
6.11	Streamlines in 3D homogeneous problem	144
6.12	The fluids production rates in 3D homogeneous problem	144
6.13	The CPU usages for the 2D and 3D waterflooding simulations	145
6.14	An illustration of parallel and diagonal grid blocks	145
6.15	Comparison of saturation contours for parallel and diagonal grid block in a five-spot displacement, $M = 1$, (Brand <i>et al.</i> , 1991)	146
6.16	An illustration of the numerical dispersion in finite-difference approximations	147

6.17	The permeability distribution in the 2D heterogeneous problem	149
6.18	The histogram of the permeability distribution in 2D heterogeneous problem	149
6.19	Streamlines in 2D heterogeneous problem	150
6.20	The fluids production rates in 2D heterogeneous problem	150
6.21	The permeability distribution in 3D heterogeneous problem	151
6.22	The histogram of the permeability distribution in 3D heterogeneous problem	151
6.23	Streamlines in 3D heterogeneous problem	152
6.24	The fluids production rates in 3D heterogeneous problem	152
6.25	Streamlines in 2D anisotropic problem	153
6.26	The fluids production rates in 2D anisotropic problem	153
6.27	Streamlines in 3D anisotropic problem	154
6.28	The fluids production rates in 3D anisotropic problem	154
6.29	The fractional flow curves, and the shock front water saturations for the four cases	156
6.30	The production profiles results from the new streamline simulator and Eclipse100 for the first case ($M = 0.5$)	157
6.31	The production profiles results from the new streamline simulator and Eclipse100 for the second case ($M = 2$)	158
6.32	The production profiles results from the new streamline simulator and Eclipse100 for the third case ($M = 10$)	159
6.33	The production profiles results from the new streamline simulator and Eclipse100 for the last case ($M = 50$)	159
7.1	The schematic of the waterflooding experiments	164
7.2	Heterogeneous glass-bead pack macro-model	166
7.3	The pressure profile for the first experiment, $\Delta P = 4.4 \text{ psi}$	173
7.4	The pressure profile for the second experiment, $\Delta P = 5.2 \text{ psi}$	173
7.5	The experimental cumulative injection and production profiles	174
7.6	Displacement front movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$. . .	176

7.6	Displacement front movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)	177
7.6	Displacement front movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)	178
7.7	Displacement front movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$. . .	179
7.7	Displacement front movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$ (cont.)	180
7.8	The relative permeability curves for a successful history match result	184
7.9	Experimental and history match results for the experiment, $\Delta P = 4.4 \text{ psi}$. .	185
7.10	Observed vs Simulated results for the experiment, $\Delta P = 4.4 \text{ psi}$	185
7.11	Experimental and simulated results for the experiment, $\Delta P = 5.2 \text{ psi}$	188
7.12	Experimental and simulated results for the experiment, $\Delta P = 5.2 \text{ psi}$	188
7.13	Capillary number at the waterfront for different streamlines, $\Delta P = 4.4 \text{ psi}$. .	190
7.14	Capillary number at the waterfront for different streamlines, $\Delta P = 5.2 \text{ psi}$. .	190
7.15	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$	192
7.15	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)	193
7.15	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)	194
7.16	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$	195
7.16	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$ (cont.)	196
A.1	An illustration of the finite difference method in grid block	215
B.1	One-dimensional flow system	218
B.2	The relative permeability plot and the fractional flow function	219
B.3	An example of the fractional flow curve and its derivative	222
B.4	The unphysical saturation profile	223
B.5	The saturation at the shock front	223
B.6	The physical saturation profile	224

B.7	The cross sectional area of a streamtube	224
D.1	Illustration of trapezoidal rule used on a sequence of samples	228
E.1	An example of the pressure nodes and grid blocks in the two-dimensional reservoir	229
E.2	The velocity at grid block interfaces	230
F.1	The two parts for plexiglass box	238
G.1	Permeability measurement experiment setup	239
H.1	Porosity measurement experiment setup	241
I.1	Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psig}$	244

List of Nomenclature

Symbols

a	Maximum relative permeability
A	Cross sectional area, [m^2]
BT	Breakthrough time, [s]
BV	Bulk volume, [m^3]
Ca	Capillary number
dl	The distance between two points, [m]
DL	The relative distance
DS	Pressure derivative slope, [Pa/s]
e	Error
f	Water fractional flow
G	Geometrical resistance, or shape factor, [$1/m$]
i	Individual point
K	Permeability, [m^2]
k	Relative permeability
L	Length, [m]
M	End point mobility ratio
n	Exponents for relative permeability
n_x, n_y	± 1
\mathbf{n}	Unit normal

N	Number of grid blocks/points
$OOIP$	Original oil in place, [m^3]
P	Pressure, [Pa]
P_c	Capillary pressure, [Pa]
PV	Pore volume, [m^3]
q	Flow rate, [m^3/s]
Q	Cumulative volume, [m^3]
R	Flow resistance in the streamtube, [Pa]
RD	Relative difference
re	Relative error
r_k	Permeability ratio
S, s	Saturation
$t, Time$	Time, [s]
T	Transmissibility, [$m^3/(s \cdot Pa)$]
u, \mathbf{u}	Darcy velocity, [m/s]
V	Streamtube volume, [m^3]

Greek Symbols

v, \mathbf{v}	Velocity, [m/s]
γ	Interfacial tension, [N/m]
Γ	Parameterization of boundary
θ	Angle, [$^\circ$]
λ	Mobility, [$m^2/(Pa \cdot s)$]
μ	Viscosity, [cp] or [$Pa \cdot s$]
ξ	Streamline arc-length, [m]
ρ	Density, [kg/m^3]
τ	Time-of-flight, [s]
ϕ	Porosity

ψ	Streamfunction
Ω	Control volume

Others

\times	Cross product
∇	Gradient operator
$\nabla \cdot$	Divergence operator
Δ	Difference operator
<i>max</i>	Maximum operator
<i>min</i>	Minimum operator

Constant Coefficients

$a_1, a_2, a_3, a_x, a_y, a_z, b, b_1, b_2$	Velocity coefficients
$c_1, c_2, c_{1x}, c_{2x}, c_{1y}, c_{3y}, c_{2z}, c_{3z}$	Velocity coefficients
$d_{1x}, d_{2x}, d_{3x}, d_{5x}, d_{2y}, d_{3y}$	Velocity coefficients
$d_{4y}, d_{6y}, d_{5z}, d_{6z}, e_1, e_2$	Velocity coefficients
$P_0, A_1, A_2, A_3, B, B_1, B_2, B_3, C, C_1, C_2, C_3$	Pressure coefficients
$D, D_1, D_2, D_3, D_4, D_5, D_6$	Pressure coefficients
$A_x, A_y, B_x, B_y, C_x, C_y, \tilde{C}$	Constant coefficients

Chapter 1

Introduction and Overview

This chapter presents the challenges of streamline methods on giving accurate simulation results and defines the objectives, motivation and outline of this research thesis.

1.1 Streamline Simulation

Numerical reservoir simulation is an engineering tool that describes the dynamic physical processes of fluid flow in a reservoir. It is widely applied to assist the management of production operations. Most of the commercial reservoir simulators apply the conventional finite-difference numerical method. With the rapid developments of reservoir characterization, fine-scale geology models with multimillion grid blocks are now commonly encountered. This requires large improvements on the computational speed in reservoir simulation. Driven by this challenge, engineering applications and mathematical foundations of streamline simulation have been developed rapidly in the last two decades. Nowadays, the streamline method has become an effective alternative simulation method to the conventional finite-difference simulation method (Datta-Gupta and King 2007).

A streamline simulator can model displacement processes in a fast and accurate manner.

It achieves this goal by fulfilling the following three steps: First, it solves the pressure equations using a grid block-based numerical method; then it approximates the velocity field in each grid block and determines streamlines; and finally, solves the transport problem along streamlines. Streamlines should be updated when the reservoir conditions are redefined or to honor a changing mobility field (Thiele *et al.* 1996). The simulation errors will be induced at every step in a calculation, the scope of this research thesis is to improve the accuracy of streamline tracing methods and the approach to solve transport problems along streamlines.

1.1.1 Streamline tracing methods

Streamlines are a family of curves that are tangential to the instantaneous velocity field, as depicted in Figure 1.1. The most intuitive feature of a streamline simulator is the power to visualize how petrophysical properties, well conditions and gravity interact to determine the flood front and flow patterns. In steady state flow, the streamline pattern is identical to the flow-lines or path-lines that describe the trajectory of fluid particles.

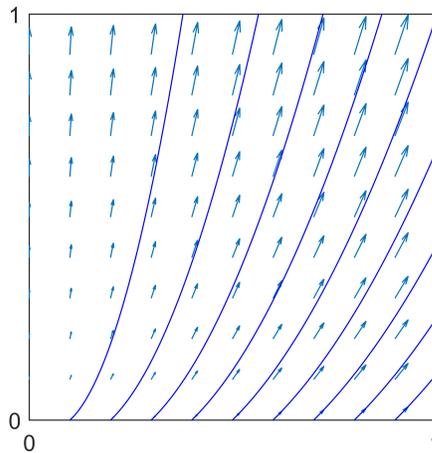


Figure 1.1: Streamlines in a given velocity field

Currently, the most practical and widely applied streamline tracing method in commercial simulators is Pollock's method (Pollock, 1988). This method is computationally highly efficient in tracing streamlines in both two- and three- dimensional problems. This is because

the streamline tracing is grid block based, and the streamline location is expressed in an explicit formula.

Tracing streamline in an orthogonal grid block using Pollock’s method is simple. First, the velocity across each grid block interface is calculated using Darcy’s law based on the grid block pressures on each side of the interface (a detailed description for solving the pressures and velocities using a finite difference method is given in Appendix A). Then, the velocity field is approximated by a linear function in each coordinate direction. As shown in Figure 1.2 , in the x -direction,

$$v(x) = v_{x_1} + a_x(x - x_1); \tag{1.1}$$

where,

$$a_x = \frac{v_{x_2} - v_{x_1}}{\Delta x}, \tag{1.2}$$

where v_{x_i} is the x -velocity at the interface when $x = x_i$, ($i = 1,2$) and a_x is the velocity gradient (or the velocity coefficient) in the x -direction.

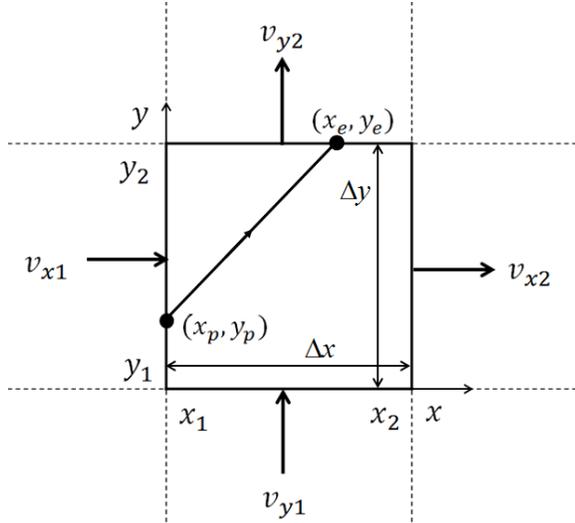


Figure 1.2: Illustration of Pollock’s tracing method through a 2D Cartesian grid block

In Pollock’s method, the streamline location is expressed in terms of time-of-flight. The time-of-flight is defined as the time that a particle takes to travel from its launching point to a certain point on the streamline. According to its definition, the incremental time-of-

flight $\Delta\tau_x$ for a particle traveling within this local grid block from (x_p, y_p, z_p) to the grid block interface x_2 is given by,

$$\Delta\tau_x = \int_{x_p}^{x_2} \frac{dx}{v_x} = \frac{1}{a_x} \ln \left[\frac{v_{x_1} + a_x (x_2 - x_1)}{v_{x_1} + a_x (x_p - x_1)} \right]. \quad (1.3)$$

In y - or z - directions, the time-of-flight for the particle to travel to y_2 or z_2 respectively, is given by,

$$\Delta\tau_y = \frac{1}{a_y} \ln \left[\frac{v_{y_1} + a_y (y_2 - y_1)}{v_{y_1} + a_y (y_p - y_1)} \right], \quad (1.4)$$

$$\Delta\tau_z = \frac{1}{a_z} \ln \left[\frac{v_{z_1} + a_z (z_2 - z_1)}{v_{z_1} + a_z (z_p - z_1)} \right], \quad (1.5)$$

where a_y and a_z is the velocity gradient (or the velocity coefficient) in y -direction and z -direction, respectively.

The particle will exit the local grid block from the interface that requires the shortest time-of-flight, therefore, the true incremental time-of-flight is,

$$\Delta\tau_e = \min(\Delta\tau_x, \Delta\tau_y, \Delta\tau_z). \quad (1.6)$$

The exit locations are easily calculated by substituting $\Delta\tau_e$ in Eq. 1.3, 1.4 and 1.5 respectively, *i.e.*,

$$x_e = \frac{1}{a_x} \ln \left[v_{x_p} \exp(a_x \Delta\tau_e) - v_{x_1} \right] + x_1, \quad (1.7)$$

$$y_e = \frac{1}{a_y} \ln \left[v_{y_p} \exp(a_y \Delta\tau_e) - v_{y_1} \right] + y_1, \quad (1.8)$$

$$z_e = \frac{1}{a_z} \ln \left[v_{z_p} \exp(a_z \Delta\tau_e) - v_{z_1} \right] + z_1, \quad (1.9)$$

where the $v_{x_p} = v_{x_1} + a_x (x_p - x_1)$, $v_{y_p} = v_{y_1} + a_y (y_p - y_1)$ and $v_{z_p} = v_{z_1} + a_z (z_p - z_1)$ are the directional velocities at the particle's initial location.

After the exit point has been located, this point can be treated as the entry point at the next grid block, and then the same algorithm introduced above can be applied to continue the tracing until the outflow boundary is reached. By following the particle traveling from an injector to a producer through grid blocks, a complete streamline is traced.

As any streamline tracing method, the essential idea of Pollock’s method is its velocity approximation method. The efficiency of Pollock’s method relies on the linear velocity field approximation function. However, this simple approximation can not ensure the local mass conservation, since

$$\nabla \cdot \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} = a_x + a_y + a_z, \quad (1.10)$$

may not equal to zero. The Pollock method may lead to large errors in both streamline location and time-of-flight (Matringe and Gerritsen 2004). Especially if the structure of the reservoir is complex, the velocity across the same grid block interface varies, theoretically even in opposite directions (Thiele 2001). The linear approximation of a velocity field is therefore generally not sufficient. The need for more accurate streamlines motivated researchers to develop other streamline tracing methods.

1.1.2 Solving transport problems along streamlines

As a reservoir simulation tool, streamline simulators are known for their high computational efficiency. Different from grid block based simulators, streamline methods treat fluids as they flow along each one-dimensional streamline rather than travel through three-dimensional grid blocks. The transport problem underlying a two-dimensional or three-dimensional grid block model is decoupled into streamlines (Bratvedt *et al.* 1992, Datta-Gupta and King 1995) or streamtubes (Higgins and Leighton 1962a, Martin and Wegner 1979, Thiele 1996). This is the primary reason why streamline methods are orders of magnitude faster than the conventional grid block based simulation methods.

Analytical Riemann solutions to the one-dimensional two-phase immiscible flow problems with certain initial conditions were described by Buckley and Leverett (1942) under constant flow rate boundary conditions and by Johansen and James (2016) under constant pressure boundary conditions. When these initial and boundary conditions apply, the solution to the mass conservation equation along one-dimensional streamtubes can be obtained analyt-

ically. More specifically, the analytical Riemann solution to these problems can be mapped along streamtubes analytically with the known changing cross-sectional area. Mapping the Riemann solution along streamtubes/streamlines is referred to as the *Riemann approach* in this research thesis.

The Buckley and Leverett (1942) fractional flow theory under constant flow rate boundary conditions has been widely applied in Riemann approaches along streamlines/streamtubes since 1962 (Higgins and Leighton 1962a, Martin and Wegner 1979). The analytical Riemann approach along streamtubes under constant pressure boundary conditions was recently published by Johansen and Liu (2016).

The Riemann approach along streamlines is very similar to the one along streamtubes, if one considers the relationship between a streamtube and its central streamline. As shown in Figure 1.3, the cross-sectional area of the streamtube A can be determined by the total flow rate q within this streamtube and the velocity v at the central streamline,

$$A(\xi) = \frac{q}{\phi v(\xi)}, \quad (1.11)$$

where ϕ is porosity.

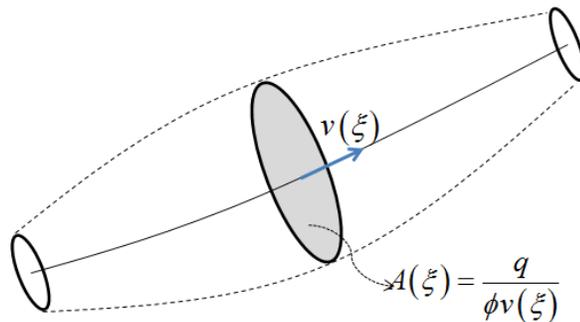


Figure 1.3: The relationship between streamtube and its central streamline

Different from streamtubes, streamlines are one-dimensional lines which have no volume. Tracing streamlines are easier than streamtubes because the complex three-dimensional geometry of the streamtube volume is difficult to describe (Bratvedt 1997). Instead, the cross-sectional area of a streamtube can be determined numerically using its central streamline.

Thus, the Riemann approach along streamlines is a semi-analytical method.

Solving the transport problems along streamlines using a semi-analytical Riemann approach is faster and more accurate than using purely numerical methods. In this research thesis, a semi-analytical Riemann solver for immiscible two-phase flow under constant pressure boundaries is introduced. This solver avoids the complicated and time consuming calculations of the purely analytical solution for the Riemann problem along streamtubes in Johansen and Liu (2016).

1.1.3 Engineering applications

Streamline simulators can be applied to quickly achieve the general objectives of reservoir simulation, including flow paths modeling, screening and ranking of possible producing schemes due to their computational efficiency. Streamline simulators are ideally suited for modeling large, geologically heterogeneous, multi-well systems where flow is dominated by pressure gradients. Streamline methods are not well suited to model the fluid displacements that are largely driven by diffusive phenomena, such as capillary pressure, transverse diffusion, or compressibility. In these situations, the fluid will flow across streamtube boundaries. Nevertheless, some modern streamline simulators can account for gravity effects (Glimm *et al.* 1983, Bratvedt *et al.* 1992), capillary effects (Berenblyum *et al.* 2003), and for compressible fluids (Ingebrigtsen *et al.*, 1999).

In addition to the computational efficiency, the visual power of streamline simulators make them very appealing to other important field applications, including quick identification of injector-producer pairs, maximization of the displacement efficiency (Thiele *et al.* 2006, Batycky *et al.* 2008), rate allocation and flood front management, “upgridding” from fine-scale models (Christie and Blunt 2001, Gautier *et al.* 1999, Samier *et al.* 2002), and history matching (Wang and Kovsky 2000, Agarwal and Blunt 2003, Caers *et al.* 2002). An illustration of the visual power of streamline simulators is given in Figure 1.4. As shown

in this figure, streamlines represent a snapshot of the instantaneous flow field, and the connections between wells.

Streamline simulators can also be applied to study lab-scale displacement processes. The physical flow pattern in a porous medium can be understood and modeled by using a streamline simulator when the fluid flow is dominated by viscous forces. When the physical flow pattern deviates from the simulated results, the capillary pressure may start to influence the flow, particularly far from wells where the capillary number is small.

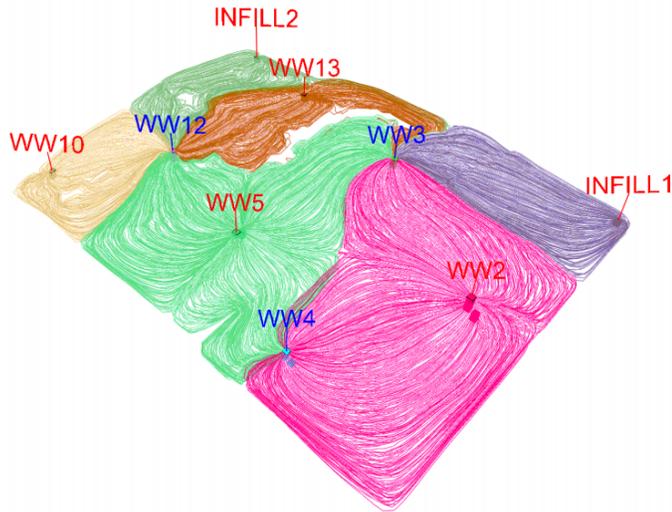


Figure 1.4: An illustration of the visual power of streamline simulators (Thiele *et al.* 2010)

1.2 Research Objectives and Motivations

This research is mainly focused on building a new three-dimensional, two-phase streamline simulator to meet the following four objectives:

1. Improve the calculation accuracy and preserve the computational efficiency in streamline tracing compared to existing methods;
2. Introduce a semi-analytical Riemann solver along streamlines;

3. Extend the applications of a semi-analytical Riemann approach along streamlines from constant flow rate boundaries to constant pressure boundaries;
4. Validate and demonstrate the ability of the streamline simulator by comparing the simulation results with the observed results from lab-scale waterflooding experiments in heterogeneous porous medium.

These four objectives are important because streamline simulation is a relatively new technology to researchers and engineers compared to grid block based simulations. Furthermore, streamline simulation has been lacking many of the theoretical foundations that are well developed in conventional finite-difference simulation (Datta-Gupta and King 2007). Therefore, there is a potential for improvement of this simulation technology, as demonstrated in this research thesis. Through this research, the principal theories of streamline tracing and transport problem solving along streamlines are rigorously examined and improved. Sufficiently accurate results can be effectively obtained by the new streamline simulator through applying these principle theories for streamline tracing and waterflooding simulations. The new streamline simulator has a large potential to save considerable time in solving transport problems compared to the conventional grid block based simulators.

The motivation for this research is originated from the new concept of tracing streamlines using continuous pressure approximation functions in both Cartesian and Polar coordinate systems (Johansen 2010). This research thesis aim to develop new streamline tracing methods based on newly proposed pressure approximation functions. Traditionally, streamlines are traced by directly using velocity approximation functions, which lack physical significance and may lead to large errors when the reservoir is structurally complex (Thiele *et al.* 2011). When the pressure gradient is the main driving force for fluid flow, the pressure distribution approximations are important for obtaining realistic velocity fields and accurate streamline tracing results. A systematic comparison of different streamline tracing methods for pressure, velocity approximation and streamline tracing is presented to illustrate the advantages and limitations of each method.

Additionally, the recent research publication of the analytical solution for two-phase Riemann problems with constant pressure boundaries (Johansen and James 2016) inspires this research to map this Riemann solution along streamlines. In real field applications, wells can be set at constant pressure control and the production flow rates vary with time. When certain initial conditions are met, the analytical Riemann approach along streamtubes (Johansen and Liu 2016) can be applied to solve the transport problems. However, tracking the geometry of streamtubes is complicated. Instead, streamlines are easier to generate. In this research thesis, the Riemann solutions are mapped along streamlines in terms of time-of-flight.

Finally, the application of new streamline simulator for modeling physical problems is validated experimentally for waterflooding processes. During this research, waterflooding experiments are performed in a square two-dimensional heterogeneous unconsolidated macro-model (James, 2012) under constant differential pressure boundaries for the first time. These experiments are designed to mimic the real waterflooding processes under constant pressure boundaries in a quarter-five spot pattern. A quarter-five spot is a flow pattern where one injector and one producer are located in the left bottom and the right top of a square reservoir bounded by no-flow boundaries, respectively. When the viscous force dominates the flow, the movements of displacement front can be understood and modeled by using the new streamline simulator. The new streamline simulator can be validated when the experimental results are in agreement with the simulations.

1.3 Thesis Outline

Chapter 2 describes the relevant important literature in the same knowledge field as this research, including streamline tracing methods, Riemann approach, waterflooding experiments, and simulations. This chapter also relates this research to other research in the same knowledge field and explains the contributions to produce more accurate simulation results

for physical two-phase immiscible displacement problems at low computational cost.

Chapter 3 introduces the new streamline tracing methods in a systematic manner. This chapter first introduces the basic concepts of streamlines and explains the general steps of streamline tracing. Thereafter, it explains why the approximations of pressure distribution and velocity field in each grid block are essential for tracing accurate streamlines. When a pressure gradient is the main driving force for fluid flow, the pressure distribution is first approximated using piece-wise polynomials, and then the velocity field is derived from Darcy's law. To trace streamlines accurately, four requirements for the pressure and velocity approximation functions are introduced. By assuming the pressure is continuous everywhere in the reservoir, a Bilinear and Trilinear method can be applied. Chapter 3 continues by providing a detailed description of the Bilinear and Trilinear streamline tracing methods in two- and three-dimensional grid blocks.

Chapter 4 introduces a Cubic streamline tracing method by assuming the pressure distribution is piece-wise high-order polynomial. This chapter introduces the high-order polynomial pressure approximation function and derives the velocity field and streamline trajectory function sequentially in both two- and three-dimensional grid blocks. Compared to the first-order polynomials, the Cubic method determines more coefficients in pressure and velocity approximations and therefore leads to more accurate streamline simulation results.

Chapter 5 systematically compares the Bilinear, Cubic and Pollock streamline tracing methods from three different perspectives. More specifically, approximations in pressure distribution, velocity field, and in streamline tracing are considered. Through comparisons, the key differences between these methods are clearly addressed. Discussions about the advantages and limitations of these three methods are given at the end of this chapter.

Chapter 6 introduces a semi-analytical Riemann approach along streamlines under constant pressure boundaries. More specifically, it presents the derivation of the semi-analytical solution for flow rates along streamlines as a function of time, and the saturation profile in

terms of time-of-flight. In addition, the new streamline simulator is developed by combining the Cubic streamline tracing method with the Riemann approach along streamlines. Several case studies with different reservoir and fluid properties are performed to estimate the accuracy and efficiency of this new simulator through comparisons with Eclipse100 (Copyright Schlumberger, 2009-2015) reservoir simulator. Through these case studies, the abilities of the new streamline simulator to give sufficiently accurate solutions for homogeneous, heterogeneous and anisotropic problems are demonstrated. Moreover, a large mobility ratio range (0.5 to 50) is tested to evaluate the performance of this simulator.

Chapter 7 discusses the experimental validation of the new streamline simulator in history matching and simulating the production profile from lab-scale waterflooding experiments. Two waterflooding experiments under different constant differential pressures are performed in the two-dimensional heterogeneous glass bead pack macro-model. Section 7.2 introduces the experimental setup, fluid properties, heterogeneous reservoir properties and experimental procedures. The production profile and displacement front as a function of time are recorded throughout the displacement process. Then, the new streamline simulator is applied to model these physical processes. Through history matching the first experiment and then simulating the second experiment, the capabilities of the new streamline simulator in modeling real physical problems are demonstrated and independently validated by experiments. When the simulation results show a good agreement with the observations, the fluid flow patterns under constant pressure boundaries are well understood.

Chapter 8 summarizes the main work in this research and draws the main conclusions. Furthermore, recommended future work in related areas is discussed.

Chapter 2

Literature Review

Much research has been done on streamline simulation in both petroleum and ground water literature. This chapter reviews the major research that has been done in the same field as this research. This includes streamline tracing methods, Riemann approaches, and experimental and numerical simulation studies of two-phase immiscible displacement processes in heterogeneous porous medium. This chapter relates this research to other research in this knowledge field and explains its contributions to produce more accurate simulation results for two-phase immiscible displacement problems at low computational cost.

2.1 Streamline Tracing Methods

The history of the theoretical foundation of streamlines can be traced back to 1781 when Lagrange introduced the two-dimensional Lagrange stream function. Muskat and Wyckoff (1934) were the first to apply the streamline methods to solve reservoir engineering problems. Since then, important contributions were made through various researchers with increasing interest in the streamline method and its applications to reservoir simulation. A brief summary of the reviewed literature is listed in Table 2.1.

Table 2.1: Streamline tracing method literature summary

Year	Author(s)	Main Contributions
1781	Lagrange	Introduced the concept of stream function in 2D
1934	Muskat and Wyckoff	First petroleum literature publication of streamline methods
1937	Muskat and Wyckoff	Derived the governing analytical solution for stream function
1951	Fay and Pratts	Introduced the semi-analytical streamline tracing method in 2D
1967	Caudle	Published the solution to analytical streamlines for different well patterns
1988	Pollock	Introduced the most efficient semi-analytical streamline tracing method in 3D using time-of-flight
1992	Cordes and Kinzelbach	Tracing streamlines based on Galerkin finite element method
1995	Datta-Gupta and King	Derived semi-analytical streamline trajectory functions in each grid block
2001	Prevost <i>et al.</i>	Traced streamlines efficiently in irregular grid
2006	Juanes and Matringe	Introduced a streamline tracing method in two dimensions based on a mixed finite element scheme
2010	Johansen	Introduced a new semi-analytical streamline tracing method based on pressure approximation functions
2012	Zhang <i>et al.</i>	Performed a comprehensive study of velocity interpolation methods for streamline tracing in polygons

Lagrange in 1781 defined stream functions for incompressible flows in two dimensions. A stream function represents the trajectories of particles in a steady state flow. The stream function can be used to plot streamlines, since it keeps a constant value along a streamline.

Muskat and Wyckoff (1934) were the first to apply the analytical streamline methods to study the effects of different well patterns on oil recovery. In 1937, they presented the governing analytical solutions for the stream function and the potential function in simple

two-dimensional domains using the assumption of incompressible flow.

Fay and Pratts (1951) applied a semi-analytical approach to predict tracer and two-phase flow in a two-well, two-dimensional, homogeneous system. This semi-analytical approach was applied to obtain a numerical solution for the stream function and generating streamlines at equal increments of the stream function. Since the streamlines would shift positions under two-phase displacement, Fay and Pratts applied multi-stage calculations to trace the instantaneous streamlines.

Caudle (1967) published the analytical stream function expressions by using the line source solution of flux and the superposition principles for different flow patterns. There are four underlying assumptions for the analytical solutions: 1) the mobility ratio is equal to one; 2) the displacement process is considered stable and piston like; 3) capillary, gravity, and compressibility effects are neglected; and 4) the porous medium is homogeneous and isotropic. Analytical streamlines give exact solutions when these underlying assumptions are met. Therefore, it is used in this research to evaluate the accuracy of different streamline tracing methods.

In general, streamline tracing methods published after Pollock (1988) are usually accomplished in three steps. First, the pressure and flux at each discrete grid block are solved using a numerical method; then, a suitable velocity field is approximated over each discrete grid block; finally, the streamlines and the incremental time-of-flight at each grid block are integrated using the velocity field (Zhang *et al.* 2012). There are many velocity interpretation methods proposed in the literature; the appropriate method for a given situation is dependent on the grid structure and numerical method applied in the first step.

The finite difference method on structured grids is widely applied in reservoir simulation. The method itself is easy to implement, and its solution is mass conservative. This numerical scheme was applied in Pollock's method (Pollock, 1988) for solving the pressure, which is the most widely applied streamline tracing method in commercial streamline simulators and

described in Chapter 1.

Pollock's method (1988) is considered as a breakthrough work for computational efficiency in three dimensions. It is also currently the most commonly used streamline tracing method and one of the fundamental theories of modern streamline-based simulators. Pollock's method is a simple and semi-analytical approach that solves the streamline tracing problem in terms of time-of-flight for any given launching point. The streamline tracing is grid block based, the streamline location and time-of-flight are calculated analytically in each grid block. By following the particle as it moves from injector to producer through different blocks, the streamline can be determined through multi-dimensional flow fields. This method is based on two approximations, 1) the flow velocity (or the volumetric flux) is approximated by the centered finite-difference approach at each grid block interface; and 2) the directional velocity components are assumed to vary linearly only in their own direction within each grid block. These two approximations ensure the computational efficiency of Pollock's method is high.

The piece-wise linear and continuous approximation of the volumetric flux is also presented by Datta-Gupta and King (1995) in a semi-analytical tracer motion modeling method. They approximated streamlines by one piece-wise hyperbolic interval per grid block. Along each of these intervals, the analytical streamline trajectory function and the transit time are solved exactly.

The geologic complicity of real reservoirs are commonly represented by irregularly meshed grid block systems. However, Pollock's method was derived assuming orthogonal grid blocks. To trace streamlines efficiently in irregular grid systems, Prevost *et al.* (2002) presented a method using space transformation. The irregular grid block is first transformed to a unit cube; then streamlines are traced within the unit cube where the Pollock's algorithm can be applied; and finally the streamline exit point is transformed back to the original irregular grid block.

Nowadays, most commercial streamline simulators apply finite difference scheme and Pollock's method due to their high efficiency in three dimensions. The underlying assumption of Pollock's method is that there is a single velocity per grid block face. Structurally complex reservoirs, on the other hand, often require multiple connections across a single interface in a grid block. For instance, in the presence of faults, grid blocks might now have multiple velocities across a single interface, theoretically even in opposite directions (Thiele *et al.* 2011). The following paragraphs review other numerical schemes and streamline tracing methods developed to achieve more accurate results.

In a structurally complex reservoir, finite element and finite volume methods provide a better representation than finite difference method. The Galerkin finite element method can be used to simulate geologically complex reservoirs. The use of triangles and polygons can represent geological structures more easily than structured grids. This method applies piecewise polynomial to represent the pressure distribution. By taking directional derivatives of the pressures, the velocity field can be computed. However, the velocities are discontinuous across element boundaries, and therefore will lead to poor streamline tracing results. In order to solve this problem, Cordes and Kinzelbach (1992) proposed a post-processing method that reconstructs the flux at sub-cell interfaces. A local "patch" is defined to calculate these fluxes by mass conservation, flux continuity and irrotationality laws. In triangular grids, the velocity field is interpreted as piece-wise constant at subtriangles. In quadratic grids, the velocity field and streamlines are determined by a generalized Pollock's method in subquadrilaterals.

Another strategy to determine the normal continuous flux at grid block interfaces is to apply the mixed finite element method (Ewing 1983, Brezzi *et al.* 1985, Brezzi and Fortin 2012). Based on this numerical scheme, Matringe *et al.* (2006) developed a high-order streamline tracing method in two dimensional triangular and quadrilateral grid systems. The velocity approximation function applied in this method can induce stream function that is suitable for streamline tracing. They concluded that for the same computational

cost, the high-order tracing method is more accurate and less sensitive to grid distortion than the low-order tracing method. This method is very robust in streamline tracing, yet has limitations. This streamline tracing is limited to two-dimensional grid blocks; and the velocity interpretation methods applied in this high-order method lack physical significance in pressure distribution approximations.

This high-order streamline tracing method (Matringe *et al.*, 2006) can adapt to the first-order finite difference scheme by the "slope limiter" technique (Matringe and Gerritsen, 2004). This technique interprets the slope of the normal velocity at grid block edges. This technique is easy to apply and significantly improves the computational speed. However, it is a discontinuous approximation for pressure gradient in grid block edges, which may not always be true especially when the pressure distribution is smooth.

The finite volume method is another option for complex geometry reservoir simulations. One major advantage of this method over the others is that it can be applied to both structured and unstructured grid blocks and allows for effective gridding in complex geometries. Zhang *et al.* (2012) systematically discussed and analyzed several velocity interpretation methods for streamline tracing in unstructured grid blocks. They recommended a lower-order locally conservative method for the most robust and numerically efficient calculation of streamline trajectories on unstructured grid blocks.

Johansen (2010) proposed a new streamline tracing concept that applies piece-wise polynomial pressure approximation functions in grid blocks to derive the velocity fields that is suitable for streamline tracing. More specifically, a semi-analytical streamline tracing method in both Cartesian and Polar coordinate systems using two- and three-dimensional regular structured grid blocks was proposed. Bilinear (for two-dimensional grid block) and trilinear (for three-dimensional grid block) pressure functions instead of piece-wise constant pressure distributions are applied in grid blocks by considering that pressure is continuously distributed in the reservoir, whereas velocity can exhibit discontinuous behavior. This bilinear pressure function is differentiable, thus the corresponding explicit algebraic expressions

for streamline trajectory and the time-of-flight can be derived.

Based on the same concept of tracing streamlines using pressure approximation functions, several higher-order polynomial pressure functions for streamline tracing purposes are considered and discussed in this research thesis. Through investigations, a Cubic method is developed to provide more accurate streamline tracing results in both two- and three-dimensional grid blocks.

2.2 The Riemann approach

Streamline or streamtube based methods solve the transport problem by mapping the one-dimensional solutions for mass conservation equations along each streamline or streamtube. Solving the mass conservation equations for the two-phase immiscible displacement process with constant boundary conditions is a Riemann problem. The analytical one-dimensional Riemann solution is given by Buckley-Leverett theory (1942) when the well rates are constant and the initial conditions in the reservoir are uniform. Mapping the one-dimensional Riemann solution along the streamline or streamtubes is called the *Riemann approach* (Thiele, 1994). A detailed description for Riemann approach based on the Buckley and Leverett (1942) fractional flow theory is presented in Appendix B. A brief summary of the reviewed literature in this research area is shown in Table 2.2.

Higgins and Leighton (1962a, 1962b, and 1964) were the first to apply the Riemann solution from the Buckley-Leverett theory (1942) to model nonlinear displacement in homogeneous, areal domains for several regular well patterns using streamtubes. In these three papers, Higgins and Leighton systematically discussed the streamtube simulations for two-phase flow (1962a) and three-phase flow (1962b), and the influence of several different well patterns (1964).

Higgins and Leighton (1962a) generated ten fixed streamtubes to simulate a two-phase

Table 2.2: Riemann approach literature summary

Year	Author(s)	Main Contributions
1962a	Higgins and Leighton	Applied the Riemann solution from Buckley-Leverett (1942) to solve transport problem using streamtubes
1962b	Higgins and Leighton	Solved three-phase flow transport problem using streamtubes
1964	Higgins and Leighton	Solved transport problem for different well patterns using streamtubes
1972	Parsons	Solved transport problem for permeability anisotropic porous medium using streamtubes
1973	Martin <i>et al.</i>	Discussed why the streamtube method fails to model the displacement processes with favorable mobility ratios
1979	Bommer and Schechter	Solved the transport problem along streamlines through a finite-difference approach
1979	Martin and Wegner	Introduced a numerical streamtube method for reservoir simulation
1981	Glimm <i>et al.</i>	Presented the front tracking method for solving the transport problems which can be applied along streamtubes/streamlines
1993	Bratvedt <i>et al.</i>	Solved the three-dimensional transport problem along streamlines using a front tracking method
1996	Bratvedt <i>et al.</i>	Accounts for gravity effects in streamline simulation
1996	Thiele	Solved nonlinear transport problems using streamtube method
2016	Johansen and James	Developed the analytical Riemann solutions for constant pressure boundaries
2016	Johansen and Liu	Solved the transport problems under constant pressure boundaries using analytical streamtube method

displacement process in a homogeneous quarter-of-a-five-spot pattern. To account for the changing mobility ratio as the displacement proceeds, the resistance within each tube was

updated at the end of each time step. The amount of fluid injected into each streamtube was then allocated proportionally to the ratio of the resistance of each streamtube to the total resistance of the system. Higgins and Leighton showed excellent agreement with laboratory waterflood data reported by Douglas *et al.* (1959) for viscosity ratios ranging over almost four orders of magnitude (0.1-1000).

Parsons (1972) estimated the effects of permeability anisotropy on oil production using streamtubes. The author showed that neglecting anisotropy might lead to erroneous interpretations of pilot projects or to bad prediction of field-wide projects using a streamline model. In this paper, the time-of-flight coordinate is applied to define a flood front where various streamlines have equal time-of-flights. This is the first time the concept of time-of-flight is introduced.

An important note was published by Martin *et al.* (1973) discussing why the streamtube method successfully models unfavorable mobility ratios, but fails for favorable mobility ratios. They argued that the fundamental assumption of fixed streamtubes is not valid when the mobility ratio is less than one. They found that the streamtube approach underestimates the recovery for favorable mobility ratios. Most of the pressure drop between the injection and production wells occurs in the “watered-out” region (or the low-mobility region). The streamlines in the “watered-out” region are almost independent of the unswept area. They suggested that a better solution can be obtained by updating the tubes several times as the flood progresses.

Bommer and Schechter (1979) solved the mass conservation equations for multi-component flow along streamlines through a finite-difference approach. Using this approach, they were able to account for chemical reactions and physical diffusion in the main direction of flow, thereby modeling the relevant physics of the leaching process.

Martin and Wegner (1979) presented a numerical streamtube method. They further studied the fixed streamtube approach by considering mobility ratios ranging from 0.1 to 1000 and

concluded that this approach is satisfactory for most two-phase problems and involves much less mathematics and computations than variable-tube methods.

Glimm *et al.* (1981) presented the front tracing method which can be applied along stream-tubes/streamlines. In this method, a rarefaction wave is discretized into a series of fronts, and each of these fronts are moved and predicted independently in multiple dimensions. Gravity was accounted for in the front tracking method by operator splitting in the vertical direction (Glimm *et al.*, 1983).

Bratvedt *et al.* (1993) presented a similar front tracking method along streamlines as that of Glimm *et al.* (1981). A one-dimensional equation along streamlines was developed to solve for the saturations and the new front location in the current time step. The method for solving the saturation equations is based on the piece-wise linear approximations of the fractional flow function. This method allows the saturation equations to be solved without stability problems and restrictions on the time step length.

Bratvedt *et al.* (1996) presented a streamline simulation method that accounts for gravity using the concept of operator splitting (Glimm *et al.* 1983). When gravity is considered, the phase velocities are not aligned with the total velocities. Therefore, phase components are not moving along the streamlines. Instead, the operator splitting method solves the mass conservation equation in two steps, the convective step followed by the gravity step. More specifically, it first solves the conservation equation along the streamlines ignoring the gravity term, and then corrects the results by considering the phase density differences along the gravity (vertical) direction.

Thiele *et al.* (1996) used the streamtube approach to find rapid and accurate solutions to the more difficult nonlinear problems in multiphase flow. They abandoned the idea of calculating tube resistances from the Higgins and Leighton method (1962a) because this method failed to properly model highly nonlinear displacements. Instead, they allowed the streamtube geometries to change with time and solved the transport problem by the

Riemann approach. This method can accurately predict the breakthrough performance for highly nonlinear displacements. The major assumption of mapping analytical solutions to recalculated paths is that the streamtube paths do not change greatly from time step to time step. Additionally, because streamtube paths remained relatively fixed, very large time steps could be taken yet still capture the displacement nonlinearities.

Previous analytical Riemann solution is only available for cases under constant flow rate (Buckley and Leverett 1942) and not for constant pressure boundaries. The analytical Riemann solutions under constant pressure boundaries were recently published by Johansen and James (2016) where the total velocity as a function of time was explicitly determined in an algorithmic fashion. Furthermore, Johansen and Liu (2016) analytically solved the three-dimensional Riemann problem in streamtubes with known cross sectional area. This analytical method has broadened the applications of analytical Riemann approach along streamtubes.

There were some attempts to calculate streamtubes in three dimensions but it is significantly more complicated than in two dimensions. Instead, the tracing of streamlines in a three-dimensional system is very efficient by following Pollok's algorithm (1988) or the new semi-analytical streamline tracing methods developed in this research thesis. The transport problem is preferentially solved along streamlines instead of streamtubes.

Considering above facts, one of the research scopes in this thesis is to consider the semi-analytical Riemann approach along streamlines instead of streamtubes for solving the transport problem in waterflooding displacement under constant pressure boundaries. This approach determines flow rate as a function of time, and saturation profiles along streamlines in terms of time-of-flight. The most favorable feature of the semi-analytical approach compared to the conventional numerical approach is that it leads to less numerical error and save considerable calculation time. Comparing to the analytical Riemann approach in streamtubes (Johansen and Liu, 2016), this semi-analytical approach is sufficiently accurate, and is more computational efficient.

2.3 Waterflooding Experiments and Simulations

In most oil reservoirs, permeability and porosity vary from location to location. These variations are known as heterogeneities and occur at all scales from kilometers down to microns. If the simulator fails to capture the fluid flow affected by reservoir heterogeneity, it may lead to erroneous results. Therefore, the numerical and experimental studies of displacement process in heterogeneous porous medium are important.

The visualization experiments performed to study the complex displacement characterizations and mass transfer phenomena are also rich in the petroleum engineering literature. Flow behavior in heterogeneous medium can also be visualized and studied by laboratory displacement experiments on well-defined heterogeneous artificial models. A successful numerical simulator can predict the flow performance for these experiments accurately and account for the dominant forces in displacements. A brief summary of the reviewed literature is shown in Table 2.3.

Brock and Orr (1991) performed series of flow visualization experiments and numerical simulations to investigate the combined effects of viscous fingering and permeability heterogeneity in four different macro-scale ($30 \text{ cm} \times 7.6 \text{ cm} \times 0.57 \text{ cm}$) two-dimensional glass bead packs. Glass beads of uniform size were used to create the homogeneous model. For heterogeneities such as layering, beads of uniform size were used within a layer, with different bead sizes for each layer. In each model, unstable displacements were performed at three different flow rates and mobility ratios. They concluded that the viscous crossflow drives the finger growth mechanisms in the homogeneous model, while in the heterogeneous models, flow was largely determined by the patterns of heterogeneity, and fingering patterns develop along the streamlines. The experiments were also simulated numerically, using a particle tracking simulator. Simulations yielded finger patterns very similar to those seen in the experiments. Thus, the simulator used represents with reasonable accuracy the physics of finger growth in the heterogeneous porous medium.

Table 2.3: Experimental and numerical studies on waterflooding literature summary

Author(s) (year)	Main Contributions	Porous medium patterns ($k_1 > k_2 > k_3 > k_4$)
Brock and Orr (1991)	Performed flow visualization experiments and numerical simulations to investigate the combined effects of viscous fingering and permeability heterogeneity in four different two-dimensional glass bead packs	
Dawe <i>et al.</i> (1992)	Reported the experimental studies of the effects of well-defined heterogeneous porous medium on immiscible flooding	
Roti and Dawe (1993)	Reported the experimental studies on the effects of layer thickness, permeability contrast, angle of layer to flow direction, viscosity ratio, and flow rate	
Huang <i>et al.</i> (1995)	Reported the experimental and numerical studies for low rate drainage/imbibition floods in cross-laminated heterogeneous sandstone	
Cinar <i>et al.</i> (2006)	Reported the numerical and flow visualization experimental studies on crossflow that occurs in two-phase displacements in layered porous medium	

Dawe *et al.* (1992) reported the results of experimental studies of the effects of well-defined heterogeneous porous medium on immiscible flooding. Drainage and imbibition displacements, with and without an initial residual fluid saturation, were carried out at a variety of flow rates. The test porous medium are two-dimensional glass beads models (58 cm \times 10 cm \times 0.6 cm). The heterogeneities were layers and lenses, with some of the lenses exhibiting a wettability contrast. More specifically, the effect of flooding rate, initial fluid saturations, and wettability on drainage and imbibition were discussed. The primary conclusions were that the capillary forces become of greater importance and can even dominate the flow, that

the balance between capillary and viscous forces is rate dependent, and that the effects of capillary forces become larger as the flow rate decreases. These conclusions coincide well with Lake (1989).

Roti and Dawe (1993) performed experiments on two dimensional glass-bead packs (25 cm \times 10 cm \times 0.6 cm) and numerical simulations (using Eclipse 100) to study the effects of layer thickness, permeability contrast, angle of layer to flow direction, viscosity ratio, and flood rate. It was shown that the geology information needs to be incorporated effectively and efficiently in reservoir simulations, and the effects of cross bedded heterogeneous should be considered in simulations.

Huang *et al.* (1995) performed experimental and numerical studies for low rate drainage/imbibition floods in cross-laminated heterogeneous sandstone (20 cm \times 10 cm \times 1 cm) and highlighted the importance of oil trapping caused by the interaction of capillary forces with small-scale reservoir heterogeneity at the lamina-set scale. Based on the results, they concluded that most of the oil is trapped in the high permeability regions upstream of low permeability layers for water-wet samples and showed an excellent match between experiment and numerical modeling using Eclipse 100.

Cinar *et al.* (2006) presented flow visualization experiments and both finite-difference and streamline simulations that demonstrated the combined effects of viscous and capillary forces and gravity segregation on crossflow that occurs in two-phase displacements in layered porous medium. They performed a series of immiscible flooding experiments in two dimensional, two-layered glass bead models (33 cm \times 8 cm \times 0.6 cm), including favorable mobility-ratio imbibition and unfavorable mobility-ratio drainage experiments. The interfacial tension was controlled by varying the concentration of components of the fluid. These immiscible flooding experiments were performed over a wide range of capillary and gravity numbers. There were three main observations: crossflow is impacted by the complex interaction effects of capillary, gravity, and viscous forces; fluid flow is dominated by capillary pressure at high interfacial tension and by gravity and viscous forces at low interfacial ten-

sion; and the transition ranges of scaling groups suggested by Zhou *et al.* (1997) are valid. The scaling groups are applied to assess the relative contribution of each driving force to flow.

In addition, Cinar *et al.* (2006) simulated the experiments by two different numerical techniques: finite-difference and streamline methods. The streamline simulator used in their research was based on the streamline simulator developed by Batycky (1997). To simulate the crossflow accurately, the effects of gravity and capillary pressure were included in both the pressure and saturation equations. In summary, the simulation results from both numerical methods agreed well with experimental observations when gravity and viscous forces were most important. However, for capillary-dominated flows, streamline simulation, even with the inclusion of a representation of capillary forces, cannot represent the flow behavior correctly; the fully implicit finite difference method was suggested as a better approach.

The primary objective of the visualization experiments in this research thesis is to validate the new streamline simulator for the waterflooding processes under constant differential pressure boundaries in a heterogeneous reservoir. In the laboratory, an unconsolidated two-dimensional (30 cm \times 30 cm \times 1 cm) glass bead macro-model is designed and fabricated. Two experiments are performed with the same macro-model and fluid properties but under different pressure boundaries. Simultaneously, the new streamline simulator is applied to history matching and direct modeling the displacement processes. The simulated and experimental results shown great agreement. Moreover, the waterfront movement under constant differential pressure boundaries are also analyzed using simulated results.

Chapter 3

Tracing Streamlines Using Continuous Pressure Approximations

3.1 Introduction

Streamline simulation is gaining research and application interests in recent decades due to its high computational efficiency and powerful visualization capability of fluid flow. In streamline simulation, the three-dimensional transport problem is decoupled to tracing streamlines and solving one-dimensional transport problem along each streamline. The pressure equation, an elliptic equation, is first solved with given boundary conditions and then streamlines are generated accordingly. These are solved once in most cases. Hence, the accuracy of pressure solution and streamline trajectories highly affects the performance of a streamline simulator.

In general, streamline tracing is usually accomplished in three steps. First, solving for the pressure at each discrete grid block using a numerical method; then, approximating the

velocity field over each grid block; and finally, integrating streamlines at each grid block using the velocity field, and calculating the time that a particle takes to travel along the streamline (time-of-flight) within the grid block (Zhang *et al.* 2012).

In standard streamline tracing methods, the pressure in each grid block is constant, and consequently, the velocity approximation functions maybe physically incorrect. Different from previous methods, the new methods introduced in this research thesis assume that the pressure distribution in each grid block is a polynomial and the velocity field is calculated accordingly.

The Bilinear (for two-dimensional problems) and Trilinear (for three-dimensional problems) streamline tracing methods presented in this chapter assume the pressure distribution is continuous everywhere in a reservoir. In these two methods, the pressure in each grid block is approximated by a bilinear/trilinear function, which satisfies the elliptic pressure equation for the grid block. Based on the bilinear pressure function, the explicit expressions for the stream function and time-of-flight are derived. The Bilinear and Trilinear streamline tracing methods have a potential to adapt to many numerical methods and various grid structures.

In this chapter, the basic concepts related to streamlines are first introduced in section 3.2. Then, the new streamline tracing methodology based on pressure approximation functions is introduced in section 3.3. Finally, in section 3.4, the Bilinear and Trilinear streamline tracing methods is introduced in two- and three- dimensional grid blocks, respectively.

3.2 Streamlines, Stream Function and Time-of-flight

Streamlines are curves that are everywhere tangent to the instantaneous velocity vector (Bear 1972). In the scope of this research thesis, streamlines are traced in steady state flow, in which the flow characteristics remain invariant with time. As shown in Figure 3.1, the

mathematical expression defining a streamline is,

$$\mathbf{v} \times d\boldsymbol{\xi} = 0, \quad (3.1)$$

where \times is the cross product; \mathbf{v} is the flow velocity vector, $\mathbf{v} = (v_x, v_y, v_z)$; $d\boldsymbol{\xi}$ is the infinitesimal arc-length of the streamline, $d\boldsymbol{\xi} = (dx, dy, dz)$. This results in the definition of a streamline as,

$$\frac{dx}{v_x} = \frac{dy}{v_y} = \frac{dz}{v_z}. \quad (3.2)$$

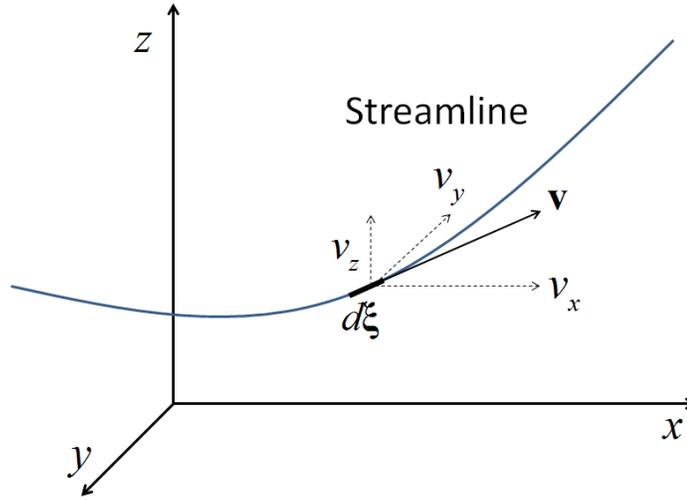


Figure 3.1: Streamline and the velocity vector

In the $x - y$ plane, the definition of a streamline Eq. 3.2 becomes,

$$\frac{dx}{v_x} = \frac{dy}{v_y} \quad \text{or} \quad v_y dx - v_x dy = 0. \quad (3.3)$$

The instantaneous geometry of the streamline can be described by,

$$\psi(x, y) = \text{constant}. \quad (3.4)$$

where $\psi(x, y)$ is the stream function, the value of the stream function is constant along a streamline (Lagrange, 1781).

Along a certain streamline, the total derivative of the stream function is,

$$d\psi = \frac{\partial\psi}{\partial x} dx + \frac{\partial\psi}{\partial y} dy = 0. \quad (3.5)$$

Combining Eq. 3.3 and Eq. 3.5 results in

$$\frac{\partial\psi}{\partial x}dx + \frac{\partial\psi}{\partial y}dy = v_y dx - v_x dy. \quad (3.6)$$

Then, the relationships for the stream function and the directional components of the flow velocity vector is revealed as,

$$v_y = \frac{\partial\psi}{\partial x}; \quad v_x = -\frac{\partial\psi}{\partial y}. \quad (3.7)$$

Therefore, the algebraic expression for the stream function can be obtained by integrating the directional components of the velocity vector,

$$\psi = \int v_y dx - \int v_x dy. \quad (3.8)$$

In this research thesis, a streamline starts from the launching point at the injector, travels through the porous medium and lands at the exit point at the producer. The time-of-flight τ is defined as the time that a particle takes to travel from its launching point (x_0, y_0, z_0) to a certain point (x, y, z) on the streamline. The time-of-flight can be expressed in terms of x , y or z , *i.e.*,

$$\tau = \int_{x_0}^x \frac{dx}{v_x} = \int_{y_0}^y \frac{dy}{v_y} = \int_{z_0}^z \frac{dz}{v_z}. \quad (3.9)$$

When calculating the time-of-flight, a parameter (x , y or z) that is monotonically changing along the streamline will be applied. This is because the directional velocity is the denominator in this equation, and it will become zero if the directional parameter is not monotonically changing along the streamline, and will induce large errors in time-of-flight calculations. If all the directional velocities become zero, the streamline is terminated at this point (stagnation point). Time-of-flight is primarily used as a spatial coordinate in streamline simulation, which means that the spatial coordinate of a point on a streamline can be represented by the time-of-flight and by the Cartesian coordinates.

3.3 Streamline Tracing Methodology

3.3.1 General steps to trace streamlines

The new streamline tracing methods introduced in this research thesis follow the concept of particle-tracking introduced by Pollock (1988) (described in section 1.1.1), the difference is, the piece-wise polynomial pressure approximation functions are used to derive the velocity approximation function. In general, in this research, a streamline is traced from an injector to a producer through grid blocks in a reservoir model. It is achieved by completing the following two steps:

1. Approximate the local velocity field in each grid block.
 - (a) Discretize the reservoir into grid blocks and assign physical properties to these grid blocks;
 - (b) Apply a numerical method to solve for the pressure at certain nodes in the grid block with given global boundary conditions;
 - (c) Choose an appropriate polynomial to approximate the pressure distribution in each grid block based on the pressure obtained from (1b);
 - (d) Derive or approximate the velocity field in each grid block based on the pressure distribution obtained from step (1c);
2. Trace a streamline from injector to producer.
 - (a) Specify a launching point of the fluid particle from the injector grid block interface to start tracing the corresponding streamline.
 - (b) Determine the streamline in the current grid block.
 - (c) Determine the exit point of the particle and the incremental time-of-flight in the current grid block according to the local velocity field obtained from step (1d).

- (d) Move to the next adjacent downstream grid block. Treat the exit point on the previous grid block as the entry point on the current grid block.
- (e) Repeat steps (2b) to (2d) until the fluid particle reaches the producer grid block.

Step 1 and 2 describe the procedures to trace one particular streamline. Given various launching points, various streamlines are then traced following the same procedures. The number of streamlines are generally based on the requirements of accuracy. Along each streamline, the results of intersection points and the time-of-flight in each grid block are stored. These two parameters are also used to solve the transport problem along streamlines in multi-phase flow.

Based on above discussions, we can conclude that the streamline tracing is grid block based, therefore, it is important to understand the new tracing algorithm at grid block level. In this research, the new streamline methods are introduced at grid block level. More specifically speaking, the pressure, velocity, streamfunction, and time-of-flight determination methods are introduced in each grid block.

In summary, the main objective in the new proposed methods are to generate more refined velocity fields, hence leading to more accurate streamline tracing results. This is achieved by introducing polynomial pressure functions that satisfy the Laplace equation locally. In the Bilinear and Trilinear methods, the full continuity of pressure is achieved everywhere. Details are described in the following sections.

3.3.2 Pressure and velocity approximation using polynomial functions

Streamlines are tangential to the instantaneous velocity field. Thus, to accurately approximate the local velocity field is essential for accurate streamline tracing in the grid blocks. In streamline simulation, the imposed pressure is the driving force for fluid flow, *i.e.* the velocity field is determined from the pressure gradient. Therefore, in order to achieve more accurate streamline tracing results, the new streamline tracing methods introduced in this

research thesis approximate the pressure distribution with piece-wise polynomial pressure functions, and then derives the velocity functions for the local velocity field. Based on these results, the streamline trajectory function and time-of-flight are derived and calculated sequentially.

To choose appropriate pressure and velocity approximation functions, several governing equations must be considered. Therefore, consider a model for single-phase flow in porous medium subject to the following assumptions: fluid and porous medium are incompressible, gravity and capillary effects are negligible, and flow is steady-state.

When the grid block has no injector or producer (source or sink term), the velocity field is divergence free, *i.e.*,

$$\nabla \cdot \mathbf{v} = 0. \quad (3.10)$$

The fluid flow in the porous medium obeys Darcy's Law, *i.e.*,

$$\mathbf{u} = -\frac{\mathbf{K}}{\mu} \nabla P, \quad (3.11)$$

where \mathbf{u} is Darcy velocity, \mathbf{K} is permeability tensor, μ is fluid viscosity and P is pressure.

The relationship between the fluid flow velocity \mathbf{v} and the Darcy velocity \mathbf{u} is,

$$\mathbf{v} = \frac{\mathbf{u}}{\phi}, \quad (3.12)$$

where ϕ is porosity.

Combining the mass conservation law (Eq. 3.10) and Darcy's law (Eq. 3.11) gives the Laplace's equation,

$$\nabla \cdot \left(\frac{\mathbf{K}}{\phi\mu} \nabla P \right) = 0. \quad (3.13)$$

Therefore, we define the four requirements that any pressure and velocity approximation functions must satisfy. These four requirements are:

1. The velocity approximation function must obey the velocity field divergence free con-

dition (Eq. 3.10);

2. The velocity approximation function is determined from Darcy's Law (Eq. 3.11) given the pressure approximation function;
3. The pressure approximation function must satisfy the Laplace equation (Eq. 3.13).
4. The pressure and velocity functions are given by explicit functions.

The streamline tracing method is semi-analytical if an explicit algebraic formulation of the stream function can be obtained. This is not a necessary condition for choosing the polynomials for pressure approximation, but it leads to less simulation errors when this condition is satisfied.

In this research, both second and third degree pressure polynomials are considered. Higher degree polynomials are not considered due to the cumbersomeness and difficulties in determining a large number of pressure coefficients. These pressure methods and the associated velocity field need to satisfy the requirements presented above. Generally, the Bilinear, Trilinear, and Cubic (Chapter 4) streamline tracing methods are introduced in this research thesis for tracing streamlines. In the next section, the Bilinear and Trilinear streamline tracing methods is introduced in two- and three-dimensional grid blocks, respectively.

3.4 The Streamline Tracing Methods Based on Continuous Pressure Approximations

Assuming pressure is continuously distributed in the reservoir, the Bilinear (for two-dimensional problems) and Trilinear (for three-dimensional problems) streamline tracing methods are developed. These two methods were initially proposed by Johansen (2010) for both Cartesian and Polar grid blocks. The author of this research thesis participated in the development of these two methods in Cartesian grid blocks. The applications of the Bilinear and Tri-

linear methods in Polar coordinates, mainly used for near-well region simulations, are not considered in this research thesis.

In brief, there are five steps for tracing streamlines in a grid block using the Bilinear or Trilinear methods:

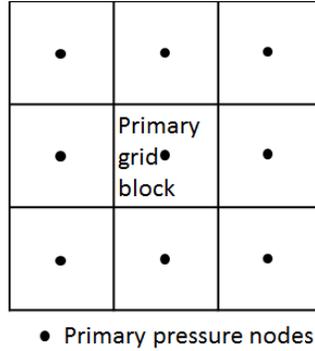
1. Divide the primary grid block into sub-cells and construct dual-cell system in the reservoir;
2. Determine the pressures at the sub-cell vertexes;
3. Approximate the pressure distribution in the sub-cells using the Bilinear (two-dimensional grid block) or the Trilinear (three-dimensional grid block) function;
4. Derive the velocity field in the sub-cells directly from the pressure functions;
5. Solve for the geometry of streamlines, and calculate the time-of-flight in each sub-cell.

The Bilinear method is semi-analytical, since in two-dimensional grid blocks, streamlines are defined by a closed formula derived from the pressure approximation function. In three-dimensional grid blocks, the Trilinear method generates streamlines by numerically solving two coupled first-order ordinary differential equations using the Runge-Kutta 4th order method (Appendix C). The methodologies of these two methods are presented in this section, calculation examples and discussions are given in Chapter 5.

3.4.1 The Bilinear Streamline Tracing Method

As discussed in section 3.3.2, the velocity field approximation is essential for accurate streamline tracing. The emphasis of this sub-section is to introduce the Bilinear method in approximating the pressure distribution, velocity field, and then tracing streamlines within each two-dimensional grid block. The grid block discretization and the computation of pressure values at *primary nodes* (as seen in Figure 3.2) are assumed to be already completed. A

detailed description for solving the pressures using a finite difference method can be seen in Appendix A.



Following the finite difference framework, the fluid and rock properties are assigned to each sub-quadrilateral as constant parameters. As shown in the Figure 3.3, the pressures at the primary nodes are known, denoted by P_1, P_2, P_3, P_4 ; the pressures at primary grid block interfaces are newly introduced pressures, denoted by P_n, P_s, P_e, P_w, P_m . The sub-quadrilaterals are numbered as 1, 2, 3 and 4. This defines the dual-cell system.

In each sub-quadrilateral grid block, the pressure distribution can be approximated by a unique bilinear equation if the pressure values at its vertexes are known,

$$P(x, y) = P_0 + A_1x + A_2y + Bxy. \quad (3.14)$$

where P_0, A_1, A_2, B are constant pressure coefficients in the sub-quadrilateral. This is the bilinear pressure function used in the Bilinear method, and it satisfies the Laplace equation 3.13 in each sub-quadrilateral.

Therefore, the pressure distribution can be approximated if the pressures at vertexes are known. Next, the newly introduced pressures are solved by imposing normal flux continuities across primary grid block interfaces and velocity field divergence free conditions. Since five pressure unknowns are introduced, five equations are needed. It is worth noting that the continuous Darcy-flux finite volume methods with full pressure support (Edwards and Zheng, 2008) applied a similar method which was proven relatively robust in minimizing spurious oscillations in the discrete pressure solutions.

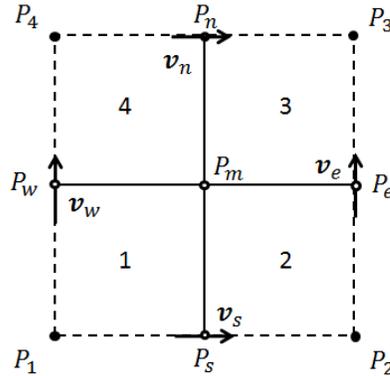


Figure 3.4: The normal fluxes across the primary grid block interfaces (w, e, n, s)

In the dual-cell shown in Figure 3.4, the four flux continuity equations across the primary grid block interfaces (w, e, n, s) are given by Darcy's law,

$$v_w = \frac{K_{1,y}}{\phi\mu} \left(\frac{P_1 - P_w}{y_w - y_1} \right) = \frac{K_{4,y}}{\phi\mu} \left(\frac{P_w - P_4}{y_4 - y_w} \right); \quad (3.15)$$

$$v_e = \frac{K_{2,y}}{\phi\mu} \left(\frac{P_2 - P_e}{y_e - y_2} \right) = \frac{K_{3,y}}{\phi\mu} \left(\frac{P_e - P_3}{y_3 - y_e} \right); \quad (3.16)$$

$$v_n = \frac{K_{4,x}}{\phi\mu} \left(\frac{P_4 - P_n}{x_n - x_4} \right) = \frac{K_{3,x}}{\phi\mu} \left(\frac{P_n - P_3}{x_3 - x_n} \right); \quad (3.17)$$

$$v_s = \frac{K_{1,x}}{\phi\mu} \left(\frac{P_1 - P_s}{x_s - x_1} \right) = \frac{K_{2,x}}{\phi\mu} \left(\frac{P_s - P_2}{x_2 - x_s} \right); \quad (3.18)$$

where, v_i are the flux across the primary grid block interface i ($i = w, e, n, s$); $K_{subcell,x}$ and $K_{subcell,y}$ is the x - and y - directional permeability in the sub-quadrilateral ($subcell = 1, 2, 3, 4$), respectively; x_{vertex} and y_{vertex} is the x - and y - coordinates of the vertex ($vertex = 1, 2, 3, 4, w, e, n, s$), respectively.

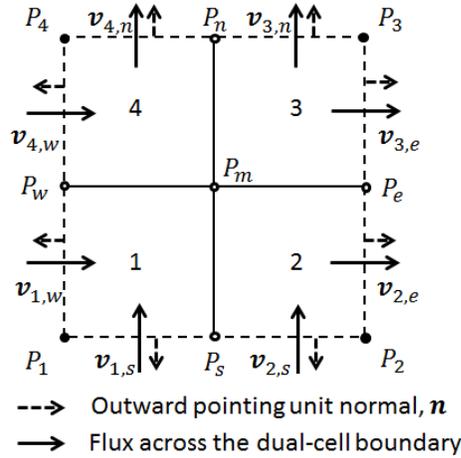


Figure 3.5: The normal fluxes at the dual-cell boundaries

Furthermore, for incompressible flow, the velocity field away from the source or sink term is divergence free. The volume integral of the velocity field \mathbf{v} over the entire region of the dual-cell Ω is,

$$\iint_{\Omega} \nabla \cdot \mathbf{v} dV = 0, \quad (3.19)$$

where V is the volume of the dual-cell. According to Gauss's Theorem in two-dimensions,

we have

$$\iint_{\Omega} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0, \quad (3.20)$$

where the closed manifold $\partial\Omega$ is the boundary of Ω , \mathbf{n} is the outward pointing unit normal of the boundary $\partial\Omega$, and Γ is the parameterization of boundary.

Since the flux varies linearly along the x - and y -axes, as shown in Figure 3.5, Eq. 3.20 becomes,

$$\begin{aligned} \int_{\partial\Omega} \mathbf{v} \cdot \mathbf{n} d\Gamma &= \mathbf{v}_{1,s} \cdot \mathbf{n}_{1,s} \Delta\Gamma_{1,s} + \mathbf{v}_{2,s} \cdot \mathbf{n}_{2,s} \Delta\Gamma_{2,s} + \mathbf{v}_{2,e} \cdot \mathbf{n}_{2,e} \Delta\Gamma_{2,e} + \mathbf{v}_{3,e} \cdot \mathbf{n}_{3,e} \Delta\Gamma_{3,e} \\ &+ \mathbf{v}_{3,n} \cdot \mathbf{n}_{3,n} \Delta\Gamma_{3,n} + \mathbf{v}_{4,n} \cdot \mathbf{n}_{4,n} \Delta\Gamma_{4,n} + \mathbf{v}_{4,w} \cdot \mathbf{n}_{4,w} \Delta\Gamma_{4,w} + \mathbf{v}_{1,w} \cdot \mathbf{n}_{1,w} \Delta\Gamma_{1,w} = 0. \end{aligned} \quad (3.21)$$

Since the grid block is regular by design, the velocity vectors and the outward unit normal in the same or the opposite directions, Eq. 3.21 can be simplified as,

$$(-v_{1,s} - v_{2,s} + v_{3,n} + v_{4,n}) \Delta x + (v_{2,e} + v_{3,e} - v_{4,w} - v_{1,w}) \Delta y = 0 \quad (3.22)$$

where, $v_{1,s}$ is the velocity module of the velocity vector $\mathbf{v}_{1,s}$, Δx and Δy are the x - and y - directional length of the grid block.

These fluxes at the midpoint can be determined using Darcy's law in terms of the vertex pressures and coordinates. For example, the flux across the boundary $(1, s)$ is given as an average by,

$$v_{1,s} = \frac{K_{1,y}}{2\phi\mu} \left(\frac{P_w - P_1}{y_w - y_1} + \frac{P_m - P_s}{y_m - y_s} \right), \quad (3.23)$$

where, $K_{1,y}$ is the y - directional permeability in the sub-quadrilateral 1; y_{vertex} is the y - coordinates of the vertex ($vertex = 1, s, m, w$) respectively.

In summary, the five equations constructed for each dual-cell system are Eq. 3.15, 3.16, 3.17, 3.18 and 3.22. Hence, the newly introduced pressures P_n, P_s, P_e, P_w, P_m are determined by solving the linear 5×5 system of equations.

Given the pressure values at each sub-quadrilateral vertex, the bilinear pressure distribution function at each sub-quadrilateral is then uniquely determined. Recall that the pressure

distribution is approximated by a bilinear equation at each sub-quadrilateral (Eq. 3.14),

$$P(x, y) = P_0 + A_1x + A_2y + Bxy.$$

where P_0, A_1, A_2, B are coefficients. These coefficients, for example in the sub-quadrilateral 1, are determined by,

$$\begin{bmatrix} A_1 \\ A_2 \\ B \\ P_0 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & x_1y_1 & 1 \\ x_s & y_s & x_sy_s & 1 \\ x_m & y_m & x_my_m & 1 \\ x_w & y_w & x_wy_w & 1 \end{bmatrix}^{-1} \begin{bmatrix} P_1 \\ P_s \\ P_m \\ P_w \end{bmatrix}, \quad (3.24)$$

where, x_{vertex}, y_{vertex} are coordinates of the vertex; and P_{vertex} is the vertex pressure ($vertex = 1, s, m, w$).

The corresponding velocity field at each sub-quadrilateral can be derived according to Darcy's Law,

$$v_x = -\frac{K_x}{\phi\mu} \frac{\partial P}{\partial x} = -\frac{K_x}{\phi\mu} (A_1 + By) = a_1 + b_1y; \quad (3.25)$$

$$v_y = -\frac{K_y}{\phi\mu} \frac{\partial P}{\partial y} = -\frac{K_y}{\phi\mu} (A_2 + Bx) = a_2 + b_2x; \quad (3.26)$$

where K_x, K_y are the principal permeabilities; ϕ is porosity and μ is fluid viscosity; and a_1, b_1, a_2, b_2 are velocity coefficients, *i.e.*,

$$a_1 = -\frac{K_x A_1}{\phi\mu}; \quad b_1 = -\frac{K_x B}{\phi\mu}; \quad a_2 = -\frac{K_y A_2}{\phi\mu}; \quad b_2 = -\frac{K_y B}{\phi\mu}. \quad (3.27)$$

It is straightforward to verify that the velocity field in each sub-quadrilateral is mass conservative, *i.e.*,

$$\nabla \cdot \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0. \quad (3.28)$$

Knowing the velocity field, the stream function can be derived. Substituting Eq. 3.25 and 3.26 into Eq 3.8 yields,

$$\psi = -\int v_y dx + \int v_x dy = a_1y - a_2x + \frac{b_1}{2}y^2 - \frac{b_2}{2}x^2. \quad (3.29)$$

Giving an arbitrary entry point (x_1, y_1) , the geometry of the corresponding streamline in the sub-quadrilateral is given by the stream function,

$$a_1y - a_2x + \frac{b_1}{2}y^2 - \frac{b_2}{2}x^2 = \psi(x_1, y_1); \quad (3.30)$$

where $\psi(x_1, y_1) = a_1y_1 - a_2x_1 + \frac{b_1}{2}y_1^2 - \frac{b_2}{2}x_1^2$ is a constant.

Rearranging the above equation 3.30 into a form of hyperbola we get,

$$\left(y + \frac{a_1}{b_1}\right)^2 - \frac{b_2}{b_1} \left(x + \frac{a_2}{b_2}\right)^2 = \tilde{C}; \quad (3.31)$$

where $\tilde{C} = \frac{2\psi(x_1, y_1)}{b_1} + \frac{a_1^2}{b_1^2} - \frac{a_2^2}{b_1 b_2}$ is a constant. If $b_1 \neq 0$ and $b_2 \neq 0$, Eq. 3.31 is a hyperbola equation, since $\frac{b_2}{b_1} = \frac{K_y}{K_x} > 0$ (derived from Eq. 3.27).

If $\tilde{C} > 0$ the streamline is a hyperbola with transverse axis parallel to the y - axis, therefore x is used as a parameter. The explicit formula for the streamline in terms of x is,

$$y = \frac{-a_1 + n_x \sqrt{a_1^2 + 2a_2b_1x + b_1b_2x^2 + 2b_1\psi(x_1, y_1)}}{b_1}; \quad (3.32)$$

where, $n_x = \pm 1$ is determined by $y_1b_1 + a_1 = n_x \sqrt{a_1^2 + 2a_2b_1x_1 + b_1b_2x_1^2 + 2b_1\psi(x_1, y_1)}$.

Otherwise, if $\tilde{C} < 0$ the streamline is a with transverse axis parallel to the x - axis and y is used as a parameter. In this case, the explicit formula for the streamline in terms of y is,

$$x = \frac{-a_2 + n_y \sqrt{a_2^2 + 2a_1b_2y + b_1b_2y^2 - 2b_2\psi(x_1, y_1)}}{b_2}; \quad (3.33)$$

where, $n_y = \pm 1$ is determined by $x_1b_2 + a_2 = n_y \sqrt{a_2^2 + 2a_1b_2y_1 + b_1b_2y_1^2 - 2b_2\psi(x_1, y_1)}$.

Then, the exit point (x_2, y_2) of the streamline in the sub-quadrilateral can be determined by solving for the intersection point between the streamline and the sub-quadrilateral boundary.

Finally, the incremental time-of-flight $\Delta\tau$ for the streamline interval in this sub-quadrilateral is determined. Assuming x is monotonically changing along the streamline (when $\tilde{C} > 0$), the incremental time-of-flight can be calculated in terms of x ,

$$\Delta\tau = \int_{x_1}^{x_2} \frac{dx}{v_x} = \int_{x_1}^{x_2} \frac{dx}{a_1 + b_1y}. \quad (3.34)$$

Substituting Eq. 3.32 into Eq. 3.34 yields the explicit expression of incremental time-of-

flight, *i.e.*,

$$\begin{aligned}\Delta\tau &= \int_{x_1}^{x_2} \frac{dx}{n_x \sqrt{a_1^2 + 2a_2b_1x + b_1b_2x^2 + 2b_1\psi(x_1, y_1)}} \\ &= \frac{1}{n_x \sqrt{A_x}} \ln \left(\frac{\frac{B_x + 2A_x x_2}{2\sqrt{A_x}} + \sqrt{A_x x_2^2 + B_x x_2 + C_x}}{\frac{B_x + 2A_x x_1}{2\sqrt{A_x}} + \sqrt{A_x x_1^2 + B_x x_1 + C_x}} \right); \end{aligned} \quad (3.35)$$

where A_x, B_x, C_x can be related to the velocity coefficients and the constant value of stream function $\psi(x_1, y_1)$ by,

$$A_x = b_1b_2; \quad B_x = 2a_2b_1; \quad C_x = a_1^2 + 2b_1\psi(x_1, y_1); \quad (3.36)$$

Following the similar procedure, the incremental time-of-flight $\Delta\tau$ can be calculated in terms of y (when $\tilde{C} < 0$),

$$\begin{aligned}\Delta\tau &= \int_{y_1}^{y_2} \frac{dy}{v_y} = \int_{y_1}^{y_2} \frac{dy}{n_y \sqrt{a_2^2 + 2a_1b_2y + b_1b_2y^2 - 2b_2\psi(x_1, y_1)}} \\ &= \frac{1}{n_y \sqrt{A_y}} \ln \left(\frac{\frac{B_y + 2A_y y_2}{2\sqrt{A_y}} + \sqrt{A_y y_2^2 + B_y y_2 + C_y}}{\frac{B_y + 2A_y y_1}{2\sqrt{A_y}} + \sqrt{A_y y_1^2 + B_y y_1 + C_y}} \right); \end{aligned} \quad (3.37)$$

where A_y, B_y, C_y are given by,

$$A_y = b_1b_2; \quad B_y = 2a_1b_2; \quad C_y = a_2^2 - 2b_2\psi(x_1, y_1); \quad (3.38)$$

Following the algorithm introduced above, the exit coordinate and the incremental time-of-flight for the streamline with given entry point have been determined in the regular two-dimensional grid blocks. This describes the main procedures to trace a streamline using the Bilinear method.

In short, the key principles of the Bilinear method are to divide the primary grid block into sub-quadrilaterals, approximate the bilinear pressure distribution, and then derive the velocity field using Darcy's law in each sub-quadrilateral. The application of the two-dimensional Bilinear streamline tracing method is neither limited to the finite difference framework nor the regular grid block structure. As shown in Figure 3.6, quadrilateral, triangular or polygonal grid blocks can all be divided into sub-quadrilaterals by connecting

the primary nodes with the edge midpoints (Figure 3.6a and 3.6b), or by locating the centroids of the original grid blocks and connecting them with the edge midpoints (Figure 3.6c). The applications of the Bilinear method in irregular grid blocks is beyond the scope of this research thesis, therefore, they are not further demonstrated.

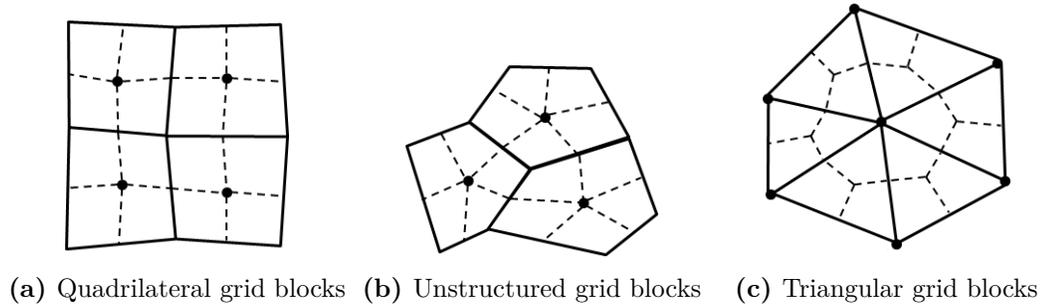


Figure 3.6: Dual-cell system in different grid structures

3.4.2 The Trilinear Streamline Tracing Method

In this sub-section, the Trilinear method is introduced to trace streamlines in three-dimensional regular grid blocks. Similar concepts introduced in subsection 3.4.1 are also applied here, since both of the Bilinear and Trilinear methods assume the pressure distribution in the reservoir is globally continuous. The grid block discretization and the pressure values at primary nodes calculation are assumed to be already completed. An example of primary pressure nodes and regular grid blocks in the reservoir is shown in Figure 3.7.

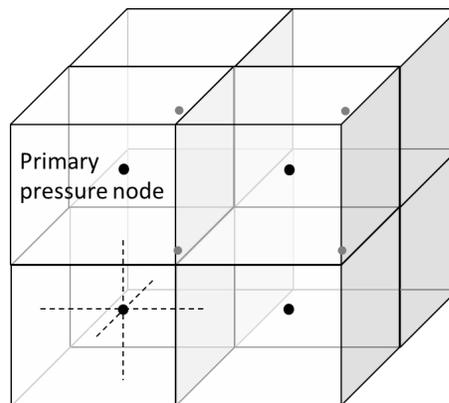


Figure 3.7: An example of the primary pressure nodes and grid blocks in a 3D reservoir

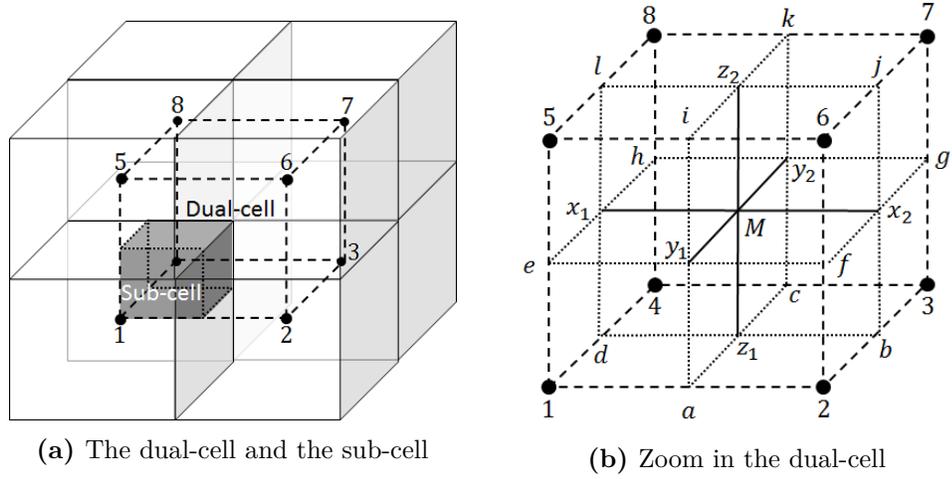


Figure 3.8: Illustration of a dual-cell and the sub-cells

To approximate the pressure distribution in each grid block, the primary grid block can be divided into sub-hexahedrals to construct a dual-cell system, as shown in Figure 3.8. The dual-cells are constructed by connecting the primary nodes with grid block interface midpoints. Assuming the pressure varies linearly along the interfaces of sub-hexahedrals, the pressure distribution can be approximated by a unique trilinear equation if the pressure values at its vertexes are known,

$$P = P_0 + A_1x + A_2y + A_3z + B_1xy + B_2xz + B_3yz + Cxyz, \quad (3.39)$$

where $P_0, A_1, A_2, A_3, B_1, B_2, B_3, C$ are constant pressure coefficients in the sub-cell. Therefore, the full pressure continuity is achieved everywhere in the reservoir.

As shown in Figure 3.8, the dual-cell system is outlined by primary pressure nodes 1 to 8 connected by dashed lines; the solid lines in Figure 3.8a are the primary grid block boundaries; and the dotted lines are the intersections between the dual-cell boundaries and the primary grid block boundaries. Following the finite difference framework, the fluid and rock properties are assigned to each sub-hexahedral as constant parameters. The pressures at primary nodes are known, denoted by $P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8$. There are nineteen newly introduced pressures, the pressures at primary grid block interface centers are denoted by $P_a, P_b, P_c, P_d, P_e, P_f, P_g, P_h, P_i, P_j, P_k, P_l$; the pressures at grid block edge midpoints are

denoted by $P_{x_1}, P_{x_2}, P_{y_1}, P_{y_2}, P_{z_1}, P_{z_2}$; and the pressures at the dual-cell center is denoted by P_M . The full pressure continuum is achieved when the unknown pressures are solved uniquely. In this Bilinear method, these nineteen newly introduced pressures are determined by twelve normal flux continuity equations (Eq. 3.40 to 3.51), plus seven divergence free equations (Eq. 3.52 to 3.57 plus Eq. 3.61) over auxiliary finite volumes.

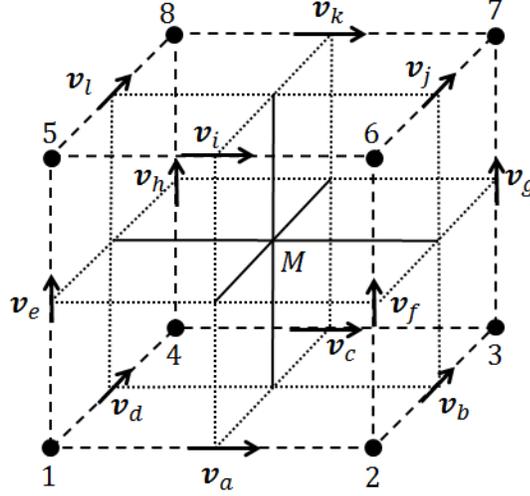


Figure 3.9: The normal fluxes across the primary grid block interfaces (a, b, \dots, l)

As shown in Figure 3.9, the twelve normal flux continuity equations at the centers of primary grid block interfaces ($a, b, c, d, e, f, g, h, i, j, k, l$) can be written,

$$v_a = \frac{K_{1,x}}{\phi\mu} \left(\frac{P_1 - P_a}{x_a - x_1} \right) = \frac{K_{2,x}}{\phi\mu} \left(\frac{P_a - P_2}{x_2 - x_a} \right); \quad (3.40)$$

$$v_b = \frac{K_{2,y}}{\phi\mu} \left(\frac{P_2 - P_b}{y_b - y_2} \right) = \frac{K_{3,y}}{\phi\mu} \left(\frac{P_b - P_3}{y_3 - y_b} \right); \quad (3.41)$$

$$v_c = \frac{K_{4,x}}{\phi\mu} \left(\frac{P_4 - P_c}{x_c - x_4} \right) = \frac{K_{3,x}}{\phi\mu} \left(\frac{P_c - P_3}{x_3 - x_c} \right); \quad (3.42)$$

$$v_d = \frac{K_{1,y}}{\phi\mu} \left(\frac{P_1 - P_d}{y_d - y_1} \right) = \frac{K_{4,y}}{\phi\mu} \left(\frac{P_d - P_4}{y_4 - y_d} \right); \quad (3.43)$$

$$v_e = \frac{K_{1,z}}{\phi\mu} \left(\frac{P_1 - P_e}{z_e - z_1} \right) = \frac{K_{5,z}}{\phi\mu} \left(\frac{P_e - P_5}{z_5 - z_e} \right); \quad (3.44)$$

$$v_f = \frac{K_{2,z}}{\phi\mu} \left(\frac{P_2 - P_f}{z_f - z_2} \right) = \frac{K_{6,z}}{\phi\mu} \left(\frac{P_f - P_6}{z_6 - z_f} \right); \quad (3.45)$$

$$v_g = \frac{K_{3,z}}{\phi\mu} \left(\frac{P_3 - P_g}{z_g - z_3} \right) = \frac{K_{7,z}}{\phi\mu} \left(\frac{P_g - P_7}{z_7 - z_g} \right); \quad (3.46)$$

$$v_h = \frac{K_{4,z}}{\phi\mu} \left(\frac{P_4 - P_h}{z_h - z_4} \right) = \frac{K_{8,z}}{\phi\mu} \left(\frac{P_h - P_8}{z_8 - z_h} \right); \quad (3.47)$$

$$v_i = \frac{K_{5,x}}{\phi\mu} \left(\frac{P_5 - P_i}{x_i - x_5} \right) = \frac{K_{6,x}}{\phi\mu} \left(\frac{P_i - P_6}{x_6 - x_i} \right); \quad (3.48)$$

$$v_j = \frac{K_{6,y}}{\phi\mu} \left(\frac{P_6 - P_j}{y_j - y_6} \right) = \frac{K_{7,y}}{\phi\mu} \left(\frac{P_j - P_7}{y_7 - y_j} \right); \quad (3.49)$$

$$v_k = \frac{K_{8,x}}{\phi\mu} \left(\frac{P_8 - P_k}{x_k - x_8} \right) = \frac{K_{7,x}}{\phi\mu} \left(\frac{P_k - P_7}{x_7 - x_k} \right); \quad (3.50)$$

$$v_l = \frac{K_{5,y}}{\phi\mu} \left(\frac{P_5 - P_l}{y_l - y_5} \right) = \frac{K_{8,y}}{\phi\mu} \left(\frac{P_l - P_8}{y_8 - y_l} \right); \quad (3.51)$$

where, v_i are the flux across the primary grid block interface i ($i = a, b, c, d, e, f, g, h, i, j, k, l$); $K_{subcell,x}$, $K_{subcell,y}$ and $K_{subcell,z}$ is the x -, y - and z - directional permeability in the sub-quadrilateral ($subcell = 1, 2, 3, 4, 5, 6, 7, 8$), respectively; x_{vertex} , y_{vertex} and z_{vertex} is the x -, y - and z - coordinate of the vertex ($vertex = 1, 2, 3, 4, 5, 6, 7, 8, a, b, c, d, e, f, g, h, i, j, k, l$), respectively.

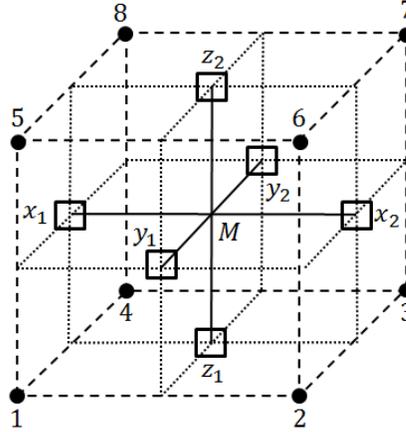


Figure 3.10: The infinitely small control volumes around the primary grid block edge midpoints ($x_1, x_2, y_1, y_2, z_1, z_2$)

Besides, as shown in Figure 3.10, the six velocity divergence free equations over the infinitely small control volumes around the primary grid block edge midpoints ($x_1, x_2, y_1, y_2, z_1, z_2$)

can be written as,

$$\iint_{\Omega_{x_1}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{x_1}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.52)$$

$$\iint_{\Omega_{x_2}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{x_2}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.53)$$

$$\iint_{\Omega_{y_1}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{y_1}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.54)$$

$$\iint_{\Omega_{y_2}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{y_2}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.55)$$

$$\iint_{\Omega_{z_1}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{z_1}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.56)$$

$$\iint_{\Omega_{z_2}} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega_{z_2}} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0; \quad (3.57)$$

where \mathbf{v} is the velocity field, Ω_{vertex} is the infinitely small control volume around the vertex ($vertex = x_1, x_2, y_1, y_2, z_1, z_2$), $\partial\Omega_{vertex}$ is the overall boundary of Ω_{vertex} ; \mathbf{n} is the outward pointing unit normal field of the boundary $\partial\Omega$; and Γ is the boundary of Ω_{vertex} .

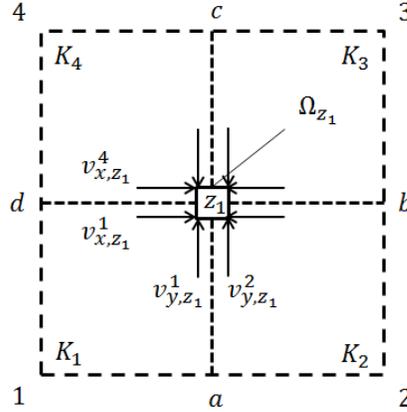


Figure 3.11: The normal fluxes at the vertex z_1 in different grid blocks

The flux along the primary grid block edge is assumed to be continuous, thus, the three-dimensional divergence free equations can be simplified to two-dimensional form, as shown

in Figure 3.11,

$$\begin{aligned} \int_{\partial\Omega_{z_1}} \mathbf{v} \cdot \mathbf{n} d\Gamma &= \mathbf{v}_{x,z_1}^1 \cdot \mathbf{n}_{x,z_1}^1 \Delta\Gamma_{x,z_1}^1 + \mathbf{v}_{y,z_1}^1 \cdot \mathbf{n}_{y,z_1}^1 \Delta\Gamma_{y,z_1}^1 + \mathbf{v}_{x,z_1}^2 \cdot \mathbf{n}_{x,z_1}^2 \Delta\Gamma_{x,z_1}^2 + \mathbf{v}_{y,z_1}^2 \cdot \mathbf{n}_{y,z_1}^2 \Delta\Gamma_{y,z_1}^2 \\ &+ \mathbf{v}_{x,z_1}^3 \cdot \mathbf{n}_{x,z_1}^3 \Delta\Gamma_{x,z_1}^3 + \mathbf{v}_{y,z_1}^3 \cdot \mathbf{n}_{y,z_1}^3 \Delta\Gamma_{y,z_1}^3 + \mathbf{v}_{x,z_1}^4 \cdot \mathbf{n}_{x,z_1}^4 \Delta\Gamma_{x,z_1}^4 + \mathbf{v}_{y,z_1}^4 \cdot \mathbf{n}_{y,z_1}^4 \Delta\Gamma_{y,z_1}^4 = 0, \end{aligned} \quad (3.58)$$

where, $\mathbf{v}_{direction,z_1}^{subcell}$ and $\mathbf{n}_{direction,z_1}^{subcell}$ are the directional ($direction = x, y$) velocity vector and the outward pointing unit normal at vertex z_1 in the subcell ($subcell = 1, 2, 3, 4$), respectively.

Since the control volume is infinitely small and the regular structured grid block is applied, Eq. 3.58 can be simplified as,

$$-v_{x,z_1}^1 - v_{y,z_1}^1 + v_{x,z_1}^2 - v_{y,z_1}^2 + v_{x,z_1}^3 + v_{y,z_1}^3 - v_{x,z_1}^4 + v_{y,z_1}^4 = 0, \quad (3.59)$$

where v_{x,z_1}^1 is the velocity module of the velocity vector \mathbf{v}_{x,z_1}^1 , which can be determined by Darcy's law,

$$v_{x,z_1}^1 = -\frac{K_{1,x}}{\phi\mu} \left(\frac{P_{z_1} - P_d}{x_{z_1} - x_d} \right). \quad (3.60)$$

The last velocity divergence free equation can be written over the entire dual-cell region Ω (as shown in Figure 3.12),

$$\iint_{\Omega} \nabla \cdot \mathbf{v} dV = \int_{\partial\Omega} \mathbf{v} \cdot \mathbf{n} d\Gamma = 0. \quad (3.61)$$

Since the flux varies linearly along the x -, y - and z - axes, Eq. 3.61 becomes,

$$\begin{aligned} &\left(-v_x^1 - v_x^4 - v_x^5 - v_x^8 + v_x^2 + v_x^3 + v_x^6 + v_x^7 \right) \Delta y \Delta z \\ &+ \left(-v_y^1 - v_y^2 - v_y^5 - v_y^6 + v_y^4 + v_y^3 + v_y^7 + v_y^8 \right) \Delta x \Delta z \\ &+ \left(-v_z^1 - v_z^2 - v_z^3 - v_z^4 + v_z^5 + v_z^6 + v_z^7 + v_z^8 \right) \Delta x \Delta y = 0 \end{aligned} \quad (3.62)$$

where $v_{direction}^{subcell}$ is the directional- ($direction = x, y, z$) velocity in sub-cell ($subcell = 1, 2, 3, 4, 5, 6, 7, 8$) as shown in Figure 3.12d; Δx , Δy and Δz is the length, width and height of the grid block, respectively.

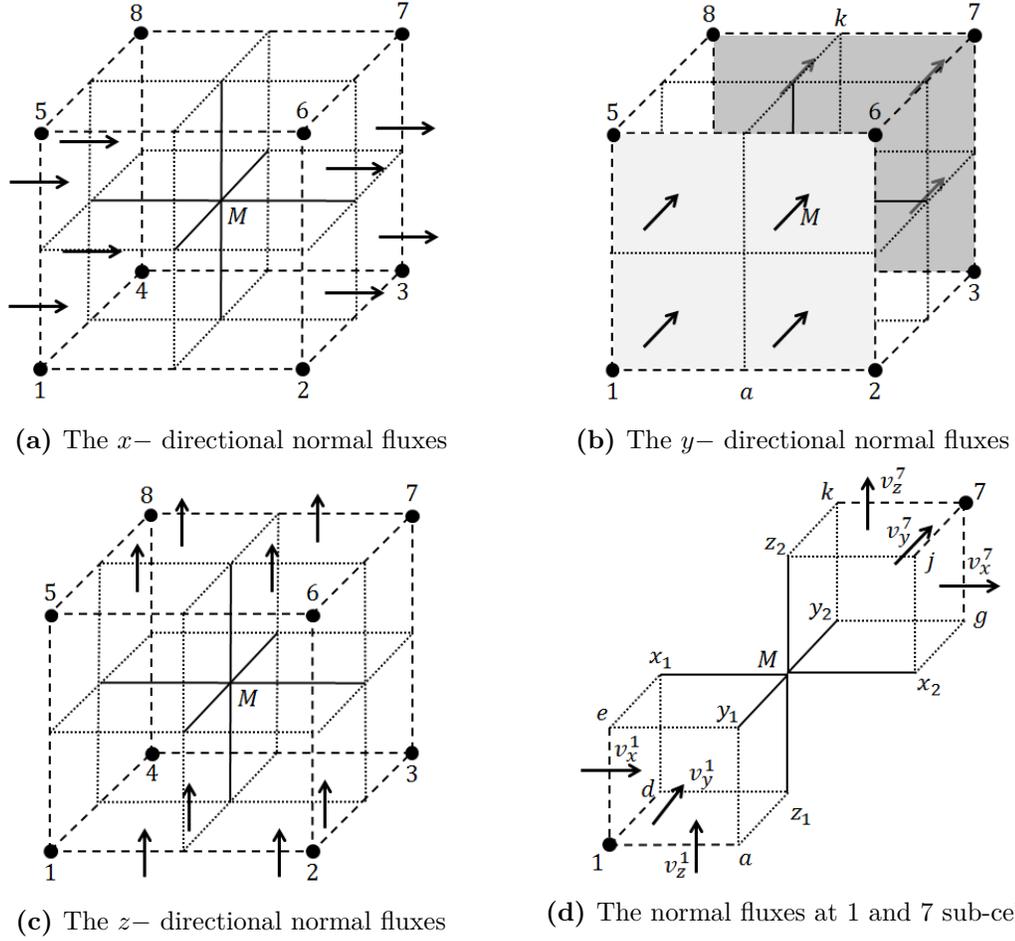


Figure 3.12: The normal fluxes at the 3D dual-cell boundaries

The average directional-velocity in sub-cell can be determined using Darcy's law, *i.e.*,

$$v_x^1 = -\frac{K_{x,1}}{4\phi\mu} \left(\frac{P_a - P_1}{x_a - x_1} + \frac{P_{z_1} - P_d}{x_{z_1} - x_d} + \frac{P_{y_1} - P_e}{x_{y_1} - x_e} + \frac{P_M - P_{x_1}}{x_M - x_{x_1}} \right). \quad (3.63)$$

Solving the nineteen equations (Eq. 3.40 to 3.51; Eq. 3.52 to 3.57; and Eq. 3.61) at each dual-cell, the nineteen newly introduced pressures can be determined. Thus, the pressure values at each sub-hexahedral vertex are known, and the pressure distribution at each sub-hexahedral can be determined.

Recall that the pressure distribution can be approximated by a trilinear equation (Eq. 3.39) at each sub-hexahedral, where $P_0, A_1, A_2, A_3, B_1, B_2, B_3, C$ are pressure coefficients. These coefficients can be determined by its local vertex coordinates and pressure values. For

example in the sub-hexahedral 1 of Figure 3.8, we have

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ B_1 \\ B_2 \\ B_3 \\ C \\ P_0 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & x_1 y_1 & x_1 z_1 & y_1 z_1 & x_1 y_1 z_1 & 1 \\ x_a & y_a & z_a & x_a y_a & x_a z_a & y_a z_a & x_a y_a z_a & 1 \\ x_{z_1} & y_{z_1} & z_{z_1} & x_{z_1} y_{z_1} & x_{z_1} z_{z_1} & y_{z_1} z_{z_1} & x_{z_1} y_{z_1} z_{z_1} & 1 \\ x_d & y_d & z_d & x_d y_d & x_d z_d & y_d z_d & x_d y_d z_d & 1 \\ x_e & y_e & z_e & x_e y_e & x_e z_e & y_e z_e & x_e y_e z_e & 1 \\ x_{y_1} & y_{y_1} & z_{y_1} & x_{y_1} y_{y_1} & x_{y_1} z_{y_1} & y_{y_1} z_{y_1} & x_{y_1} y_{y_1} z_{y_1} & 1 \\ x_M & y_M & z_M & x_M y_M & x_M z_M & y_M z_M & x_M y_M z_M & 1 \\ x_{x_1} & y_{x_1} & z_{x_1} & x_{x_1} y_{x_1} & x_{x_1} z_{x_1} & y_{x_1} z_{x_1} & x_{x_1} y_{x_1} z_{x_1} & 1 \end{bmatrix}^{-1} \begin{bmatrix} P_1 \\ P_a \\ P_{z_1} \\ P_d \\ P_e \\ P_{y_1} \\ P_M \\ P_{x_1} \end{bmatrix}, \quad (3.64)$$

where, x_a, y_a, z_a are coordinates of the vertex a , and P_a is the vertex pressure.

By applying Darcy's Law, the velocity field can then be derived as,

$$v_x = -\frac{K_x}{\phi\mu} \frac{\partial P}{\partial x} = -\frac{K_x}{\phi\mu} (A_1 + B_1 y + B_2 z + C y z); \quad (3.65)$$

$$v_y = -\frac{K_y}{\phi\mu} \frac{\partial P}{\partial y} = -\frac{K_y}{\phi\mu} (A_2 + B_1 x + B_3 z + C x z); \quad (3.66)$$

$$v_z = -\frac{K_z}{\phi\mu} \frac{\partial P}{\partial z} = -\frac{K_z}{\phi\mu} (A_3 + B_2 x + B_3 y + C x y); \quad (3.67)$$

where, K_x, K_y, K_z are the principal permeabilities; ϕ is porosity and μ is fluid viscosity.

It is also straightforward to verify that the velocity field in each sub-quadrilateral is mass conservative, *i.e.*,

$$\nabla \cdot \mathbf{v} = \frac{dv_x}{dx} + \frac{dv_y}{dy} + \frac{dv_z}{dz} = 0. \quad (3.68)$$

Next, the streamlines can be generated using the approximated velocity field.

If x changes monotonically along the streamline, $v_x \neq 0$ holds true within the local sub-cell, the streamline can be generated using x as the parameter. Substituting the velocity

approximation function Eq. 3.65 to the mathematical definition of streamline Eq. 3.2 yields two coupled first-order differential equations,

$$\frac{dy}{dx} = \frac{v_y}{v_x} = \frac{K_y (A_2 + B_1x + B_3z + Cxz)}{K_x (A_1 + B_1y + B_2z + Cyz)}; \quad (3.69)$$

$$\frac{dz}{dx} = \frac{v_z}{v_x} = \frac{K_z (A_3 + B_2x + B_3y + Cxy)}{K_x (A_1 + B_1y + B_2z + Cyz)}. \quad (3.70)$$

Given an arbitrary entry point (x_1, y_1, z_1) at local sub-hexahedral, the streamline geometry can be calculated by solving these two coupled first-order differential equations simultaneously using a numerical method, for example, the Runge-Kutta 4th order method (Appendix C). The exit point (x_2, y_2, z_2) of the streamline in the current sub-hexahedral is determined by solving the intersection point between the streamline and the boundary of the sub-hexahedral.

Then, the incremental time-of-flight in sub-cell can be integrated numerically (a description of numerical integration using the trapezoidal method is given in Appendix D) using x as the parameter, *i.e.*,

$$\Delta\tau = \int_{x_1}^{x_2} \frac{dx}{v_x} = \int_{x_1}^{x_2} \frac{dx}{-\frac{K_x}{\phi\mu} (A_1 + B_1y + B_2z + Cyz)}. \quad (3.71)$$

Similarly, if y changes monotonically along the streamline, the streamline can be generated using y as the parameter,

$$\frac{dx}{dy} = \frac{v_x}{v_y} = \frac{K_x (A_1 + B_1y + B_2z + Cyz)}{K_y (A_2 + B_1x + B_3z + Cxz)}; \quad (3.72)$$

$$\frac{dz}{dy} = \frac{v_z}{v_y} = \frac{K_z (A_3 + B_2x + B_3y + Cxy)}{K_y (A_2 + B_1x + B_3z + Cxz)}. \quad (3.73)$$

In this case, the incremental time-of-flight is integrated numerically using y as the parameter, *i.e.*,

$$\Delta\tau = \int_{y_1}^{y_2} \frac{dy}{v_y} = \int_{y_1}^{y_2} \frac{dy}{-\frac{K_y}{\phi\mu} (A_2 + B_1x + B_3z + Cxz)}. \quad (3.74)$$

Otherwise, if z changes monotonically along the streamline, the streamline can be generated

using z as the parameter,

$$\frac{dx}{dz} = \frac{v_x}{v_z} = \frac{K_x (A_1 + B_1 y + B_2 z + C y z)}{K_z (A_3 + B_2 x + B_3 y + C x y)} \quad (3.75)$$

$$\frac{dy}{dz} = \frac{v_y}{v_z} = \frac{K_y (A_2 + B_1 x + B_3 z + C x z)}{K_z (A_3 + B_2 x + B_3 y + C x y)}. \quad (3.76)$$

The incremental time-of-flight can be integrated numerically using z as the parameter, *i.e.*,

$$\Delta\tau = \int_{z_1}^{z_2} \frac{dz}{v_z} = \int_{z_1}^{z_2} \frac{dz}{-\frac{K_z}{\phi\mu} (A_3 + B_2 x + B_3 y + C x y)}. \quad (3.77)$$

In the above derivations from Eq. 3.69 to 3.77, at least one of these situations will always hold true because otherwise the streamline would be arriving at a point inside the block where it vanishes (stagnation point), and the streamline is terminated at this point.

Following the algorithm introduced above, the exit coordinate and the time-of-flight with given entry point have been determined at three-dimensional grid blocks by constructing dual-cells. Hence, the streamlines can be traced using the Trilinear method. The calculation examples and discussions on accuracy and efficiency of both Bilinear and Trilinear methods will be presented in Chapter 5.

Chapter 4

Tracing Streamlines Using Cubic Pressure Approximation Functions

The new streamline tracing methods introduced in this research assume the pressure distribution in a reservoir is piece-wise polynomial. This chapter introduces a Cubic streamline tracing method. Different from the Bilinear and Trilinear method, the Cubic method abandons the pressure continuity everywhere in the reservoir and assumes the pressure distributions are third-degree polynomials. It determines the velocity field by Darcy's Law using the velocity approximation function derived from third-degree pressure functions.

The structure of this chapter is: first, discuss the possibility of using high-degree polynomials to approximate pressure distributions; second, introduce the cubic pressure and velocity approximation functions together with the stream function for streamline tracing purposes; third, propose the Cubic streamline tracing method in both two- and three- dimensional grid blocks.

4.1 The High-degree Polynomials for Pressure and Velocity Approximations

The essential goal for accurate streamline tracing method is to approximate the velocity field at grid blocks more accurately. Assuming the pressure gradient is the main driving force for fluid flow in porous medium (as opposed to the capillary pressure), the velocity field is proportional to the pressure gradient. Then, high-degree polynomials are considered for pressure and velocity approximations.

There are potentially many high degree polynomials that can be used to approximate pressure distributions. The four criteria for appropriate pressure and velocity approximation functions are listed as follows, as discussed in section 3.3.2:

1. The velocity approximation function must obey the velocity field divergence free condition (Eq. 3.10);
2. The velocity approximation function is derived from the pressure function through Darcy's Law (Eq. 3.11);
3. The pressure approximation function must obey the Laplace equation (Eq. 3.13);
4. The pressure and velocity functions are given by explicit functions.

There are two sets of quadratic equations to approximate pressure in two-dimensional grid blocks that satisfy these four requirements. These pressures, velocities and stream functions are summarized as,

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_ky^2; \quad (4.1)$$

$$v_x = -\frac{K_x}{\phi\mu} (A_1 + 2Bx) = a_1 + bx; \quad (4.2)$$

$$v_y = -\frac{K_y}{\phi\mu} (A_2 - 2Br_ky) = a_2 - by; \quad (4.3)$$

$$\psi = \int v_y dx - \int v_x dy = a_2x - a_1y - bxy; \quad (4.4)$$

and,

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_ky^2 + Cxy; \quad (4.5)$$

$$v_x = -\frac{K_x}{\phi\mu}(A_1 + 2Bx + Cy) = a_1 + bx + c_1y; \quad (4.6)$$

$$v_y = -\frac{K_y}{\phi\mu}(A_2 - 2Br_ky + Cx) = a_2 - by + c_2x; \quad (4.7)$$

$$\psi = \int v_y dx - \int v_x dy = a_2x - a_1y - bxy + \frac{1}{2}c_2x^2 - \frac{1}{2}c_1y^2; \quad (4.8)$$

where P_0, A_1, A_2, B, C are pressure coefficients; a_1, a_2, b, c_1, c_2 are velocity coefficients; K_x and K_y is the x - and y - directional permeability, respectively; and $r_k = \frac{K_x}{K_y}$ is the permeability ratio.

Similarly, the quadratic equation for pressure approximations in three-dimensional grid blocks and its velocity approximations functions which satisfy these four requirements are given as,

$$P = P_0 + A_1x + A_2y + A_3z + B_1x^2 + B_2y^2 - \frac{K_xB_1 + K_yB_2}{K_z}z^2; \quad (4.9)$$

$$v_x = -\frac{K_x}{\phi\mu}(A_1 + 2B_1x) = a_1 + b_1x; \quad (4.10)$$

$$v_y = -\frac{K_y}{\phi\mu}(A_2 + 2B_2y) = a_2 + b_2y; \quad (4.11)$$

$$v_z = -\frac{K_z}{\phi\mu}(A_3 - 2\frac{K_xB_1 + K_yB_2}{K_z}z) = a_3 - (b_1 + b_2)z; \quad (4.12)$$

The velocity functions in Eq. 4.2 and 4.10 are applied by Pollock's method in two- and three-dimensional structured grid blocks. We note that Pollock's method cannot ensure the local mass conservation; but in structured finite-difference grid blocks, the approximated velocity field using Pollock's method satisfies the mass conservation law locally. This is because the fluxes determined using a finite-difference approach are mass conservative, and the area of grid block interfaces (Δx , Δy and Δz) keeps constant in each grid block. More specifically, as shown in Figure 1.2, the mass balance equation applied by a finite difference approach is,

$$\Delta y(v_{x2} - v_{x1}) + \Delta x(v_{y2} - v_{y1}) = 0. \quad (4.13)$$

Substituting the velocity functions of Pollock’s method (Eq. 1.1),

$$v(x) = v_{x1} + a_x(x - x_1); \quad v(y) = v_{y1} + a_y(y - y_1);$$

into Eq. 4.13 yields,

$$a_x + a_y = 0; \tag{4.14}$$

where a_x and a_y is velocity gradient in x - and y - directions, respectively.

Therefore, the velocity field approximated by Pollock’s method is divergence free for this case, *i.e.*, $\nabla \cdot \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = a_x + a_y = 0$, and the local mass conservation is satisfied.

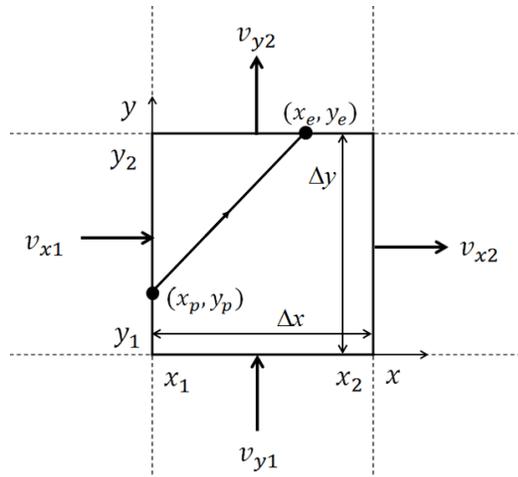


Figure 4.1: Illustration of Pollock’s tracing method through a 2D Cartesian grid block (same with Figure 1.2)

The pressure function in Eq. 4.5 is the pressure distribution assumption of the high-order streamline tracing method on triangles (Matringe and Juanes, 2006). These two methods are very robust in streamline tracing, but yet have limitations. The linear velocity field approximation applied in Pollock’s method may lead to large errors in both streamline location and in time-of-flight along streamlines in structurally complex reservoirs. For instance, a streamline can theoretically enter and exit in the same grid block interface when the grid block has multiple velocities across a single interface, but Pollock’s method fails to model this behavior. The application of high-order streamline tracing method (Matringe and Juanes, 2006) is limited to two-dimensional grid blocks.

Nevertheless, the above discussions also indicate that considering pressure approximation functions in streamline simulation is fundamentally correct. Not surprisingly, these widely used and studied tracing algorithms, Pollock's method (Pollock 1988) and high-order method (Matringe and Juanes 2006) all support this concept.

A cubic polynomial for pressure approximations in two-dimensional grid blocks that satisfies these four requirements, and its velocity and stream functions are given as,

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_ky^2 + Cxy + Dxy^2 - Dx^2y - \frac{D}{3r_k}x^3 + \frac{Dr_k}{3}y^3; \quad (4.15)$$

$$v_x = a_1 + bx + c_1y + d_2y^2 - 2d_2xy - d_1x^2; \quad (4.16)$$

$$v_y = a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2; \quad (4.17)$$

$$\psi = a_2x - a_1y - bxy + \frac{c_2}{2}x^2 - \frac{c_1}{2}y^2 + d_1x^2y + d_2xy^2 - \frac{d_1}{3}x^3 - \frac{d_2}{3}y^3.; \quad (4.18)$$

where P_0, A_1, A_2, B, C, D are pressure coefficients; $a_1, a_2, b, c_1, c_2, d_1, d_2$ are velocity coefficients; and $r_k = K_x/K_y$ is permeability ratio.

The velocity coefficients are related to the pressure coefficients by,

$$a_1 = -\frac{K_x A_1}{\phi\mu}; \quad a_2 = -\frac{K_y A_2}{\phi\mu}; \quad b = -\frac{2K_x B}{\phi\mu}; \quad c_1 = -\frac{K_x C}{\phi\mu}; \quad (4.19)$$

$$c_2 = -\frac{K_y C}{\phi\mu}; \quad d_1 = -\frac{K_y D}{\phi\mu}; \quad d_2 = -\frac{K_x D}{\phi\mu}. \quad (4.20)$$

Similarly, the third-degree polynomial for pressure approximations in three-dimensional grid blocks and its velocity functions that satisfies these four requirements are obtained as,

$$\begin{aligned} P = & P_0 + A_1x + A_2y + A_3z + \frac{B_1x^2}{2} + \frac{B_2y^2}{2} - \frac{K_x B_1 + K_y B_2}{2K_z}z^2 + C_1xy + C_2xz + C_3yz \\ & + \frac{D_1}{3}x^3 + D_2xy^2 + D_3x^2y + \frac{D_4}{3}y^3 + D_5x^2z + D_6y^2z - \frac{D_1K_x + D_2K_y}{K_z}xz^2 \\ & - \frac{D_3K_x + D_4K_y}{K_z}yz^2 - \frac{D_5K_x + D_6K_y}{3K_z}z^3; \end{aligned} \quad (4.21)$$

$$v_x = a_1 + b_1x + c_1xy + c_2xz + d_1x^2 + d_2xy^2 + 2d_3xy + 2d_5xz - e_1z^2; \quad (4.22)$$

$$v_y = a_2 + b_2y + c_1yx + c_3yz + 2d_2yx + d_3yx^2 + d_4y^2 + 2d_6yz - e_2z^2; \quad (4.23)$$

$$v_z = a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2; \quad (4.24)$$

where $P_0, A_1, A_2, A_3, B_1, B_2, C_1, C_2, C_3, D_1, D_2, D_3, D_4, D_5, D_6$ are pressure coefficients; the lowercase letters are the velocity coefficients.

The twenty-three velocity coefficients are related to the pressure coefficients through Darcy's law,

$$a_1 = -\frac{K_x A_1}{\phi\mu}; \quad a_2 = -\frac{K_y A_2}{\phi\mu}; \quad a_3 = -\frac{K_z A_3}{\phi\mu}; \quad b_1 = -\frac{K_x B_1}{\phi\mu}; \quad (4.25)$$

$$b_2 = -\frac{K_y B_2}{\phi\mu}; \quad c_{1x} = -\frac{K_x C_1}{\phi\mu}; \quad c_{2x} = -\frac{K_x C_2}{\phi\mu}; \quad c_{1y} = -\frac{K_y C_1}{\phi\mu}; \quad (4.26)$$

$$c_{3y} = -\frac{K_y C_3}{\phi\mu}; \quad c_{2z} = -\frac{K_z C_2}{\phi\mu}; \quad c_{3z} = -\frac{K_z C_3}{\phi\mu}; \quad d_{1x} = -\frac{K_x D_1}{\phi\mu}; \quad (4.27)$$

$$d_{2x} = -\frac{K_x D_2}{\phi\mu}; \quad d_{3x} = -\frac{K_x D_3}{\phi\mu}; \quad d_{5x} = -\frac{K_x D_5}{\phi\mu}; \quad d_{2y} = -\frac{K_y D_2}{\phi\mu}; \quad (4.28)$$

$$d_{3y} = -\frac{K_y D_3}{\phi\mu}; \quad d_{4y} = -\frac{K_y D_4}{\phi\mu}; \quad d_{6y} = -\frac{K_y D_6}{\phi\mu}; \quad d_{5z} = -\frac{K_z D_5}{\phi\mu}; \quad (4.29)$$

$$d_{6z} = -\frac{K_z D_6}{\phi\mu}; \quad (4.30)$$

$$e_1 = -\frac{K_x D_1 K_x + D_2 K_y}{K_z \phi\mu}; \quad \text{and} \quad e_2 = -\frac{K_y D_3 K_x + D_4 K_y}{K_z \phi\mu}. \quad (4.31)$$

In the following sections, the velocity field approximation and streamline tracing in two- and three- dimensional grid blocks using the Cubic method are introduced.

4.2 The Cubic Streamline Tracing Method

Assuming the pressure distribution and velocity field in a reservoir are piece-wise cubic functions as shown in the Eq. 4.15 and 4.21, the Cubic streamline tracing method (the Cubic method in short) is developed. In brief, there are three steps for tracing streamlines in a grid block using the Cubic method:

1. Interpret the velocities at grid block interfaces;

2. Approximate the velocity field in grid blocks by determining velocity coefficients;
3. Solve for the geometry of streamline and time-of-flight.

The Cubic method is a semi-analytical method since in two-dimensional grid blocks, the stream function is an explicit formula derived from the Cubic pressure approximation function; whereas in three-dimensional grid blocks, the streamlines are generated by numerically solving two coupled first-order differential equations.

This Cubic method solves for more coefficients in pressure and velocity approximations than the Bilinear and Pollock methods, theoretically, it gives more accurate results in velocity field approximations under the same grid resolution, and the accuracy of streamline tracing results is improved. This improvement for velocity field approximation is sometimes crucial when the pressure distribution and velocity field vary significantly in simulation domain. This statement will be further discussed through numerical case studies in Chapter 5.

4.2.1 The Cubic method in two-dimensional grid blocks

The emphasis of this sub-section is to introduce the Cubic method in two-dimensional regular structured grid blocks. Similar to the Bilinear method, the grid blocks discretization and the pressure values at primary nodes calculation are assumed to be already completed by applying a finite-difference method. An example of pressure nodes and regular grid blocks in the reservoir is given in Figure 4.2. The objective is to approximate the velocity field and then tracing streamlines.

Recall the velocity approximation functions for the Cubic method in a two-dimensional grid block are, Eq. 4.16 and 4.17

$$v_x = a_1 + bx + c_1y + d_2y^2 - 2d_2xy - d_1x^2;$$

$$v_y = a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2;$$

where, $a_1, a_2, b, c_1, c_2, d_1, d_2$ are the velocity coefficients.

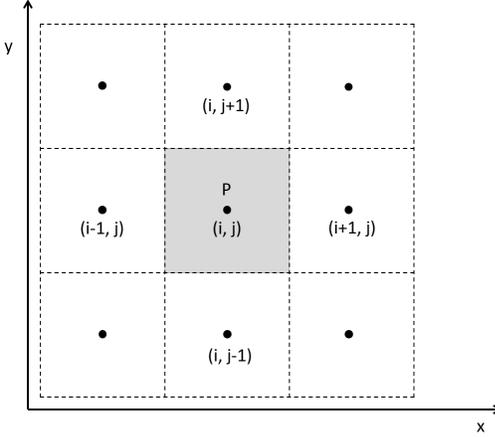


Figure 4.2: An example of pressure nodes and grid blocks in a 2D reservoir

In this Cubic method, the velocity field is approximated by determining these seven velocity coefficients. These seven velocity coefficients can be calculated if the x - and y - directional velocities at the grid block vertexes (eight velocities in total) are known. We note that these eight vertex velocities have seven degrees of freedom, since that according to the velocity divergence free condition, the summation of these velocities equals to zero.

In order to determine two velocities per interface, high-order numerical methods such as mixed finite element method can be applied (Matringe *et al.*, 2006). However, applying these methods may lead to a considerable increase in computational time. In this Cubic method, a new method is applied to interpret two velocities per interface based on the pressure solutions obtained from the finite-difference method. Compared to the mixed finite element method, this method is easy to implement and requires less computational effort.

Considering that pressure is continuously distributed in the reservoir, and that it varies more smoothly than the velocity, the vertex velocities interpretation method applied in the Cubic method is based on two assumptions:

1. $\frac{\partial P}{\partial x}$ is continuous in the y - direction;
2. the distribution of $\frac{\partial P}{\partial x}$ in the y - direction is a linear function in each grid block.

The same assumptions are also applied to $\frac{\partial P}{\partial y}$ in the x - direction.

As shown in Figure 4.3, these two assumptions indicate that the pressure derivatives vary continuously in its perpendicular direction. The assumption of continuous pressure derivatives lack physical significance. In fact, it may contradict the physical behavior at large permeability contrasts. Nevertheless, this assumption imposes a numerical smoothing effect on the numerical results. In other words, the continuous assumption is made for numerical reasons only. The continuous pressure derivative assumption may not hold true in some cases. This limitation can be overcome by separating the reservoir into several sections where the pressure varies smoothly, and applying this assumption in each individual section to approximate continuous pressure derivations.

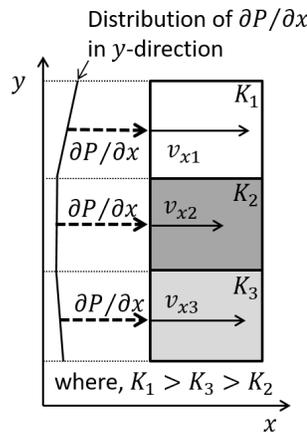


Figure 4.3: An illustration of the assumption that pressure varies continuously in its perpendicular direction

To simplify the description of the velocity interpretation method below, v_x is interpreted along the y -direction. The same algorithms can also be applied to interpret v_y along the x -direction.

Below, we show the three steps to interpret the velocities at grid block vertexes in the x -direction,

1. Calculate the x -directional pressure derivative at the center of grid block interfaces

using numerical differentiation;

2. Approximate the x -directional pressure derivative at grid block vertexes along the y -direction by linear functions;
3. Calculate the x -directional velocity at each grid block vertex using Darcy's law.

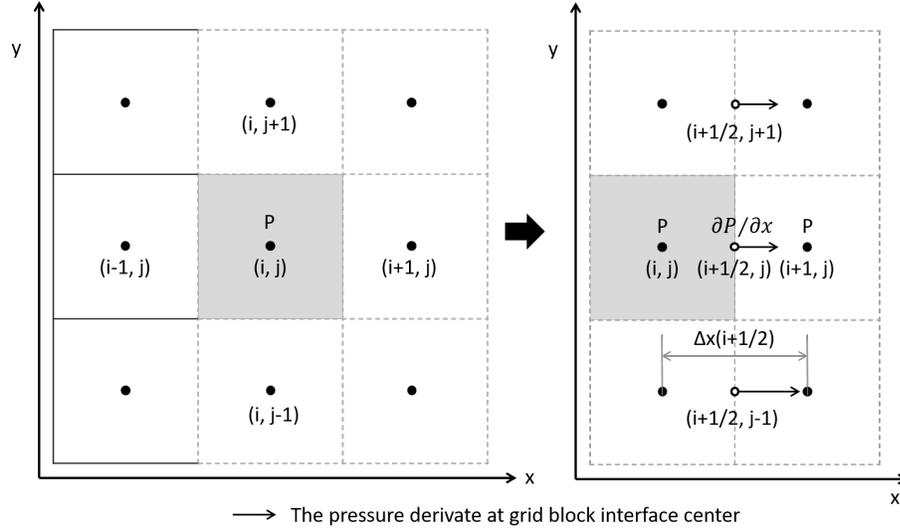


Figure 4.4: Pressure derivative at the center of grid block interfaces

In the first step, as shown in Figure 4.4, the pressure derivative at the center of grid block interface $(i + 1/2, j)$ can be determined by using numerical differentiation, *i.e.*

$$\left. \frac{\partial P}{\partial x} \right|_{i+1/2, j} = \frac{P_{i+1, j} - P_{i, j}}{\Delta x_{i+1/2, j}}, \quad (4.32)$$

where $\Delta x_{i+1/2, j}$ is the x -directional length between pressure nodes at (i, j) and $(i + 1, j)$.

In the second step, the x -direction derivative at the grid block vertexes are determined by using linear functions. According to the two assumptions made for pressure derivatives, the distribution of $\frac{\partial P}{\partial x}$ along y -direction is a continuous line composed of several segments (one line segment per grid block). For example, Figure 4.4 indicates that the problem in this step is to approximate a continuous line composed of three straight line segments with three known points at the center of grid block interfaces. There are $(n + 1)$ points required for defining a continuous line with (n) straight line segments. In this case, only one extra

slope at (J) , *i.e.*,

$$\left. \frac{\partial P}{\partial x} \right|_{J+1/2} = \left. \frac{\partial P}{\partial x} \right|_J + DS(J) \frac{1}{2} \Delta y_J + o(\Delta y_J^2). \quad (4.34)$$

The pressure derivatives at other grid block vertexes can be easily interpreted by applying continuous linear functions sequentially from the location J to reservoir boundaries along y axis. For example, the pressure derivative at interface $(J - 1/2)$ is given by,

$$\left. \frac{\partial P}{\partial x} \right|_{J-1/2} = 2 \left. \frac{\partial P}{\partial x} \right|_J - \left. \frac{\partial P}{\partial x} \right|_{J+1/2}. \quad (4.35)$$

Finally, the corresponding velocity at grid block vertex can be obtained by applying Darcy's law,

$$v_x|_{i+1/2, j+1/2} = - \left(\frac{K_x}{\phi \mu} \frac{\partial P}{\partial x} \right)_{i+1/2, j+1/2}, \quad (4.36)$$

where $K_x|_{i+1/2}$ is the upscaled permeability in x -direction at the interface $i + 1/2$. As shown in Figure 4.6, it is given by the harmonic mean of permeability, *i.e.*,

$$K_x|_{i+1/2} = \frac{\Delta x_i + \Delta x_{i+1}}{\frac{\Delta x_i}{K_x(i)} + \frac{\Delta x_{i+1}}{K_x(i+1)}}, \quad (4.37)$$

where Δx is the grid block length in x -direction.

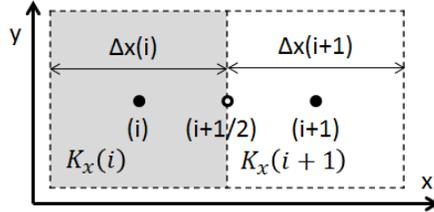


Figure 4.6: The x -directional permeability of adjacent grid blocks

Applying the same algorithm for other grid blocks in the reservoir, the v_x at all grid block vertexes can be determined, as shown in Figure 4.7a. Following the same procedures, the v_y at all grid block vertexes can also be determined, as shown in Figure 4.7b. Finally, there are eight vertex velocities interpreted at each grid block, as an example shown in Figure 4.8.

The velocity field in the grid block can be approximated by determining the seven velocity coefficients. Recall the velocity approximation functions in the Cubic method for two-

dimensional grid block are Eq. 4.16 and 4.17,

$$\begin{aligned}v_x &= a_1 + bx + c_1y + d_2y^2 - 2d_2xy - d_1x^2, \\v_y &= a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2.\end{aligned}$$

where, $a_1, a_2, b, c_1, c_2, d_1, d_2$ are the velocity coefficients.

The velocity coefficients are given by substituting the vertex velocities and coordinates into Eq. 4.16 and 4.17. As an example given in Figure 4.8, the velocity coefficients are given by,

$$\begin{bmatrix} a_1 \\ a_2 \\ b \\ c_1 \\ c_2 \\ d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_1 & y_1 & 0 & -x_1^2 & y_1^2 - 2x_1y_1 \\ 1 & 0 & x_2 & y_2 & 0 & -x_2^2 & y_2^2 - 2x_2y_2 \\ 1 & 0 & x_3 & y_3 & 0 & -x_3^2 & y_3^2 - 2x_3y_3 \\ 1 & 0 & x_4 & y_4 & 0 & -x_4^2 & y_4^2 - 2x_4y_4 \\ 0 & 1 & -y_1 & 0 & x_1 & -x_1^2 + 2x_1y_1 & y_1^2 \\ 0 & 1 & -y_2 & 0 & x_2 & -x_2^2 + 2x_2y_2 & y_2^2 \\ 0 & 1 & -y_3 & 0 & x_3 & -x_3^2 + 2x_3y_3 & y_3^2 \end{bmatrix}^{-1} \begin{bmatrix} v_{x1} \\ v_{x2} \\ v_{x3} \\ v_{x4} \\ v_{y1} \\ v_{y2} \\ v_{y3} \end{bmatrix}, \quad (4.38)$$

where the subscripts x_{vertex} , y_{vertex} are the coordinates at vertex ($vertex = 1, 2, 3, 4$); $v_{x(vertex)}$ and $v_{y(vertex)}$ is the x - and y -directional velocity at the vertex, respectively. We note that there are seven independent linear equations, because the sum of these velocities equals to zero.

Furthermore, the pressure distribution can be approximated using Eq. 4.15,

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_ky^2 + Cxy + Dxy^2 - Dx^2y - \frac{D}{3r_k}x^3 + \frac{Dr_k}{3}y^3.$$

The pressure coefficients are determined using the known velocity coefficients,

$$A_1 = -\frac{a_1\phi\mu}{K_x}; \quad A_2 = -\frac{a_2\phi\mu}{K_y}; \quad B = -\frac{b\phi\mu}{2K_x}; \quad (4.39)$$

$$C = -\frac{c_1\phi\mu}{K_x}; \quad D = -\frac{d_1\phi\mu}{K_y}; \quad (4.40)$$

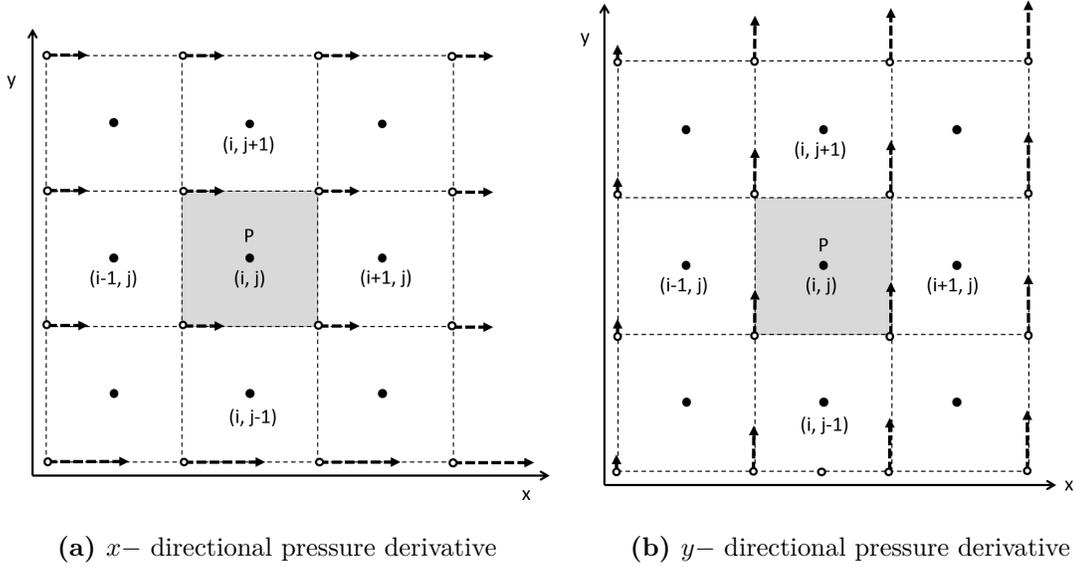


Figure 4.7: The pressure derivatives at every grid block vertex

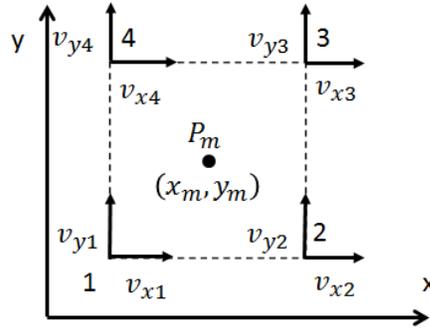


Figure 4.8: Eight vertex velocities interpreted at one grid block

and P_0 is determined by,

$$\begin{aligned}
 P_0 = & P_m - A_1 x_m - A_2 y_m - B x_m^2 + B r_k y_m^2 - C x_m y_m - D x_m y_m^2 + D x_m^2 y_m \\
 & + \frac{D}{3 r_k} x_m^3 - \frac{D r_k}{3} y_m^3.
 \end{aligned} \tag{4.41}$$

where, as shown in Figure 4.8, P_m is the grid block pressure solved from the finite-difference method; x_m and y_m are the coordinate of pressure node.

The stream function can then be derived, *i.e.*,

$$\psi = \int v_y dx - \int v_x dy = a_2 x - a_1 y - b x y + \frac{c_2}{2} x^2 - \frac{c_1}{2} y^2 + d_1 x^2 y + d_2 x y^2 - \frac{d_1}{3} x^3 - \frac{d_2}{3} y^3. \tag{4.42}$$

Given an arbitrary entry point (x_1, y_1) , the geometry of the corresponding streamline in the grid block is given by,

$$a_2x - a_1y - bxy + \frac{c_2}{2}x^2 - \frac{c_1}{2}y^2 + d_1x^2y + d_2xy^2 - \frac{d_1}{3}x^3 - \frac{d_2}{3}y^3 = \psi(x_1, y_1); \quad (4.43)$$

where $\psi(x_1, y_1) = a_2x_1 - a_1y_1 - bx_1y_1 + \frac{c_2}{2}x_1^2 - \frac{c_1}{2}y_1^2 + d_1x_1^2y_1 + d_2x_1y_1^2 - \frac{d_1}{3}x_1^3 - \frac{d_2}{3}y_1^3$ is the constant value of the stream function. Then, the exit point (x_2, y_2) of the streamline in the grid block can be determined by solving for the intersection point between the streamline and the grid block boundary.

Finally, the incremental time-of-flight for the streamline interval within the local grid block can be calculated. If x changes monotonically along the streamline, $v_x \neq 0$ holds true within the grid block, then the time-of-flight can be integrated numerically (trapezoidal method in Appendix D) using x as the parameter,

$$\Delta\tau = \int_{x_1}^{x_2} \frac{dx}{v_x} = \int_{x_1}^{x_2} \frac{dx}{a_1 + bx + c_1y + d_2y^2 - 2d_2xy - d_1x^2}. \quad (4.44)$$

Otherwise, y changes monotonically along the streamline, $v_y \neq 0$ holds true within the grid block, and then the time-of-flight can be integrated numerically using y as the parameter,

$$\Delta\tau = \int_{y_1}^{y_2} \frac{dy}{v_y} = \int_{y_1}^{y_2} \frac{dy}{a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2}. \quad (4.45)$$

Following the Cubic algorithm introduced above, the exit coordinate and the incremental time-of-flight for the streamline with given entry point have been determined at two-dimensional structured grid blocks.

4.2.2 The Cubic method in three-dimensional grid blocks

Tracing streamlines using the Cubic pressure function in three-dimensional grid blocks is similar to the method in two-dimensional grid blocks. The difference is, the pressure and velocity approximation functions are more complex, and more fluxes at grid block interfaces are determined for velocity field approximations.

In this section, the applications of the Cubic streamline tracing method is extended to three-dimensional grid blocks. To start, the grid blocks discretization and the pressure values at primary nodes calculation are assumed to be already completed. An illustration of primary pressure nodes and regular grid blocks is shown in Figure 3.7.

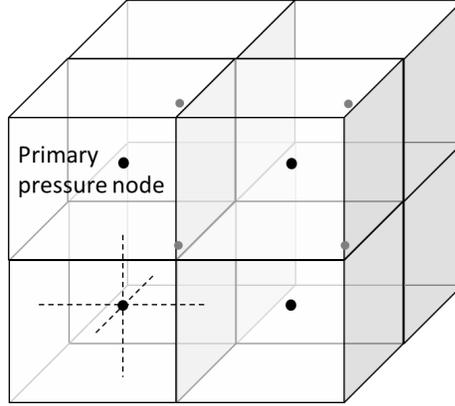


Figure 4.9: An example of the pressure nodes and grid blocks in the 3D reservoir (same as Figure 3.7)

Recall the velocity approximation functions in Cubic method for three-dimensional grid block are Eq. 4.22, 4.23 and 4.24,

$$\begin{aligned}
 v_x &= a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2; \\
 v_y &= a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2; \\
 v_z &= a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz \\
 &\quad - (d_{5x} + d_{6y})z^2;
 \end{aligned}$$

where, $a_1, a_2, \dots, e_1, e_2$ are velocity coefficients. The velocity field can be approximated by determining these seventeen velocity coefficients. These velocity coefficients can be calculated if the x -, y - and z - directional velocities at grid block edges (two velocities per edge, and twenty four velocities in total) are known. We note that these twenty four edge velocities have twenty-three degrees of freedom. This is because, according to the velocity divergence free condition, the sum of these velocities equals to zero.

The velocity interpretation in three-dimensional grid blocks is based on the same assump-

tions as the method applied in two-dimensional grid blocks, introduced in section 4.2.1. Here, the velocity interpretation method is repeated briefly. Similarly, the velocity interpretation method in three-dimensional grid blocks is based on two assumptions: first, the derivatives of pressure is continuously distributed along their perpendicular directions; second, they vary linearly in their perpendicular directions in each grid block.

To simplify the description below, v_y is determined in its perpendicular directions (x - and z - directions). The same algorithms can also be applied to interpret v_y and v_z in their perpendicular directions.

There are three steps in this method to interpret the velocities in the grid block edges,

1. Calculate the y -directional pressure derivative at the center of grid block interfaces using numerical differentiation;
2. Approximate the y -directional pressure derivative at grid block vertexes along x - and z - directions by linear functions;
3. Calculate the y -directional velocity at the grid block edges using Darcy's law.

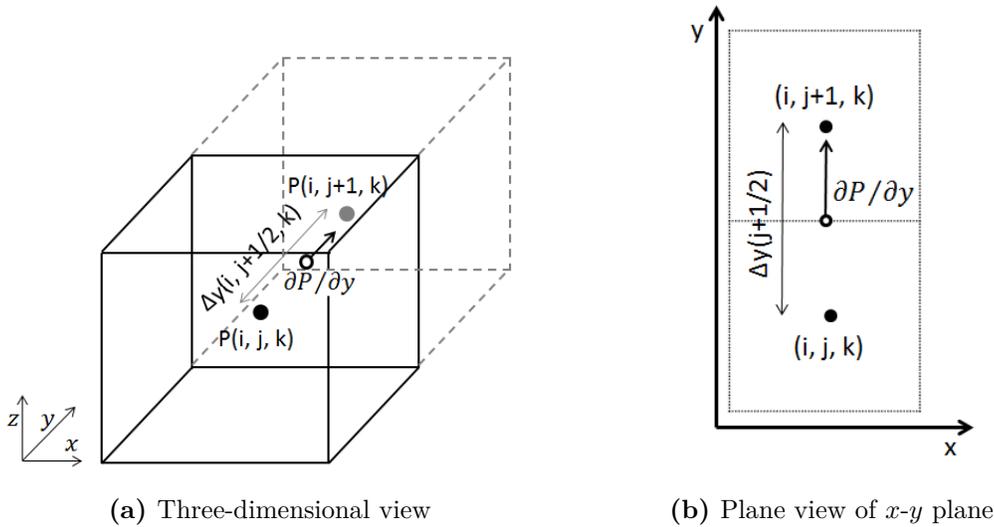


Figure 4.10: y - directional pressure derivative at the center of grid block interfaces

In the first step, as shown in Figure 4.10, the pressure derivative at the center of grid block interface $(i, j + 1/2, k)$ can be determined by using numerical differentiation, *i.e.*

$$\left. \frac{\partial P}{\partial y} \right|_{i, j+1/2, k} = \frac{P_{i, j+1, k} - P_{i, j, k}}{\Delta y_{i+1/2, j, k}}, \quad (4.46)$$

where $\Delta y_{i+1/2, j, k}$ is the x -directional length between pressure nodes at (i, j, k) and $(i, j + 1, k)$.

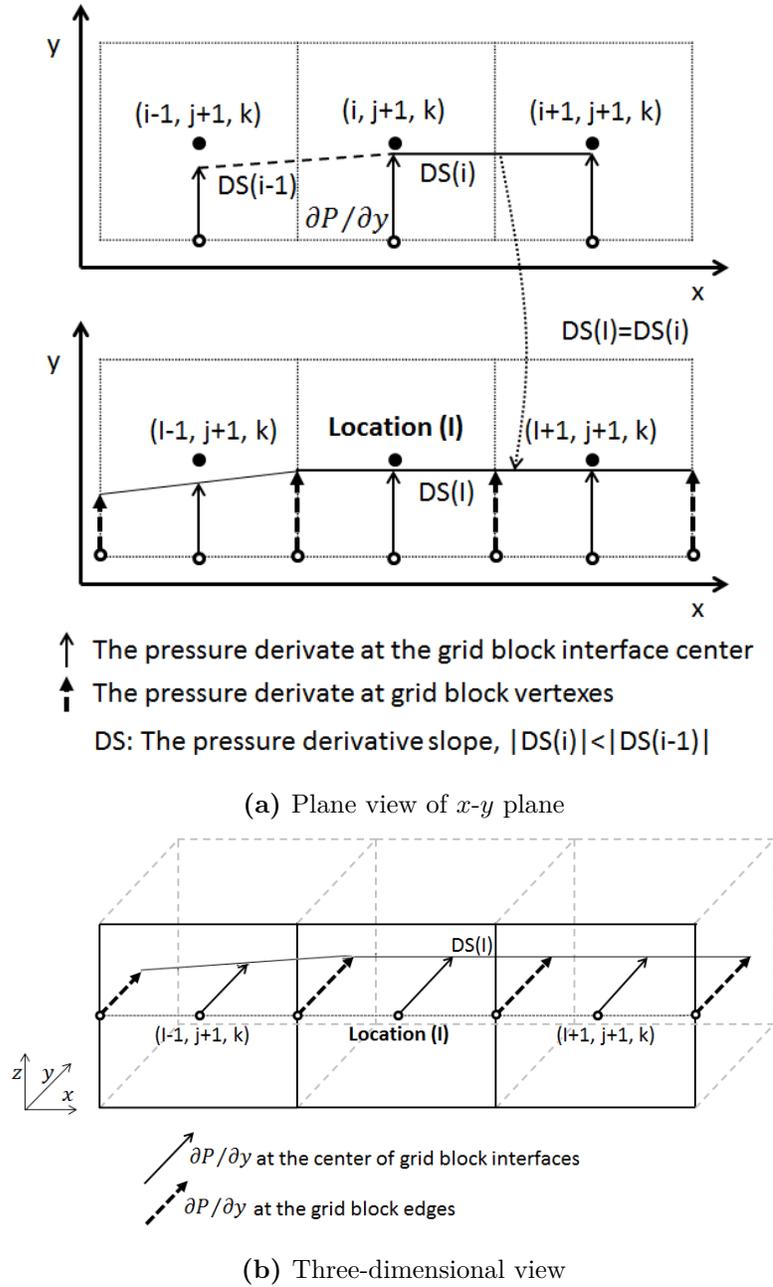


Figure 4.11: y - directional pressure derivative along x - direction

In the second step, the distribution of y - direction derivative along the x - direction is determined by linear functions. As shown in Figure 4.11, the problem is to approximate a continuous line composed of n ($n =$ the number of grid blocks) straight line segments with n known points at the center of grid block interfaces. The extra value of $\frac{\partial P}{\partial y}$ required to solve this problem is determined at position (I), where the absolute value of pressure derivative slope $DS(i)$ between two adjacent grid blocks (i) and ($i + 1$) is minimized (or equals to zeros) at $i \in [1, \dots, N_x - 1]$. $DS(i)$ is defined as,

$$DS(x) = \frac{\frac{\partial P}{\partial y}|_{i+1} - \frac{\partial P}{\partial y}|_i}{\Delta x_{i+1/2}}, \quad (4.47)$$

where $\Delta x_{i+1/2}$ is the x -directional length between pressure nodes at (i) and ($i + 1$).

Therefore, the pressure derivatives at ($I + 1/2$) can be determined by the pressure derivative slope at (I), *i.e.*,

$$\frac{\partial P}{\partial y}\Big|_{I+1/2} = \frac{\partial P}{\partial y}\Big|_I + DS(I) \frac{1}{2} \Delta x_I. \quad (4.48)$$

The pressure derivatives at other grid block edges can be easily interpreted by applying the continuous linear functions sequentially from the location I to reservoir boundaries along x - axis. Finally, the corresponding velocity at the grid block edges can be obtained by applying Darcy's law,

$$v_y|_{j+1/2} = - \left(\frac{K_y}{\phi \mu} \frac{\partial P}{\partial y} \right)_{j+1/2}, \quad (4.49)$$

where $K_y|_{j+1/2}$ is the upscaled permeability in y -direction at the interface $j + 1/2$, it is given by the harmonic mean of permeability, *i.e.*,

$$K_y|_{j+1/2} = \frac{\Delta y_j + \Delta y_{j+1}}{\frac{\Delta y_j}{K_y|_j} + \frac{\Delta y_{j+1}}{K_y|_{j+1}}}; \quad (4.50)$$

where Δy is the grid block length in j -direction.

Applying the same algorithm for other grid block rows (along x - direction) and columns (along y - direction) in the reservoir, the v_y at the grid block edges can be determined. Similarly, the v_x and v_z at the grid block edges can also be determined. Finally, there are

twenty-four velocities interpreted at each grid block, as an example shown in Figure 4.12. We note that these interpreted velocities have twenty-three degrees of freedom.

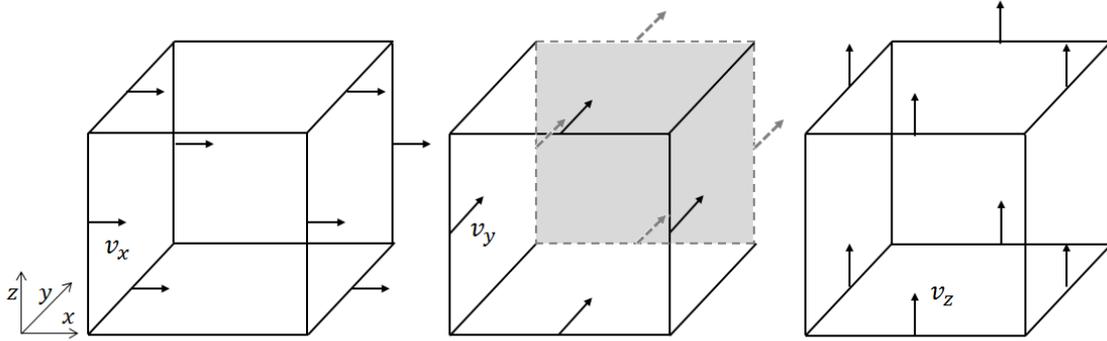


Figure 4.12: The twenty four velocities at the grid block edges

The velocity field in the grid block can be approximated by determining the velocity coefficients. Recall the velocity approximation functions in Cubic method for three-dimensional grid block Eq. 4.22, 4.23, and 4.24,

$$\begin{aligned}
 v_x &= a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2; \\
 v_y &= a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2; \\
 v_z &= a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz \\
 &\quad - (d_{5x} + d_{6y})z^2;
 \end{aligned}$$

where, $a_1, a_2, \dots, e_1, e_2$ are the velocity coefficients.

In each grid block, twenty three independent linear equations can be obtained by substituting the twenty three independent velocities at grid block edges and the corresponding coordinates into the above three equations. Solving this linear equation system, the velocity coefficients and the velocity field can be determined.

Furthermore, the pressure distribution can be approximated using, Eq. 4.21,

$$\begin{aligned}
P = & P_0 + A_1x + A_2y + A_3z + B_1x^2 + B_2y^2 - \frac{K_x B_1 + K_y B_2}{K_z} z^2 + C_1xy + C_2xz + C_3yz \\
& + \frac{D_1}{3}x^3 + D_2xy^2 + D_3x^2y + \frac{D_4}{3}y^3 + D_5x^2z + D_6y^2z - \frac{D_1K_x + D_2K_y}{K_z} xz^2 \\
& - \frac{D_3K_x + D_4K_y}{K_z} yz^2 - \frac{D_5K_x + D_6K_y}{3K_z} z^3;
\end{aligned}$$

where the pressure coefficients can be determined using the velocity coefficients through Eq. 4.25 to 4.31, and the grid block pressure solved from the finite-difference method.

Next, the streamline can be generated using a numerical method.

If x changes monotonically along the streamline, $v_x \neq 0$ holds true within a local sub-cell, the streamline can be generated using x as the parameter to generate streamlines. Substituting the velocity approximation function Eq. 4.22, 4.23 and 4.24 into the mathematical definition of streamline Eq. 3.2 yields two coupled first-order differential equations,

$$\frac{dy}{dx} = \frac{a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2}{a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2}; \quad (4.51)$$

$$\frac{dz}{dx} = \left[\frac{a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2}{a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2} \right] \quad (4.52)$$

Given an arbitrary entry point (x_1, y_1, z_1) at the grid block, the streamline geometry can be calculated by solving the these coupled first-order differential equations simultaneously using the Runge-Kutta 4th order method (Appendix C). The exit point (x_2, y_2, z_2) of the streamline in the current grid block is determined by solving the intersection point between the streamline and the boundary of grid block.

Then, the incremental time-of-flight can be integrated numerically (for example trapezoidal method in Appendix D) using x as the parameter, *i.e.*,

$$\Delta\tau = \int_{x_1}^{x_2} \frac{dx}{v_x} = \int_{x_1}^{x_2} \frac{dx}{a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2}. \quad (4.53)$$

Similarly, if y changes monotonically along the streamline, the streamline can be generated using y as the parameter,

$$\frac{dx}{dy} = \frac{a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2}{a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2}; \quad (4.54)$$

$$\frac{dz}{dy} = \left[\frac{a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2}{a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2} \right]. \quad (4.55)$$

In this case, the incremental time-of-flight is integrated numerically using y as the parameter,

$$\Delta\tau = \int_{y_1}^{y_2} \frac{dy}{v_y} = \int_{y_1}^{y_2} \frac{dy}{a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2}. \quad (4.56)$$

Otherwise, z changes monotonically along the streamline, the streamline can be generated using z as the parameter,

$$\frac{dx}{dz} = \left[\frac{a_1 + b_1x + c_{1x}y + c_{2x}z + d_{1x}x^2 + d_{2x}y^2 + 2d_{3x}xy + 2d_{5x}xz - e_1z^2}{a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2} \right]; \quad (4.57)$$

$$\frac{dy}{dz} = \left[\frac{a_2 + b_2y + c_{1y}x + c_{3y}z + 2d_{2y}xy + d_{3y}x^2 + d_{4y}y^2 + 2d_{6y}yz - e_2z^2}{a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2} \right]. \quad (4.58)$$

The incremental time-of-flight is integrated numerically using z as the parameter, *i.e.*,

$$\Delta\tau = \int_{z_1}^{z_2} \frac{dz}{v_z} = \int_{z_1}^{z_2} \left[\frac{dz}{a_3 - (b_1 + b_2)z + c_{2z}x + c_{3z}y + d_{5z}x^2 + d_{6z}y^2 - 2(d_{1x} + d_{2y})xz - 2(d_{3x} + d_{4y})yz - (d_{5x} + d_{6y})z^2} \right]. \quad (4.59)$$

Following the Cubic algorithm introduced above, the exit coordinates and the incremental time-of-flight for the streamline with given entry point are determined for three-dimensional structured grid blocks. Streamline tracing results using the Cubic method are presented in the following Chapter together with the comparative results using the Pollock and Bilinear methods. These comparisons illustrate the advantages and limitations of each method.

Chapter 5

Comparisons of Different Streamline Tracing Methods

The Bilinear, Trilinear and Cubic methods have been introduced in Chapters 3 and 4. In this chapter, the accuracy of the Pollock, Bilinear, and Cubic methods are discussed by comparing 1) the pressure and velocity approximations with analytical solutions; and 2) streamline tracing results using various grid resolutions and reservoir properties with very accurate numerical solutions. Moreover, the abilities of the Bilinear, Trilinear and Cubic methods to deal with more realistic reservoir simulation problems are demonstrated using several case studies. Finally, the advantages and limitations of each method are addressed.

5.1 Comparisons of Pressure and Velocity Approximations with Analytical Solutions

Approximating the pressure distribution and velocity field everywhere in a reservoir is essential for tracing streamlines. Since streamlines are generated based on the approximated velocity field, the quality of pressure and velocity distributions directly affects the accuracy

of streamline tracing results. In this section, the pressure and velocity field approximations using Pollock’s method, Bilinear method and Cubic method are compared with the existing analytical solutions in a homogeneous quarter five-spot well pattern.

5.1.1 Analytical solution to the pressure and velocity field distribution

Five-spot well patterns are widely used in oil fields and commonly applied to examine the accuracy of streamline tracing methods. In this subsection, the analytical solutions for pressure and velocity field distributions with given boundary conditions are presented. These analytical solutions will be compared to the approximated results using different streamline tracing methods.

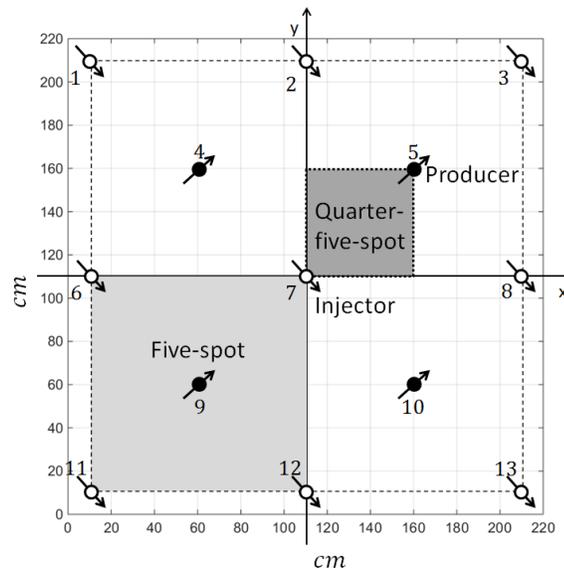


Figure 5.1: Five-spot well pattern

Consider a two-dimensional homogeneous five-spot pattern as shown in Figure 5.1, where four injectors are located at the corners of a square and a producer is located in the center. The distance between an injector and its adjacent producer is (50 cm, 50 cm). A quadrant of a five-spot pattern is called a quarter-five-spot pattern. When all wells are operating under the same flow rate, all boundaries in the quarter-five-spot are no flow boundaries because

of symmetry. Assuming that the reservoir is homogeneous, isotropic, and the single phase fluid is incompressible, then, the analytical pressure and velocity solutions can be obtained using line source solution and the principle of superposition (Caudle, 1966). The pressure distribution is given by,

$$P = -\frac{1}{4\pi} \frac{\mu}{Kh} \sum_i^n q_i \ln \left[(x - x_i)^2 + (y - y_i)^2 \right]; (n = 13), \quad (5.1)$$

where P is pressure; K is permeability, h is reservoir thickness; q_i is the volumetric flow rate of well No. i , positive for an injector and negative for a producer; n is the total number of wells; (x_i, y_i) is the location coordinate of well No. i .

Also, the analytical velocity field is given by,

$$v_x = -\frac{K}{\phi\mu} \frac{\partial P}{\partial x} = \frac{1}{2\pi h\phi} \sum_{i=1}^n q_i \frac{(x - x_i)}{(x - x_i)^2 + (y - y_i)^2}; \quad (5.2)$$

$$v_y = -\frac{K}{\phi\mu} \frac{\partial P}{\partial y} = \frac{1}{2\pi h\phi} \sum_{i=1}^n q_i \frac{(y - y_i)}{(x - x_i)^2 + (y - y_i)^2}; \quad (5.3)$$

$$v_t = \sqrt{v_x^2 + v_y^2}; \quad (5.4)$$

where v_x and v_y is x - and y - directional velocity field, respectively; v_t is total velocity; ϕ is porosity.

Table 5.1: The reservoir and fluid properties, and boundary conditions for the homogeneous quarter five-spot pattern

Average reservoir pressure	20.0 Pa
Permeability	1.0 D
Viscosity	1.0 cP
Porosity	1.0
Well flow rate for injectors	0.04 cm ³ /s
Well flow rate for producers	-0.04 cm ³ /s

Given the parameters in Table 5.1 and the well locations shown in Figure 5.1, the analytical pressure and velocity field distributions are shown in Figure 5.2. The x - and y - directional velocity fields are symmetric about the 45° diagonal. As mentioned before, these figures indicate that when all the wells are operating under the same flow rate, the five-spot

well pattern can be treated as repeated quarter-five-spots bounded by no-flow boundaries. Therefore, the quarter-five-spot patterns are applied to examine the accuracy of different streamline tracing methods in the following sections.

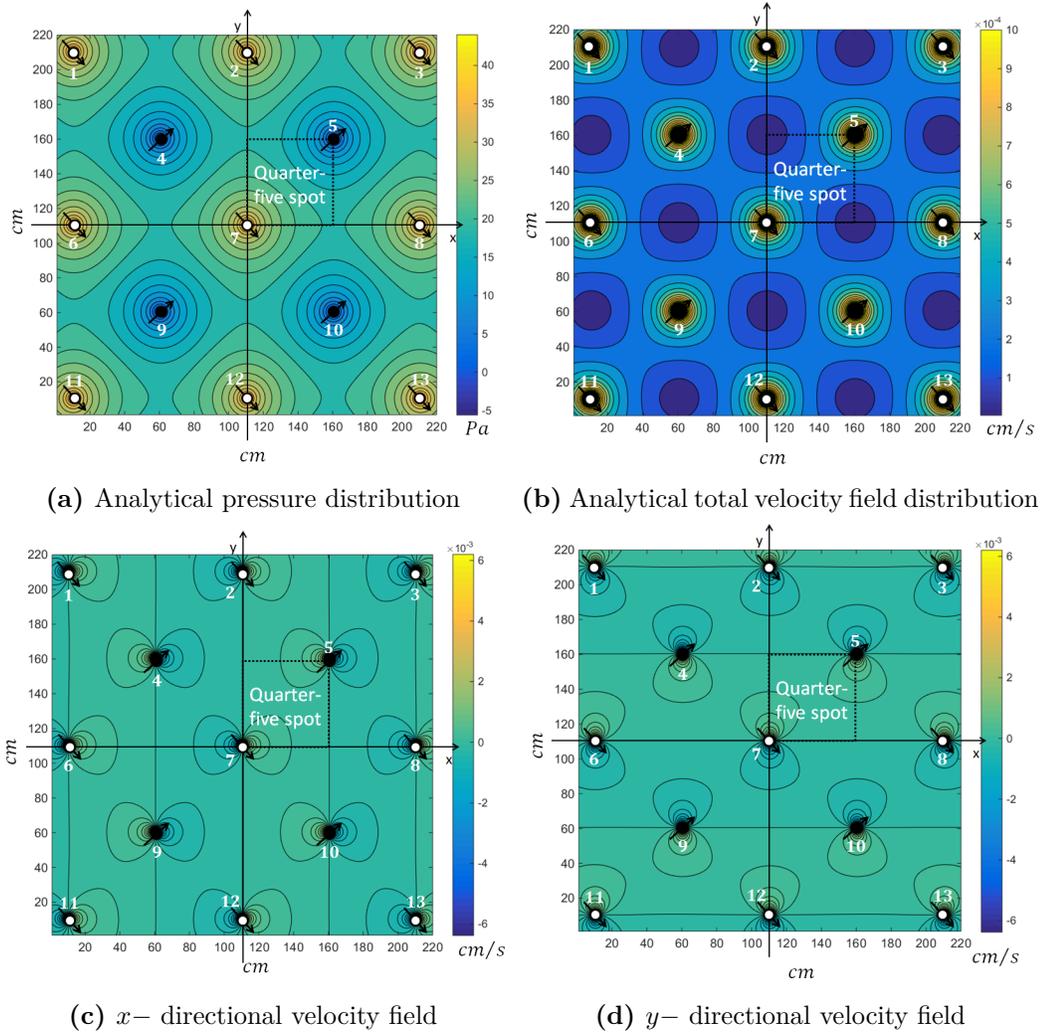


Figure 5.2: Analytical pressure and velocity field

5.1.2 Pressure distribution approximations

In this subsection, the pressure distribution approximation results obtained from Pollock, Bilinear, and Cubic methods are presented. The accuracy of the pressure approximations are quantified by calculating the relative errors compared to the analytical pressure solution.

Recall that the basic concept of the streamline tracing methods introduced in this research thesis is to apply piece-wise polynomial pressure approximation functions in streamline generations. The polynomial pressure functions applied by different methods for two-dimensional problems are summarized here. For Pollock's method (Eq. 4.1),

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_k y^2;$$

for Bilinear method (Eq. 3.14),

$$P = P_0 + A_1x + A_2y + Bxy;$$

and for Cubic method (Eq. 4.15),

$$P = P_0 + A_1x + A_2y + Bx^2 - Br_k y^2 + Cxy + Dxy^2 - Dx^2y - \frac{D}{3r_k}x^3 + \frac{Dr_k}{3}y^3.$$

In the above equations, the capital letters P_0, A_1, A_2, B, C, D are pressure coefficients; and $r_k = K_x/K_y$ is permeability ratio.

Using the same parameters in Table 5.1, the pressure distributions can be approximated for each method. More specifically, the quarter-five-spot reservoir is first discretized into (5×5) grid blocks, and then, the grid-centered finite-difference method is applied to solve for the pressure at grid blocks (Appendix A). Finally, the pressure distribution is approximated at each grid block through determining the pressure coefficients (P_0, A_1, A_2, B, C, D) for the functions listed above. The determination procedures for Pollock, Bilinear, and Cubic methods can be found in Appendix E, section 3.4.1 and 4.2.1, respectively. They all follow similar procedures by solving the linear equations with interpreted pressures or velocities at grid block interfaces.

We note that the pressure distributions in the injector and producer grid blocks are not approximated. This is because the pressure approximation functions in Pollock's, Bilinear, and Cubic methods assume velocity field in grid blocks are divergence free and have no injector or producer (source or sink term), as required by Eq. 3.10; thus, these functions are not suitable for pressure approximations in the injector and producer grid blocks. In

addition, streamlines are not generated within injector and producer grid blocks. When streamlines are close to wells, a near-well bore streamline tracing method (Johansen, 2010) using Polar coordinate system should be applied in order to obtain accurate results.

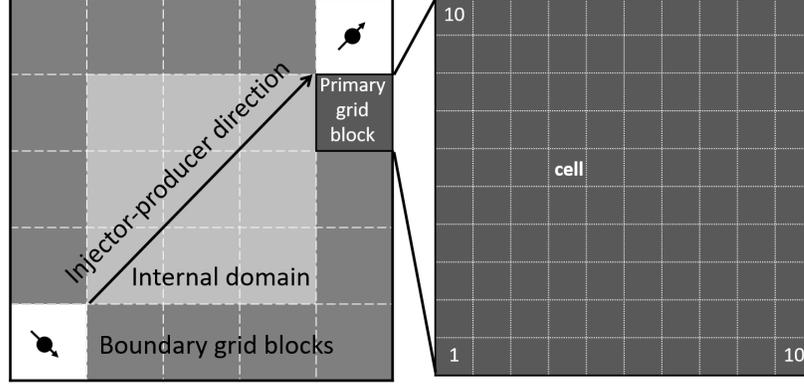


Figure 5.3: The pressure approximation domain, primary grid blocks and cells

In order to quantify the relative errors in pressure approximations, the averaged relative error compared to the analytical solutions is introduced. As shown in Figure 5.3, each of the primary grid block (23 original grid blocks, excepts the injector and producer grid blocks) is evenly divided into 100 square cells (2300 cells in total). The pressure at cell centers are approximated through substituting its coordinate into the approximate pressure functions (Eq. 4.1, 3.14 and 4.15). The averaged relative error in the pressure approximations rep is defined as,

$$rep = \frac{\sum_{i=1}^N rep_i}{N}, \quad (5.5)$$

where, i is the cell's number; N , ($N = 2300$) is total number of cells; and rep_i is the pressure approximation relative error in each cell i , given by,

$$rep_i = \frac{|P_i - P_{a,i}|}{\Delta P} \times 100\%; \quad (5.6)$$

where, P_i is the approximated pressure value at the center of cell i ; $P_{a,i}$ is the analytical pressure value at the center of cell i ; $\Delta P = \max(P_{a,i}) - \min(P_{a,i})$ is the greatest pressure difference in the approximation domain, named as the reference pressure.

Figure 5.4 shows the analytical pressure distribution in the reservoir. As can be observed,

the pressure distribution is smooth and continuous everywhere. It supports the basic assumptions of the new streamline tracing methods introduced in this research thesis, *i.e.*, the continuous pressure assumption made by the Bilinear method and the continuous pressure derivatives assumption made by the Cubic method.

The pressure approximation and relative errors obtained by using Pollock, Bilinear, and Cubic methods are given in Figure 5.5 to 5.6.

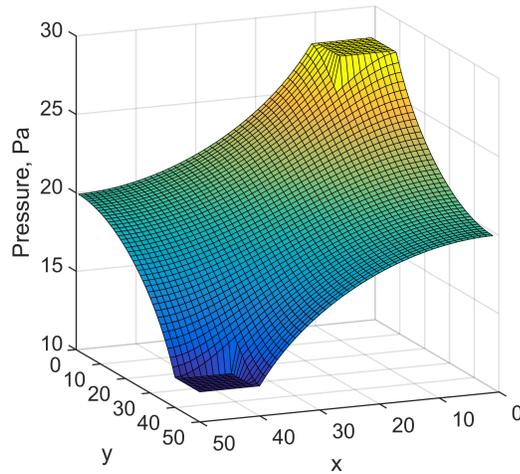


Figure 5.4: The analytical pressure distribution of the quarter-five-spot well pattern

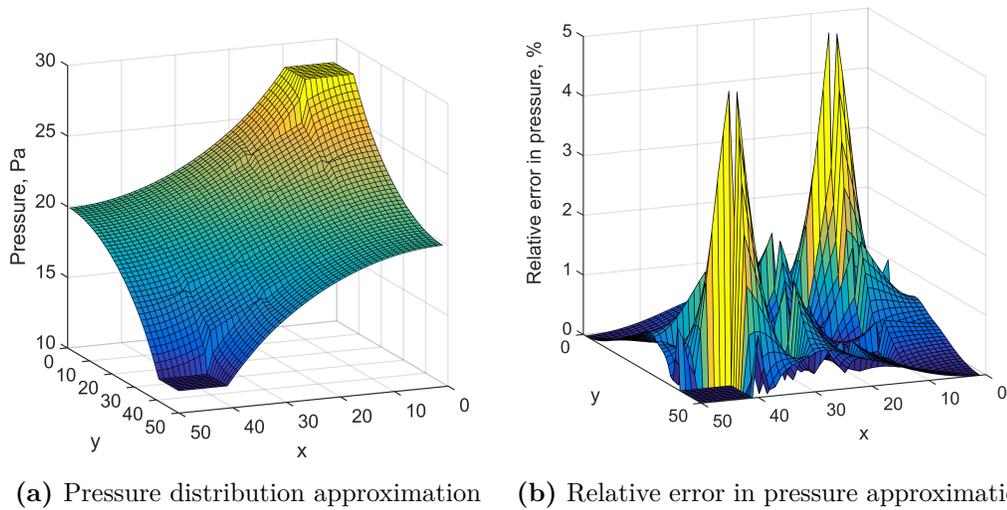


Figure 5.5: Pressure distribution approximation and its relative error obtained by using Pollock's method

Figure 5.5 shows that Pollock’s method may give discontinuous pressures across grid block interfaces. The discontinuities are severe along the injector-producer direction, where the pressure drop is the most significant. Based on this observation, the streamline tracing results of Pollock’s method are expected to give more errors along the injector-producer direction. On the other hand, the pressure distribution is very smooth at boundary grid blocks, where the streamline results are expected to be more accurate.

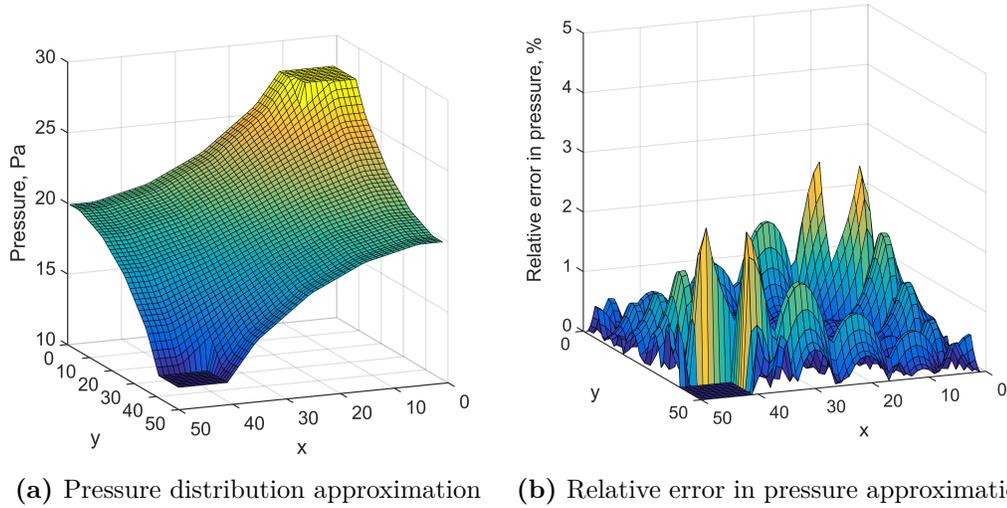


Figure 5.6: Pressure distribution approximation and its relative error obtained by using the Bilinear method

Figure 5.6 indicates that the Bilinear method achieves full pressure continuity in the reservoir. Besides, the Bilinear method approximates the pressure distribution in sub-cells instead of primary grid blocks, thus, the approximation accuracy is improved. The approximated pressure distribution varies sharply near to no-flow boundaries. This may lead to discontinuities in the velocity field. However, the pressure approximation relative errors in the internal domain are very small.

Figure 5.7 shows that the Cubic method gives a pressure distribution that is not fully continuous, however yields a very close approximation to the analytical solution. The pressure discontinuities across grid block interfaces are less severe compared to the Pollock results (Figure 5.5). For boundary grid blocks, the smoothness in the pressure distribution is not

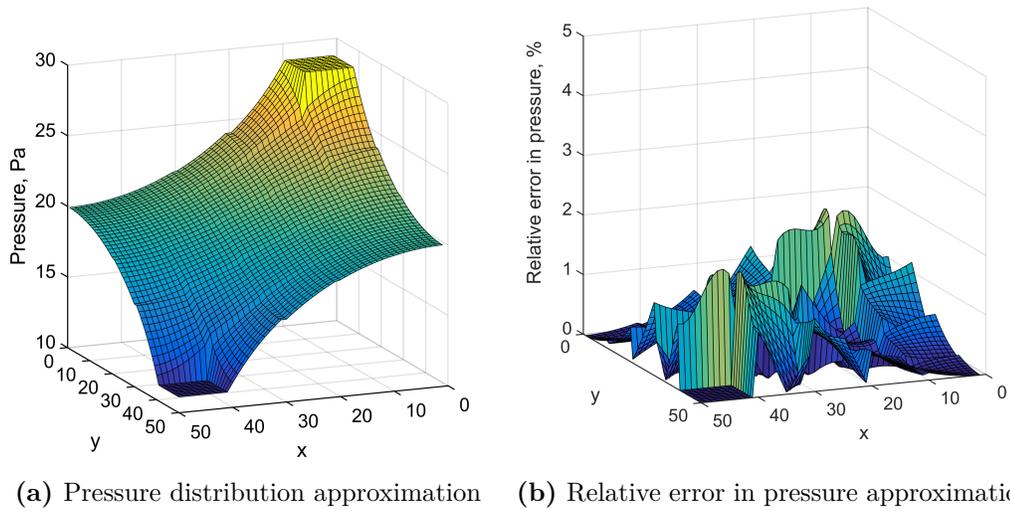


Figure 5.7: Pressure distribution approximation and its relative error obtained by using the Cubic method

fully achieved. The pressure approximations in the internal domain have small relative errors.

The averaged relative errors in pressure approximation rep (Eq. 5.6) for different streamline tracing methods are summarized in Table 5.2. Overall, the pressure approximations using these methods are all very accurate ($< 1.0\%$) even using low grid resolutions. The pressure approximation is a fundamental concept of accurate streamline tracing. Based on the above discussions on accurate pressure approximations, all of these streamline tracing methods are able to deliver accurate streamline tracing results when certain conditions apply.

Table 5.2: The relative error in pressure approximation using different streamline tracing methods

	Pollock method	Bilinear method	Cubic method
The relative error in the pressure approximation	0.43%	0.41%	0.40%

5.1.3 Velocity field distribution approximations

Following the same structure in the previous subsection, the velocity field distribution approximations from the Pollock, Bilinear and Cubic methods are presented in this subsection. The accuracy of an approximated velocity field has significant effects on the accuracy of streamlines, because streamlines are everywhere tangential to velocity field. The accuracy of velocity approximations are quantified by calculating the relative errors compared to the analytical velocity solution.

Velocity approximation functions can be obtained by differentiating the pressure functions according to Darcy's law. The velocity functions applied by different methods for two-dimensional problems are summarized here. For Pollock's method (Eq. 4.2 and 4.3),

$$\begin{aligned}v_x &= a_1 + bx, \\v_y &= a_2 - by;\end{aligned}$$

for the Bilinear method (Eq. 3.25 and 3.26),

$$\begin{aligned}v_x &= a_1 + c_1y, \\v_y &= a_2 + c_2x;\end{aligned}$$

and for the Cubic method (Eq. 4.16 and 4.17),

$$\begin{aligned}v_x &= a_1 + bx + c_1y + d_2y^2 - 2d_2xy - d_1x^2, \\v_y &= a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2.\end{aligned}$$

In the above velocity equations, the lower case letters $a_1, a_2, b, c_1, c_2, d_1, d_2$ are velocity coefficients.

The velocity field using the Pollock, Bilinear, and Cubic methods are approximated under the same conditions as for the above quarter-five-spot pattern. The velocity field distribution is approximated at each grid block through determining the velocity coefficients $(a_1, a_2, b, c_1, c_2, d_1, d_2)$ for the functions listed above. The determination procedures for the

Pollock, Bilinear, and Cubic methods can be found in Appendix E, section 3.4.1 and 4.2.1.

The comparisons between approximated and analytical velocities are performed at the center of 2300 cells (shown in Figure 5.3). The velocity at cell centers are approximated through substituting its coordinate into the approximation velocity functions (Eq. 4.2, 4.3, 3.25, 3.26, 4.16 and 4.17). The averaged relative error in velocity approximation rev is defined as,

$$rev = \frac{\sum_{i=1}^N rev_i}{N}, \quad (5.7)$$

where, i is the cell's number; N , ($N = 2300$) is the total number of cells; and rev_i is the velocity approximation relative error in cell i , given by,

$$rev_i = \frac{|v_i - v_{a,i}|}{\Delta v} \times 100\%; \quad (5.8)$$

where, v_i is the approximated velocity value at center of cell i ; $v_{a,i}$ is the analytical velocity value at center of cell i ; $\Delta v = \max(v_{a,i}) - \min(v_{a,i})$ is the greatest velocity difference in the approximation domain, named as the reference velocity.

The analytical velocity field solutions are shown in Figure 5.8. As can be observed, the analytical solutions for directional and total velocity fields are continuous and smooth surfaces in the entire reservoir domain. x - and y - directional velocity field are symmetric about the 45° diagonal. Thus, only the comparison results for y - directional (v_y) and total velocities are shown graphically in Figure 5.9, 5.10 and 5.11.

In Figure 5.9, the velocity v_y obtained from Pollock's method is continuous in y - direction while not in x - direction. This is because the velocity (v_y) approximation function of Pollock's method varies in its own direction (y - direction): recall the velocity function in Eq. 4.3,

$$v_y = a_2 - by;$$

i.e., v_y is a function of y only. The approximation relative errors in v_y and v_t are relatively large along the injector-producer direction compared to boundary grid blocks. However, the

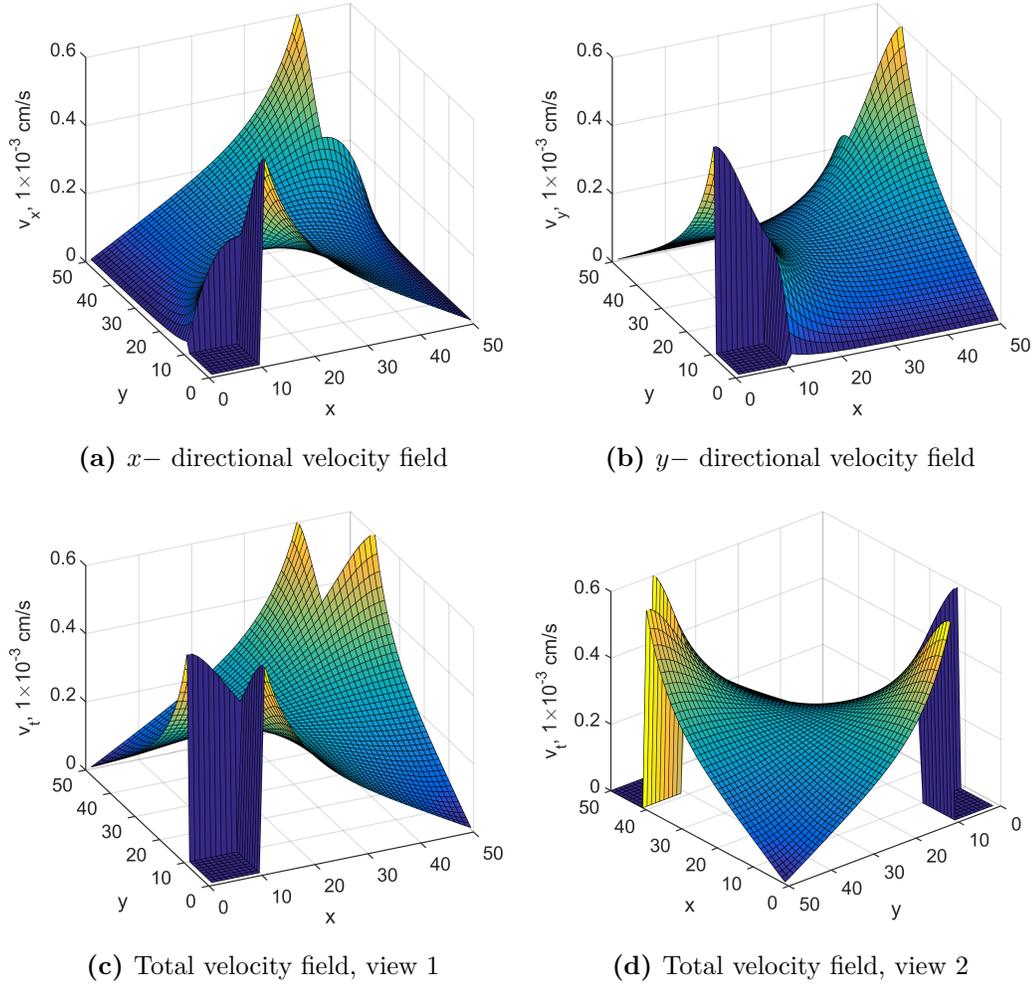


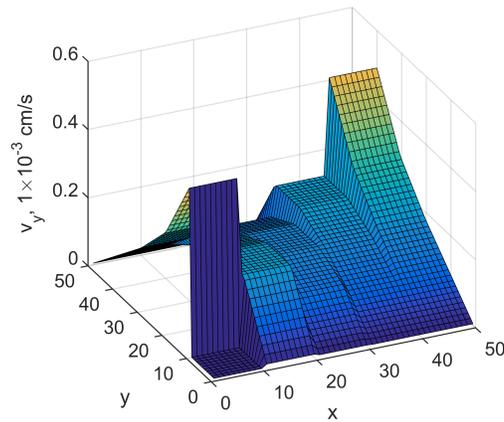
Figure 5.8: The analytical velocity field of the quarter-five-spot well pattern

smoothness in v_t along no-flow boundaries is achieved.

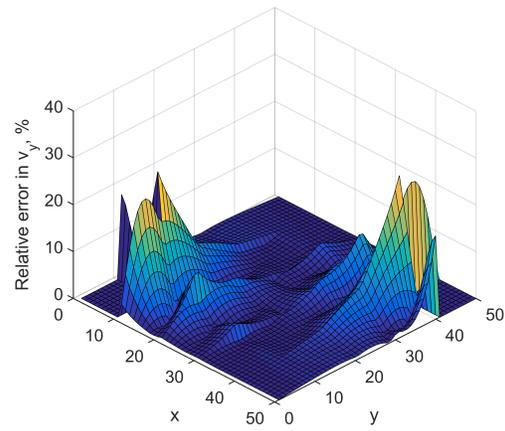
In Figure 5.10, the velocity v_y obtained from Bilinear method is continuous in x -direction while not in y -direction. This is because the velocity (v_y) function applied in the Bilinear method varies in its perpendicular direction (x - direction): recall the velocity function in Eq. 3.26,

$$v_y = a_2 + c_2x;$$

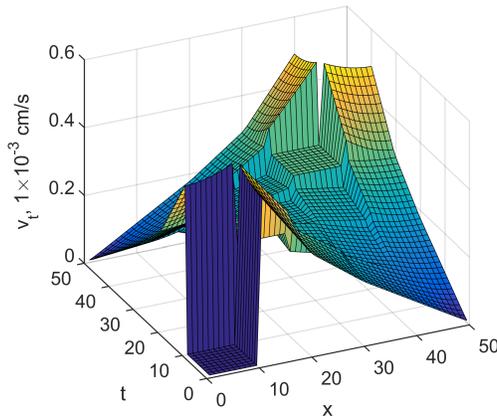
i.e., v_y is a function of x only. In boundary grid blocks, the relative errors in v_t are much larger compared to the internal region. The smoothness in v_t along no-flow boundaries is



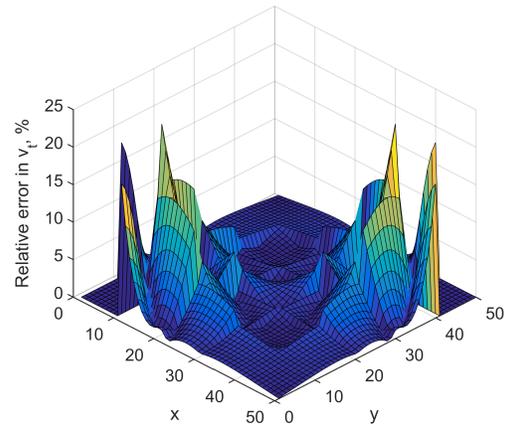
(a) y - directional velocity field approximation



(b) Relative error in v_y approximation



(c) Total velocity field approximation



(d) Relative error in total velocity field approximation

Figure 5.9: Velocity field approximation and its relative error obtained by using the Pollock method

not achieved.

As shown in Figure 5.11, the Cubic method gives the smallest relative error (see Table 5.3) in both v_y and v_t approximations. However, the velocity continuity is not achieved. The velocity v_y obtained from Cubic method varies in both x - and y -directions: recall the

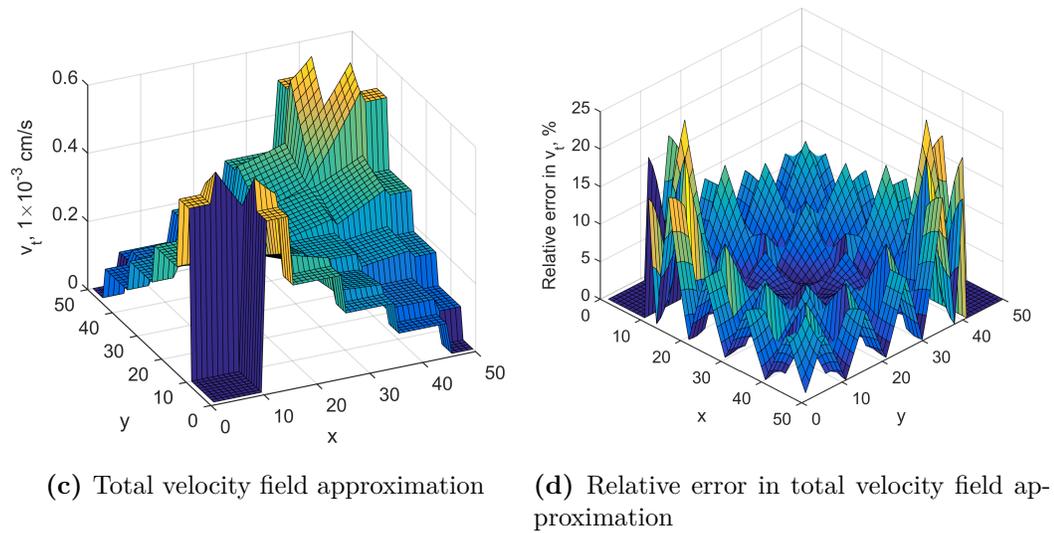
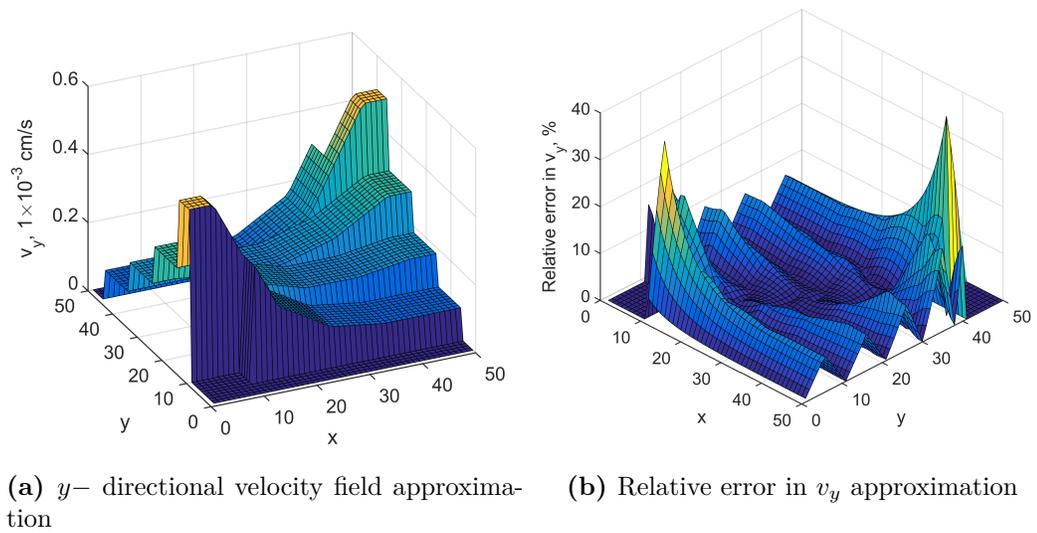
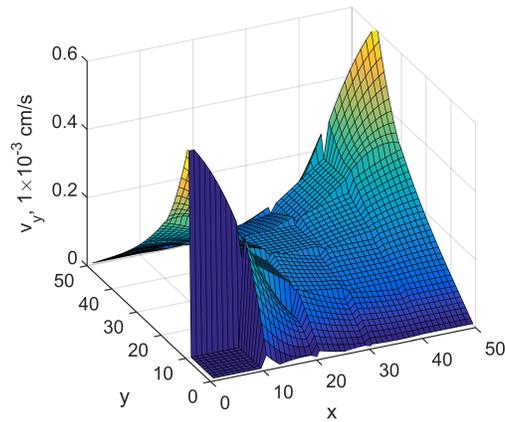


Figure 5.10: Velocity field approximation and its relative error obtained by using the Bilinear method

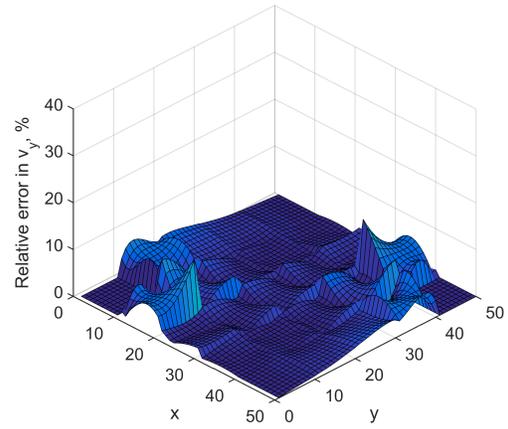
velocity function in Eq. 4.17,

$$v_y = a_2 - by + c_2x + 2d_1xy - d_1x^2 + d_2y^2;$$

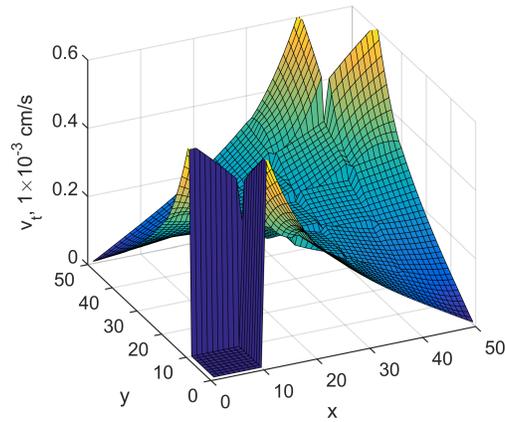
where v_y is a function of both x and y . The Cubic method delivers a better velocity approximation than the Bilinear and Pollock's methods. However, the velocity approximation relative errors in Cubic method are greater compared to Pollock's method in boundary grid blocks.



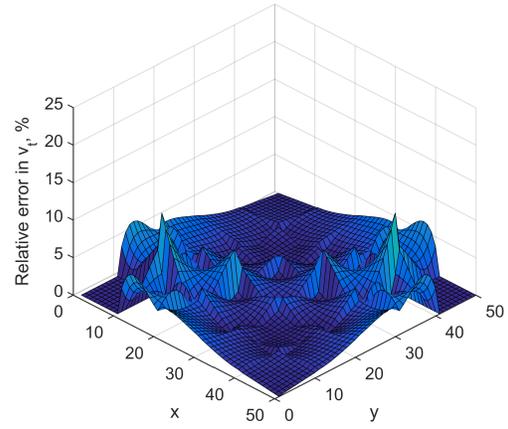
(a) y -directional velocity field approximation



(b) Relative error in v_y approximation



(c) Total velocity field approximation



(d) Relative error in total velocity field approximation

Figure 5.11: Velocity field approximation and its relative error obtained by using the Cubic method

The average relative errors in velocity approximation rev (Eq. 5.7) for different streamline tracing methods are summarized in Table 5.3.

In summary, the Pollock, Bilinear and Cubic methods can all accurately approximate the pressure and velocity distribution everywhere in this simple reservoir. The performance of Pollock's method is better in boundary grid blocks than in internal grid blocks; while, the Bilinear method is better in internal grid blocks than in boundary grid blocks. The results

Table 5.3: The relative error in velocity approximations using different streamline tracing methods

	Pollock's method	Bilinear method	Cubic method
The relative error in the v_x approximation	2.42%	4.24%	1.59%
The relative error in the v_y approximation	2.42%	4.24%	1.59%
The relative error in the v_t approximation	2.48%	4.16%	1.70%

obtained from Cubic method is the closest approximations to the analytical solutions, but continuity in pressure and velocity distributions is not achieved.

The pressure and velocity field distributions are closely related to streamline tracing results. In the next two sections, the errors in streamline tracing results obtained from the Pollock, Bilinear, and Cubic methods are quantitatively analyzed.

5.2 Comparisons of Streamline Tracing Results with Accurate Numerical Solutions

In this section, the performance of the Pollock, Bilinear, and Cubic methods in streamline tracing in two-dimensional reservoirs is evaluated.

According to the discussions made in section 5.1, the performance of the Pollock, Bilinear and Cubic methods in pressure and velocity approximations are different for internal and boundary grid blocks. Therefore, the performance of these methods are analyzed separately in these two different domains. An example of the two domains are shown in Figure 5.12.

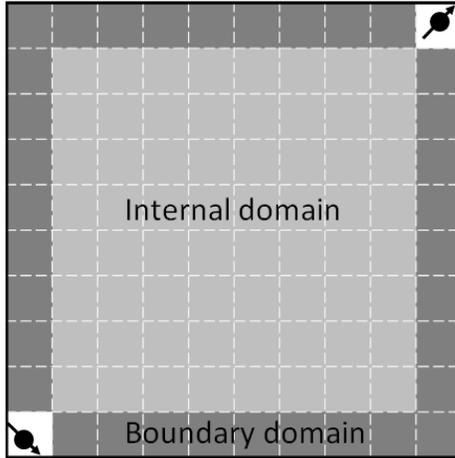


Figure 5.12: An illustration of internal and boundary domains in a 2D reservoir

5.2.1 Tracing streamlines in the internal reservoir domain

In this subsection, the performance of Pollock, Bilinear and Cubic methods are evaluated in different two-dimensional internal reservoir domains. These two-dimensional quarter-five spot reservoirs are defined with increasing complexities in permeability distributions. Streamlines will be traced in homogeneous, heterogeneous, and anisotropic (which is also a heterogeneous) reservoirs sequentially.

Case Study: tracing streamlines in a homogeneous reservoir

The streamline tracing performance for different methods are evaluated in the homogeneous case first, which we have the analytical solutions for pressure and velocities (Eq. 5.1 and 5.2). The reservoir, fluid properties and boundary conditions are given in Table 5.4.

Focusing on the streamline tracing in the internal reservoir domain, the streamline launching points and ending points are located away from boundary grid blocks. The launching points are selected to give evenly distributed streamlines in the reservoir. Given the parameters in Table 5.4, the true solutions for streamlines and time-of-flight are obtained by integrating the analytical velocity solution (Eq. 5.2). The streamline tracing procedures for the Pollock, Bilinear, and Cubic methods can be found in Appendix E, section 3.4.1 and 4.2.1,

Table 5.4: The reservoir, fluid properties and well controls for the homogeneous case

Reservoir dimension	100 cm × 100 cm
Permeability	1.0 D
Viscosity	1.0 cP
Porosity	1.0
Well flow rate at injectors	0.01 cm ³ /s
Well flow rate at producers	-0.01 cm ³ /s

respectively. The analytical and the approximated streamlines at a low grid resolution (10 × 10) are shown in Figure 5.13.

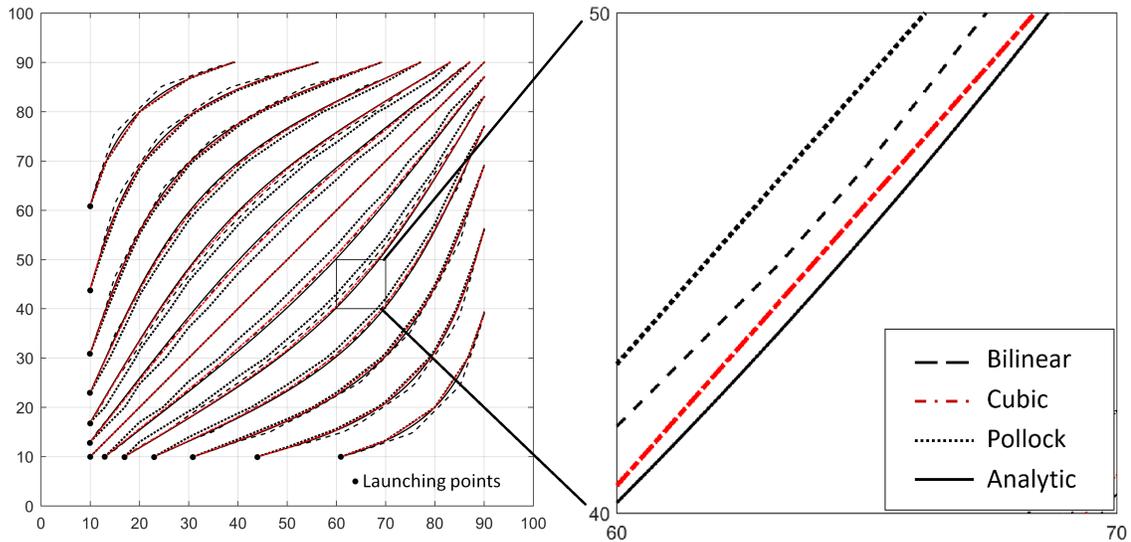


Figure 5.13: The analytical and approximated streamlines in the homogeneous case under the low grid resolution (10 × 10)

The difference between the streamlines launching from the same point can be visually observed in Figure 5.13. As shown in this figure, the distance from Pollock to analytical streamlines are relatively large compared to Bilinear and Cubic streamlines, especially along the injector-producer direction, where the pressure drop is the most significant. This is because the velocity approximation function of Pollock’s method over-simplifies the velocity field in this region, where the discontinuities of v_x in y -direction and v_y in x -direction are severe. The Cubic and Bilinear methods can approximate this behavior more accurately.

These observations are consistent with the conclusions made in section 5.1.

In order to quantify the errors in approximated streamlines, the distance from approximated to analytical streamlines (launching from the same point) are determined at five locations. Since the spatial coordinate of a point on a streamline can be represented by time-of-flight, these five points are located at given time-of-flight values. More specifically, as shown in Figure 5.14, the five points are located by three steps: first, evenly divide the total time-of-flight (true/analytical solution) into five intervals; second, calculate the time-of-flight value at the center of these intervals; third, locate the spatial coordinate of the five points in the approximated/analytical streamline with the known time-of-flight.

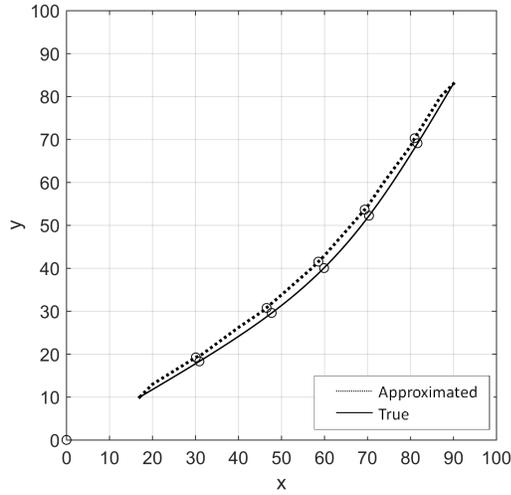


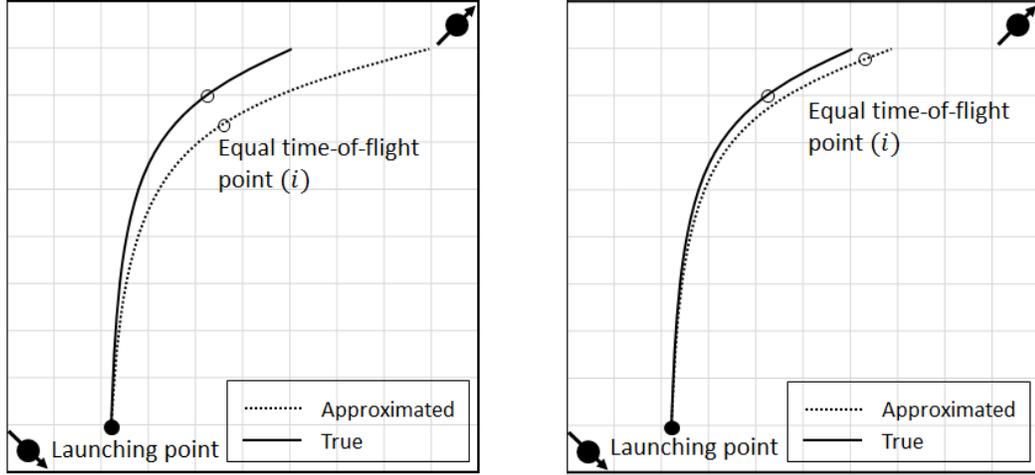
Figure 5.14: The distance between approximated and analytical streamlines at five equal time-of-flight points

We define the *Distance* (dl) between approximated and analytical streamlines at equal time-of-flight point (i) by,

$$dl_i = \sqrt{(x_i - X_i)^2 + (y_i - Y_i)^2}; \quad (5.9)$$

where, x_i and y_i are the coordinates for the point (i) at an approximated streamline; X_i and Y_i are the coordinates for the point (i) at an analytical streamline.

The *Distance* (dl) between approximated and analytical streamlines at equal time-of-flight point (i) evaluates two types of errors. First, as shown in Figure 5.15a, the approximated



(a) The error in streamline location determination (b) The error in time-of-flight determination

Figure 5.15: An illustration of errors in streamline location and time-of-flight determinations

streamline is far away from the true streamline, and *Distance* (dl) reflects the error in streamline location determination. Second, as shown in Figure 5.15b, although the approximated streamline is close to the true solution, the time-of-flight difference between them is large; in this situation, *Distance* (dl) reflects the error in time-of-flight determination.

The *Relative Distance* between the approximated and analytical streamlines in the entire domain is defined by,

$$DL = \frac{\sum_{i=1}^N dl_i}{N\Delta L}; \quad (5.10)$$

where, ΔL is the reference length; and N is the total number of equal time-of-flight points. In the example given in Figure 5.13, there are 13 streamlines, the number of equal time-of-flight points $N = 5 \times 13 = 65$. The reference length used in this research is the grid block length at (10×10) low grid resolution, $\Delta L = 10cm$.

The *Relative Distance* between approximated and analytical streamlines are dependent on grid resolution. As shown in Figure 5.16, the *Relative Distance* decreases as the grid resolution increases. Both the Bilinear and Cubic methods are more accurate than Pollock's

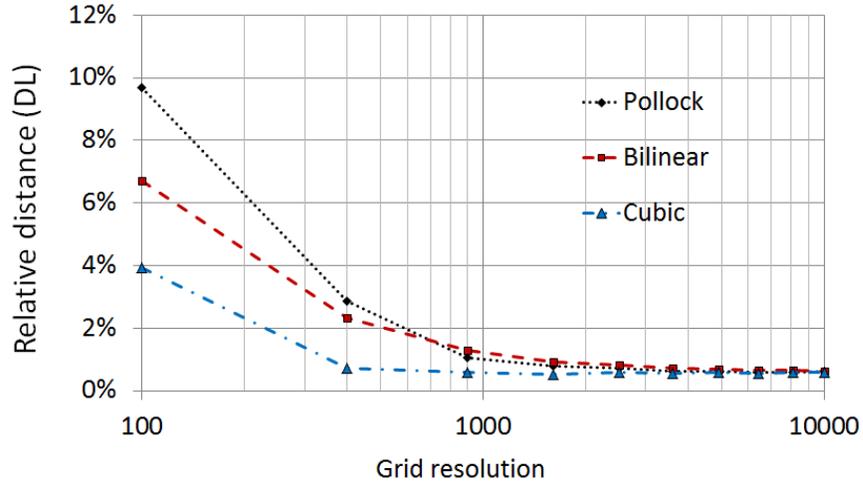


Figure 5.16: The *Relative Distance* between approximated and analytical streamlines in the homogeneous case under different grid resolution

method, especially when the grid resolution is low (less than 30×30). The *Relative Distance* at high grid resolution (100×100) is less than 0.8%, which is considered as sufficiently accurate. Less *Relative Distance* can be obtained when higher grid resolution is applied, however the convergence rate is very slow.

Table 5.5: The *Relative Distance* to analytical streamlines in the homogeneous case under low grid resolution (10×10)

	Pollock method	Bilinear method	Cubic method
<i>Relative Distance</i> to the analytical streamline	9.7%	6.7%	3.9%
The improvement in accuracy compared to Pollock's method		31%	60%

The *Relative Distance* to analytical streamlines at (10×10) grid resolution is summarized in Table 5.5. In this table, the improvement in accuracy compared to Pollock's method is calculated by using,

$$\frac{DL_{Pollock} - DL_{method}}{DL_{Pollock}} \times 100\%. \quad (5.11)$$

Based on this table, the Cubic method has significant advantages over the other methods.

The *Relative Distance* is less than 3.9% over the entire testing grid resolution range. Besides, at (10×10) grid resolution, the *Relative Distance* has decreased by 60% compared to Pollock’s method. This is because the Cubic method applied a high-degree streamline tracing method, which gives a more accurate velocity field; and the pressure derivative continuity assumption in the Cubic method holds true in this case.

The Bilinear method also shows better performance than Pollock’s method in the homogeneous case. When the grid resolution is 10×10 , the *Relative Distance* has decreased by 31% using the Bilinear method compared to Pollock’s method. According to the discussions made in section 5.1, this is because the Bilinear method is more accurate than Pollock’s method in pressure and velocity distribution approximations in internal reservoir domain.

From a CPU time usage point of view, as shown in Table 5.6, all of the streamline tracing methods achieve high efficiency ($<1.0s$ in high grid resolution). The Pollock method is the fastest approach, the Bilinear method is the second fastest, and the Cubic method requires the most computational effects. However, it is much less significant compared to the CPU usage in the transport problem simulations (as evaluated in Chapter 6), since streamlines are calculated only once before the time-consuming production simulation starts.

Table 5.6: The CPU time usage for the Pollock, Bilinear and Cubic streamline tracings in different grid resolutions

CPU time (s)			
Grid resolution	Pollock method	Bilinear method	Cubic method
10×10	0.05	0.08	0.20
100×100	0.12	0.59	0.80

Case Study: tracing streamlines in a heterogeneous reservoir

The streamline tracing performance for different methods are evaluated in the heterogeneous case next. The same fluid properties and well controls as applied in Table 5.4 are also applied in this case. The heterogeneous region is shown by the gray area in Figure 5.17, and the

permeabilities are:

Permeability in the reservoir is 1.0 D;

Permeability in the heterogeneous region is 0.5 D.

The same launching points for streamlines used in the homogeneous case are also used here. The analytical velocity solution is not applicable in the heterogeneous case, therefore, the numerical solutions obtained using the Bilinear method under the (100×100) grid resolution are applied as the true solutions. This is because the Bilinear method is the most accurate method compared to the other methods in this case (will be explained later in the text); and the difference between the Bilinear streamlines for (100×100) and (200×200) grid resolutions is less than 0.1%.

Given the above information, the true streamlines and the approximated streamlines using the Pollock, Bilinear and Cubic methods at a low grid resolution (10×10) are shown in Figure 5.17. As can be observed, the visual difference between streamlines is more obvious compared to the homogeneous case shown in Figure 5.13. The Bilinear streamlines are almost overlapping with the true solutions; whereas the distance between the Pollock and true streamlines are relatively large, especially in the heterogeneous region.

To quantify the errors in the approximated streamlines in this case, the *Relative Distance* between the true and approximated streamlines are determined at the five equal time-of-flight points using Eq. 5.9 and 5.10. The *Relative Distance* between the approximated and true streamlines are given in Figure 5.18 for increasing grid resolutions. Generally speaking, the *Relative Distance* between the Pollock and true streamlines has increased compared to the homogeneous case. Similar to the situation in the homogeneous case, both Bilinear and Cubic methods are more accurate than Pollock's method when the grid resolution is low (less than 30×30).

The *Relative Distance* to the true streamlines at the (10×10) grid resolution is summarized in Table 5.7. As can be observed, the Bilinear method gives the most accurate solutions.

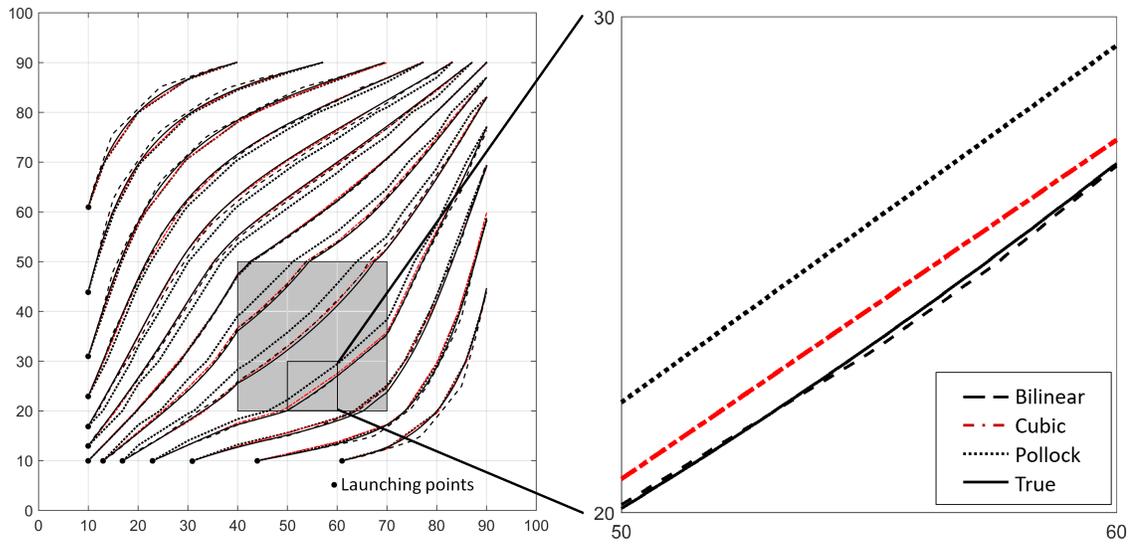


Figure 5.17: The true and approximated streamlines in the heterogeneous case under the low grid resolution (10×10)

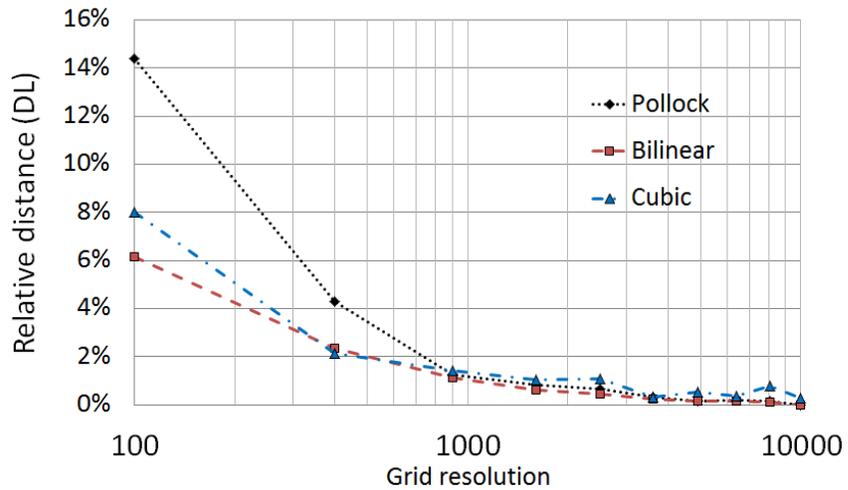


Figure 5.18: The *Relative Distance* between the approximated and analytical streamlines in the homogeneous case under different grid resolution

The *Relative Distance* given by the Bilinear streamlines in the heterogeneous case (6.2% in Table 5.5) is even less than the *Relative Distance* in the homogeneous case (6.7%). Moreover, the *Relative Distance* has decreased by 57% compared to Pollock's method. The advantages of approximating globally continuous pressure distribution and applying a dual-cell system is more obvious in the heterogeneous region compared to the homogeneous region. These

Table 5.7: The *Relative Distance* to the true streamlines in the heterogeneous case under the low grid resolution (10×10)

	Pollock method	Bilinear method	Cubic method
<i>Relative Distance</i> to the true streamline	14.4%	6.2%	8.0%
The improvement in accuracy compared to Pollock’s method		57%	44%

are part of the reasons for using the Bilinear streamlines under (100×100) grid resolution as the true streamlines.

The Cubic method also shows better performance than Pollock’s method in this case. When the grid resolution is 10×10 , the *Relative Distance* has decreased by 44% using the Cubic method compared to Pollock’s method. One of the reasons that the Cubic method is less accurate in this case is because the Cubic method has no explicit solutions for calculating time-of-flight, instead, a linear numerical integration is applied, which will induce some errors.

Case Study: tracing streamlines in an anisotropic reservoir

Finally, the streamline tracing performance for different methods are evaluated in the anisotropic case (which is also a heterogeneous case). The same fluid properties and well controls are applied as in Table 5.4. The anisotropic region is shown by the gray area in Figure 5.19, and the permeabilities are listed below:

Isotropic permeability in the reservoir is 1.0 D;

x– directional permeability in the anisotropic region is 0.2 D;

y– directional permeability in the anisotropic region is 0.3 D.

The same launching points for streamlines used in the homogeneous case are still applied here, and the Bilinear streamlines under the (100×100) grid resolution are also applied as

the true solutions. This is because the Bilinear method is the most accurate method compared to the other methods in this case; and the difference between the Bilinear streamlines for (100×100) and (200×200) grid resolutions is less than 0.1%.

Given the above information, the true streamlines and the approximated streamlines using the Pollock, Bilinear and Cubic methods at a low grid resolution (10×10) are shown in Figure 5.19. Comparing Figure 5.19 and 5.17, the visual difference between streamlines has increased as the complexity in permeability is increasing.

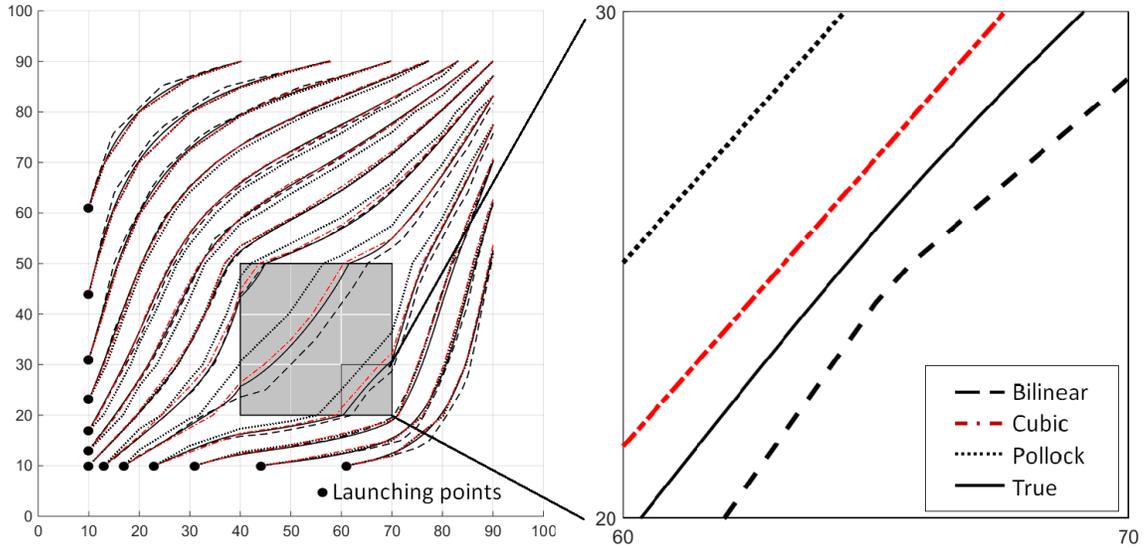


Figure 5.19: The true and approximated streamlines in the anisotropic case under the low grid resolution (10×10)

The *Relative Distance* between the true and approximated streamlines is determined to quantify the deviations. The results are given in Figure 5.20 for increasing grid resolutions. As can be observed, the *Relative Distance* between the Pollock and the true streamlines has increased significantly compared to the heterogeneous and the homogeneous cases. Still, both Bilinear and Cubic methods are more accurate than Pollock’s method when the grid resolution is low (less than 30×30).

The *Relative Distance* to the true streamlines at the (10×10) grid resolution is summarized in Table 5.8. Based on this table, the Bilinear method gives the most accurate solutions.

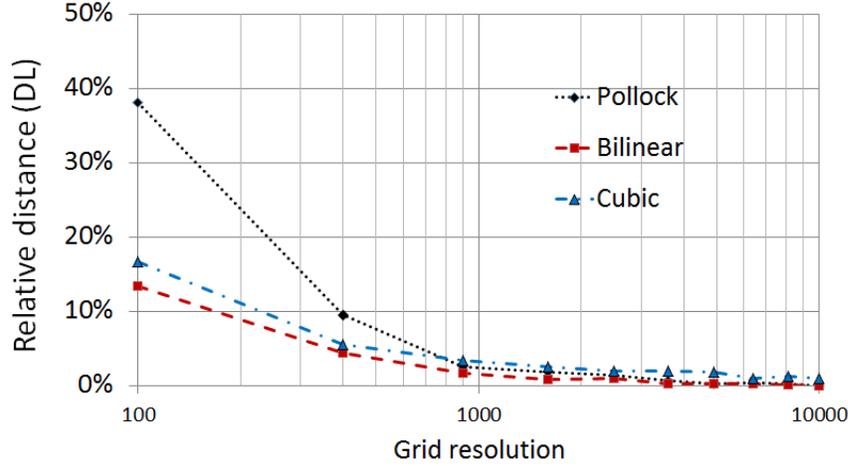


Figure 5.20: The *Relative Distance* between the approximated and analytical streamlines in the anisotropic case under different grid resolution

Table 5.8: The *Relative Distance* to the true streamlines in the anisotropic case under the low grid resolution (10×10)

	Pollock method	Bilinear method	Cubic method
<i>Relative Distance</i> to the true streamline	38.1%	13.5%	16.7%
The improvement in accuracy compared to Pollock’s method		65%	56%

By using the Bilinear method, the *Relative Distance* has decreased by 65% compared to Pollock’s method. This value is greater than the improvement in the heterogeneous case (57% in Table 5.7). The advantage of the Bilinear method is more significant when the permeability distribution in the reservoir becomes more complex.

The same situation is also found in the Cubic method. In this case, by using the Cubic method, the *Relative Distance* has decreased by 56% compared to Pollock’s method. This value is also greater than the improvement in the heterogeneous case (44% in Table 5.7).

Summary for this subsection

The difference between the streamlines generated by the Pollock, Bilinear and Cubic methods can be visually observed at a (10×10) grid resolution; they all converged at a high grid resolution (100×100) . Both the Cubic and Bilinear methods give more accurate results than Pollock's method when the grid resolution is lower than (30×30) . The Cubic method gives the most accurate solutions in the homogeneous case; the Bilinear method gives the most accurate solutions in the heterogeneous and anisotropic cases. The errors in the Pollock streamlines increase when the complexity of the permeability distribution increases. To the contrary, the advantages of using the Bilinear and Cubic methods are more obvious when the permeability distribution becomes more complex.

5.2.2 Streamline distributions in the entire reservoir

The above comparisons are based on the streamlines that travel within the internal reservoir domain. The performance for streamline tracing methods at the entire reservoir including boundary grid blocks will be evaluated in this subsection.

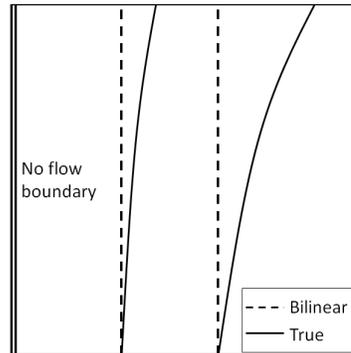


Figure 5.21: The Bilinear and the true streamlines in a no-flow boundary grid block

Both the Cubic and Pollock methods can be applied to trace streamlines in boundary grid blocks. However, the Bilinear method is not accurate for streamline tracing within boundary grid blocks when the grid resolution is low. The reason is given below. According to the

velocity functions applied in the Bilinear method (Eq. 3.25 and 3.26),

$$v_x = a_1 + c_1y,$$

$$v_y = a_2 + c_2x;$$

i.e., it assumes the velocity component is constant along its own direction. In boundary grid blocks, the normal velocity across the boundary is zero, and it will be zero in the entire grid block. As shown in Figure 5.21, it means that the streamlines launching from a no-flow boundary grid block will travel along the straight lines parallel to the boundary. This is true for the streamlines very close to the boundaries, however it will lead to errors when the grid resolution is low.

Since the boundary effects are more significant when the grid resolution is low, the accuracy of the Pollock and Cubic methods are evaluated at (10×10) grid resolution. The same fluid properties and well controls are applied as in Table 5.4. The launching points of streamlines are located at the injector grid block interfaces, and are selected to give evenly distributed streamlines in the reservoir.

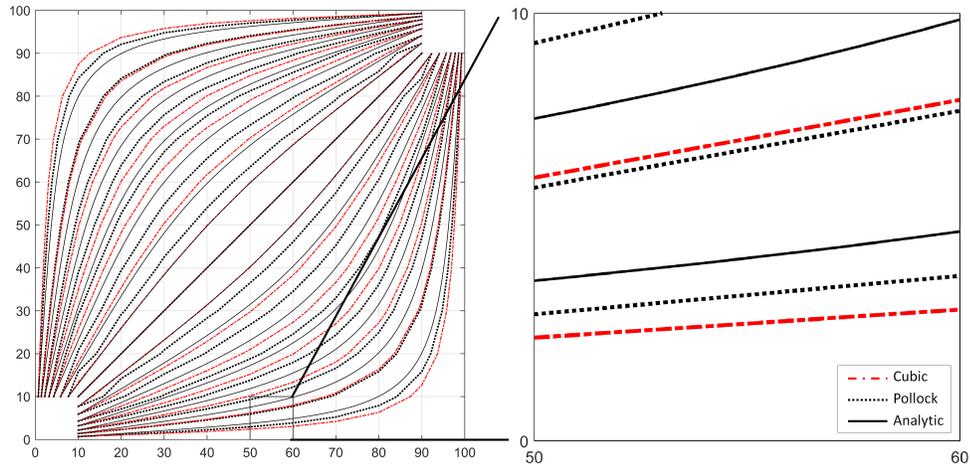


Figure 5.22: The analytical and approximated streamlines in the homogeneous reservoir with low grid resolution (10×10)

With the given information, the analytical, Pollock and Cubic streamlines are generated and shown in Figure 5.22. Similar observations for the internal domain streamlines (Figure 5.13)

can also be seen in these full field streamlines (Figure 5.22). In general, the Cubic method gives more accurate results than Pollock's method, especially along the injector-producer direction. The three Cubic streamlines (dash-dot lines) in the reservoir center overlap with the true streamlines (solid lines), while the Pollock streamlines (dot-dot lines) differ from the solid streamlines.

To quantify the accuracy of each approximated streamline, we number the streamlines in order, as shown in Figure 5.23, and calculate the *Relative Distance* (DL in Eq. 5.10, where $N = 5$) for individual streamlines. The results are shown in Figure 5.24.

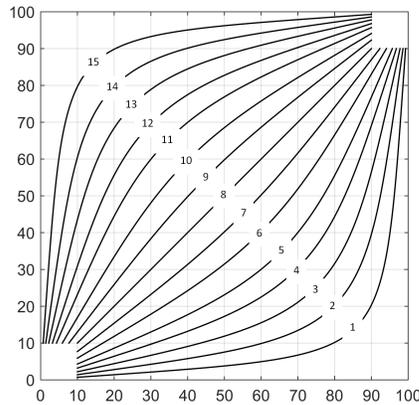


Figure 5.23: Streamline numbers

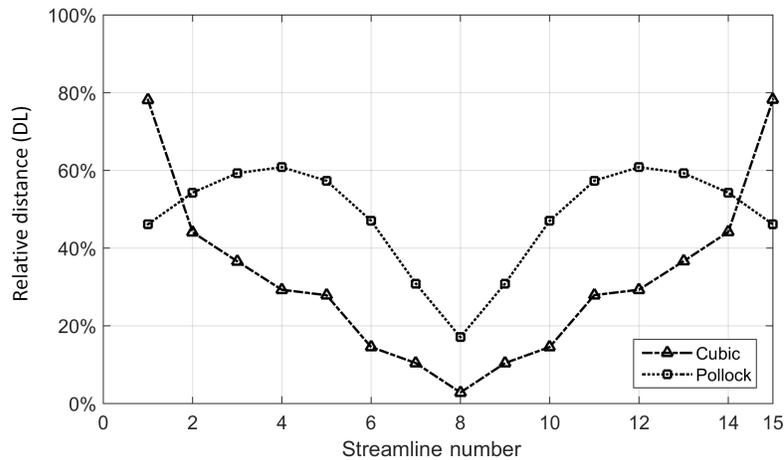


Figure 5.24: The *Relative Distance* for individual streamlines in the entire homogeneous reservoir with low grid resolution (10×10)

As can be observed in Figure 5.22 and 5.24, Pollock’s method gives better streamline tracing results than the Cubic method only when the launching point is very close to the boundary. This is because Pollock’s method can approximate no-flow boundaries accurately. The normal velocity across a no-flow boundary is identical to zero. The Cubic method has quadratic terms for velocity approximations, and therefore leads to small errors at no flow boundaries. This observation also agrees with the conclusions made in section 5.1.

However, this disadvantage is insignificant for most cases since only few streamlines traveling very close to no flow boundaries. Moreover, this advantage of Pollock’s method will be weaker when boundary grid blocks have heterogeneous properties. As an example shown in Figure 5.25, where the gray area has a higher permeability of $5 D$. For the streamline number 1, the Cubic streamline overlaps with the Pollock streamline.

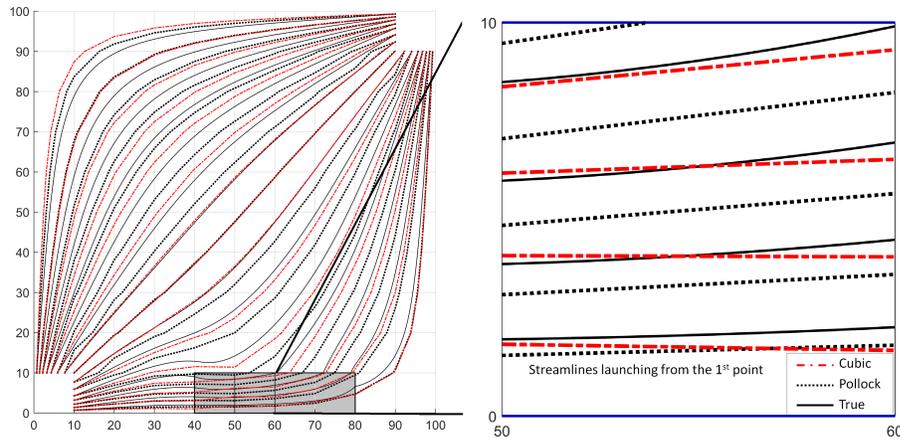


Figure 5.25: The true and approximated streamlines in the entire heterogeneous reservoir under the low grid resolution (10×10)

To quantify the accuracy of each streamline, the *Relative Distance* (DL in Eq. 5.10, where $N = 5$) for individual streamlines are calculated. The results are shown in Figure 5.26. As can be observed, comparing to the homogeneous case the accuracy of the Cubic method is improved for streamline number 1, which travels through the high permeable region. This is because the velocity field becomes more complex due to the effects of the heterogeneity, thus, the advantage of Pollock’s method in the no-flow boundaries becomes less important.

In addition, the no-flow boundary effects on the streamlines will become less important when the grid resolution increases. The *Relative Distance* for the approximated streamline number 1 (the streamline most close to a no-flow boundary) in a homogeneous reservoir are calculated under different grid resolutions. As shown in Figure 5.27, the difference between *Relative Distance* for the Pollock and Cubic streamlines decreases as the grid resolution increases.

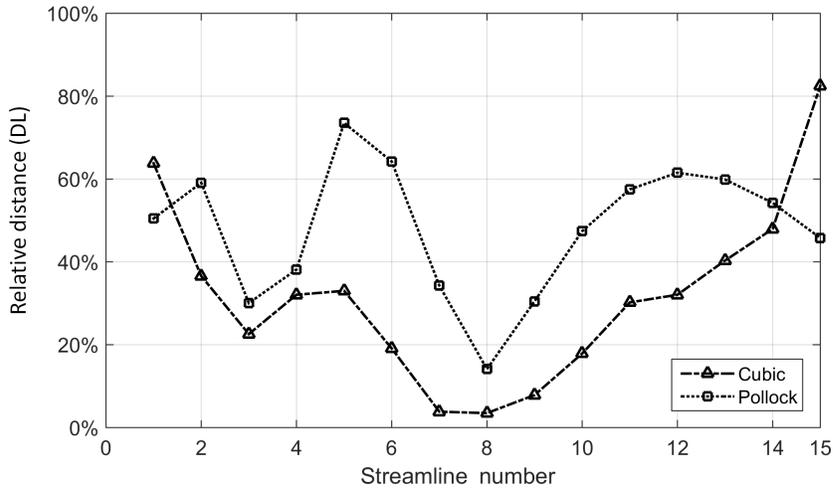


Figure 5.26: The *Relative Distance* for individual streamlines in the entire heterogeneous reservoir at low grid resolution (10×10)

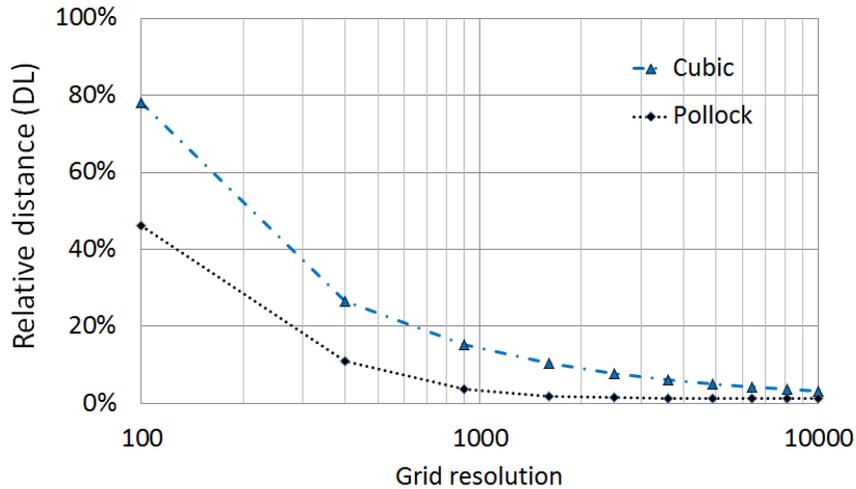


Figure 5.27: The *Relative Distance* for the approximated streamline No.1 to the analytical solution in a homogeneous reservoir under different grid resolutions

Moreover, as an other example shown in Figure 5.28, the Bilinear streamlines completely overlaps with the analytical streamlines. The no-flow boundary effects on the Bilinear streamlines can hardly be observed.

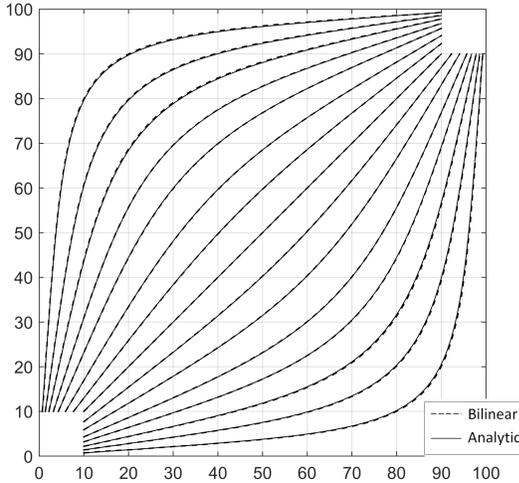


Figure 5.28: The analytical and Bilinear streamlines in a homogeneous reservoir under high grid resolution (100×100)

5.3 Demonstrations of the Bilinear, Trilinear and Cubic Streamline Tracing Methods

In this section, we demonstrate the ability of the tracing methods to deal with more realistic reservoir simulation problems including heterogeneous permeability fields and some special cases.

5.3.1 Layered reservoir

Many reservoirs are found to be composed of a number of layers whose characteristics are different from each other. Streamlines can visually illustrate the effects of different layers in the system. To better illustrate the effects of layers, a two-dimensional ($x - z$ plane)

example is shown in Figure 5.29, and the permeability for different layers are listed below:

Permeability in the reservoir is 1.0 D;

Permeability in the high permeable layer (light gray) is 5.0 D.

Permeability in the low permeable layer (dark gray) is 0.2 D.

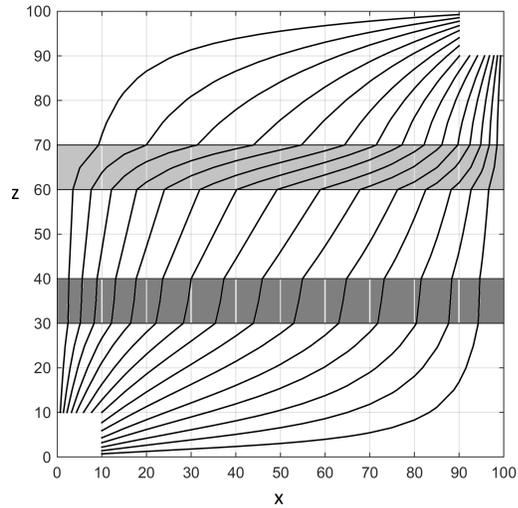


Figure 5.29: The streamline distribution in the layered reservoir (Cubic method)

The streamlines in Figure 5.29 are generated using the Cubic method with (20×20) low grid resolution. As can be observed, the streamlines become dense in the high permeable layer, and become more sparse in the lower permeable layer. The density of streamlines reflects the velocity of fluids. When streamlines become dense, the fluid velocity increases.

In addition, the streamlines tend to cross the low permeable layer in the perpendicular direction; while, the streamlines tend to stay in the high permeable layer and travel toward the producer. Therefore, in order to displace more fluids originally in the reservoir, the producer should be put in the high permeable layer; otherwise, more fluids will travel within the high permeable layer and less fluids will be recovered from other regions.

The similar effects of high and low permeable layers can also be observed in three dimensional cases. As an example of the three-dimensional five-spot pattern shown in Figure 5.30, four

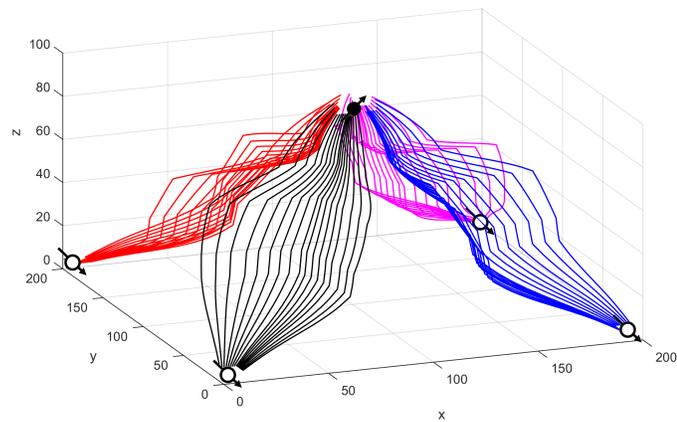
injectors are located in the four corners of the bottom layer, and one producer is located in the center of the top layer. The permeability for different layers are listed below:

Permeability in the reservoir is 1.0 D;

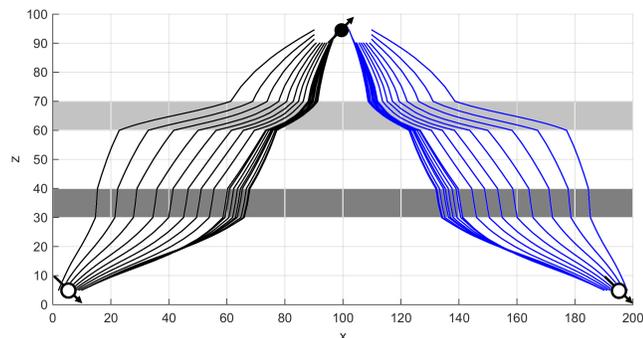
Permeability in the high permeable layer (light gray) is 5.0 D.

Permeability in the low permeable layer (dark gray) is 0.2 D.

The streamlines in Figure 5.30 are generated using the Cubic method under $(20 \times 20 \times 20)$ low grid resolution. As shown in the side view ($x - z$ plane), the streamlines become dense in the high permeable layer, and tend to stay in this layer and travel toward the producer; to the contrary, streamlines become more sparse in the lower permeable layer, and tend to leave this layer by crossing it perpendicularly.



(a) Full view



(b) Side view ($x - z$ plane)

Figure 5.30: The streamline distribution in the 3D layered reservoir (Cubic method)

Based on the above two cases, the ability of the Cubic method to trace streamlines in both two- and three- dimensional layered reservoirs under low grid resolutions is demonstrated.

5.3.2 Heterogeneous reservoir

Tracing streamlines can sometimes be very challenging in a heterogeneous reservoir. For example heterogeneity may cause streamlines to enter and exit a grid block at the same interface. Pollock's method cannot model this behavior correctly. On the other hand, when the assumption of a continuous pressure derivative is not valid in the physical reservoir, the Cubic method fails to trace streamlines accurately. The advantage of the Bilinear method is more obvious in these heterogeneous cases.

For example, as shown in the Figure 5.31, there are two non-permeable shale layers in the reservoir (shown by the black regions in this figure), and the streamlines are traced successfully using the Bilinear method under (50×50) grid resolution. Using the Cubic or Pollock methods cannot successfully trace streamlines under the same grid resolution.

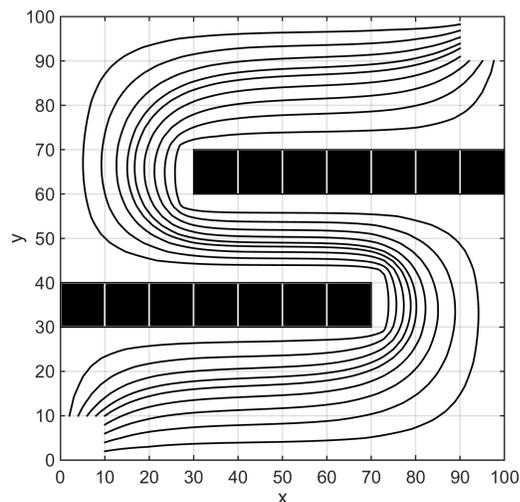


Figure 5.31: The streamline distribution in the reservoir has two shale layers (Bilinear method)

5.3.3 Coarse grid block in a large reservoir

One important feature of the Bilinear and Trilinear methods is their ability to approximate the globally continuous pressure distribution in the reservoir. This feature has a potential to save considerable simulation time when the pressure distribution in a large region can be accurately approximated using one bilinear/trilinear function. In this case, streamlines traveling through this region can be determined using a single coarse grid block.

Table 5.9: Reservoir, fluid properties, and well controls for the 3D homogeneous case

Reservoir dimension	$100\text{ cm} \times 500\text{ cm} \times 50\text{ cm}$
Grid resolution	$20 \times 100 \times 10$
Permeability	1.0 D
Viscosity	1.0 cP
Average reservoir pressure	20.0 Pa
Porosity	1.0
Well flow rate at injectors	$1.0\text{ cm}^3/\text{s}$
Well flow rate at producers	$-1.0\text{ cm}^3/\text{s}$

For example, for a homogeneous three-dimensional rectangular reservoir with given conditions listed in Table 5.9, the pressure distributions in bottom and top layers are shown in Figure 5.32. As can be observed, the pressure distribution in the large region bounded by solid lines is relatively flat, and can be approximated by a trilinear function. Thus, one coarse grid block may be applied to trace streamlines traveling within this region without losing much of the accuracy.

The streamline tracing results with and without the coarse grid block are shown in Figure 5.33. In this figure, the dashed lines are streamlines tracing within a coarse grid block, and the solid lines are streamlines tracing within sub-cells. Comparing these two results, not much visual difference can be observed. Therefore, the application of the coarse grid block in this reservoir model has proven to be successful.

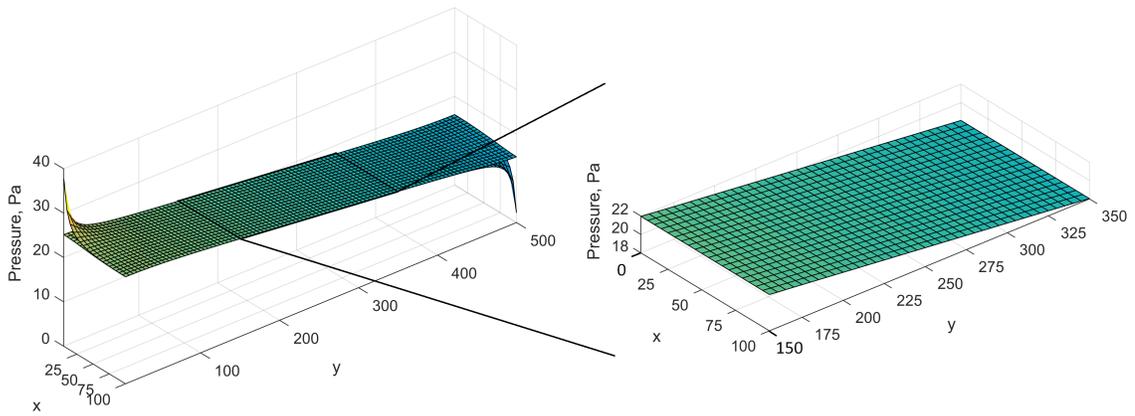


Figure 5.32: The pressure distribution in a rectangular reservoir

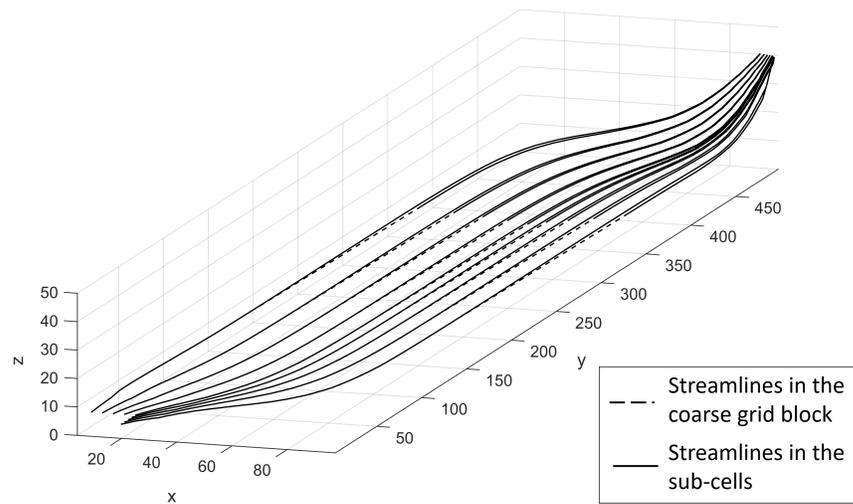


Figure 5.33: The streamline tracing results with and without the coarse grid block

5.4 Discussion and Conclusion of the Chapter

Up to this point, a new concept of tracing streamlines based on pressure approximation functions has been introduced. Through approximating pressure distribution at each grid block by a polynomial, the velocity field that directly defines streamlines can be determined. There are four requirements which the approximation polynomials must obey to give accurate solutions for a streamline:

1. The velocity approximation function must obey the velocity field divergence free condition;
2. The velocity approximation function is derived from the pressure function through Darcy's Law;
3. The pressure approximation function must obey the Laplace's equation;
4. The pressure and velocity functions are given by explicit functions.

Based on these requirements, the Bilinear, Trilinear, and Cubic methods are developed and introduced. These methods have their own advantages and limitations, that are summarized in the following paragraphs.

The Bilinear and Trilinear methods assume globally continuous pressure distribution, which meets the physical reality in the reservoir. They both apply a dual-cell system and approximate the pressure distribution at sub-cells rather than at primary grid blocks. This process naturally provides more information about pressure distribution, and improves the accuracy for velocity field approximation. Both Bilinear and Trilinear methods impose normal flux continuous conditions and divergence free condition over control volumes. The analytical solutions for the stream function and the time-of-flight are available in the Bilinear method, which reduces the numerical error and computational time for the streamline tracing using this method.

Compared to Pollock's method, the Bilinear method gives more accurate solutions for the pressure, velocity, and streamline determinations in the internal domain of the two-dimensional reservoir. This advantage is more obvious when the reservoir has heterogeneous properties. In some special cases, the heterogeneous reservoir will cause the streamlines to enter and exit in the same interface of a grid block. In this condition, only the Bilinear method can deliver accurate results. Moreover, the computational efficiency for the Bilinear method is high; it is only slightly slower than Pollock's method. One limitation of the

Bilinear method is that it may lead to errors for tracing streamlines in boundary grid blocks when the grid resolution is low.

The Trilinear method is also sufficiently accurate to trace streamlines in three-dimensional grid blocks. When the reservoir is relatively large, the Bilinear and Trilinear methods have a potential to reduce the computational time by using coarse grid blocks without losing much accuracy. This is achieved by using one coarse grid block where one bilinear/trilinear function can accurately approximate the pressure distribution in this region.

The Cubic method abandons the global pressure continuity and assumes the pressure distributions are third-degree polynomials. It determines the velocity field by Darcy's Law using the velocity approximation function derived from the third-degree pressure functions. The Cubic method is adaptable to a first-order finite difference method by using the velocity interpretation method that interprets the normal velocity at grid block interfaces. The underlying assumptions of the Cubic method are: first, the pressure derivative is continuous in its perpendicular direction; and second, the pressure derivative distribution in its perpendicular direction is a linear function in each grid block. After the normal velocities at grid block interfaces are known, the velocity field which defines streamlines can be approximated. This Cubic method has analytical solutions for streamline function in two-dimensional grid blocks, which reduces the numerical error and computational time for streamline tracing. This method can deliver sufficiently accurate streamline tracing results even with low grid resolution.

Since the Cubic method assumes third-degree pressure functions, it provides more coefficients for pressure and velocity approximations. This improvement for velocity field approximation is sometimes crucial when the velocity field varies significantly in two or three coordinate directions. Compared to the Bilinear and Pollock methods, the Cubic method gives the closest approximations to the analytical pressure and velocity distributions under the same grid resolution. The accuracy of the velocity field is important for multiphase flow simulations such as the Riemann approach along streamlines introduced in Chapter 6, since

the cross sectional area of the streamtube is determined by using the velocity.

Compared to Pollock's method, the Cubic method delivers more accurate streamlines. This advantage is significant in all tested cases in this research, including homogeneous, heterogeneous and anisotropic cases. The trade off is that the computational cost is increased compared to Pollock's method. Nevertheless, high computational efficiency is still achieved. For streamlines traveling very close to no flow boundaries, the Pollock method gives better results than the Cubic method. However, this disadvantage is insignificant for most cases, since only few streamlines traveling very close to boundaries. Moreover, it becomes less important when the reservoir has heterogeneous properties or the grid resolution increases.

The Cubic method is also very accurate to trace streamlines in three-dimensional grid blocks. The streamlines generated by the Cubic method can accurately illustrate the effects of different layers in three-dimensional reservoirs even with low grid resolutions.

An overall summary of Pollock, Bilinear, and Cubic methods is given in Table 5.10. The new streamline simulator developed in this research applies the Cubic method to trace streamlines, since it is more accurate than Pollock's method in velocity approximations and streamline generations. In the next chapter, a semi-analytical Riemann approach is introduced to solve transport problems along one-dimensional streamlines traced by the Cubic method.

Table 5.10: Advantages and limitations of Pollock, Bilinear, and Cubic methods

Method	Advantages	Limitations
Pollock	<p>Has analytical solutions for streamline location and time-of-flight in both 2D and 3D grid blocks;</p> <p>Achieves the highest computational efficiency compared to the other two methods;</p> <p>Has been extended to other streamline tracing methods;</p> <p>Is the most widely applied streamline tracing method in commercial reservoir simulators;</p>	<p>Will lead to large errors when it over-simplifies the velocity field in a coarse grid block.</p>
Bilinear	<p>Achieves full pressure continuity;</p> <p>Applicable to different grid structures and numerical methods;</p> <p>Much more accurate than Pollock's method in homogeneous, heterogeneous and anisotropic cases;</p> <p>Has analytical solutions for streamline location and time-of-flight in 2D grid blocks;</p> <p>Has a large potential to save considerable computational time in large reservoir models;</p> <p>Achieves computational high efficiency;</p>	<p>May lead to large errors in streamline tracing in boundary grid blocks when the grid resolution is low.</p>
Cubic	<p>Approximates the most accurate pressure and velocity distributions compared to the other two methods;</p> <p>Adapts to first-order finite-difference numerical methods;</p> <p>Much more accurate than Pollock's method in homogeneous, heterogeneous and anisotropic cases;</p> <p>Has analytical solutions for streamline location in 2D grid blocks;</p> <p>Achieves computational high efficiency;</p>	<p>The computational time is increased compared to Pollock's method;</p> <p>Less accurate than Pollock's method when streamlines travel too close to boundaries.</p>

Chapter 6

Solving Transport Problems along Streamlines Using A Semi-analytical Riemann Solver

In previous chapters, the streamline tracing methods based on polynomial pressure approximations were introduced to determine the geometry of streamlines in single phase flow. In this chapter, a semi-analytical Riemann approach is introduced to solve the two phase immiscible flow problem in a reservoir along streamlines/streamtubes.

In general, streamline methods for two phase flow (which refers to the methods applied to solving transport problems along streamlines/streamtubes) are effective approaches for reservoir simulation. They achieve high efficiency and are orders of magnitude faster than the conventional three-dimensional finite-difference based methods. In streamline-based methods, three-dimensional transport problems in grid blocks are decoupled to a series of one-dimensional problems along streamlines/streamtubes, in which analytical solutions can be applied. This one-dimensional transport problem can be solved by the Riemann approach. The Riemann approach refers to mapping the analytical Riemann solution along

the streamtubes or streamlines.

Analytical Riemann solutions in one-dimensional two phase immiscible flow problems with uniform initial conditions are solved by Buckley and Leverett (1942) under constant flow rate boundary conditions and by Johansen and James (2016) under constant pressure boundary conditions. Most streamline simulators apply the classical Riemann solution under constant total flow rates conditions. However, the boundary conditions are also usually specified by constant inlet/injection and outlet/production pressures. For instance, constant differential pressure flow is maintained during the rate decline period of reservoir depletion; wells in low permeability reservoirs are often, by necessity, produced at constant pressure; (Ehlig-Economides 1979); injectors usually operate below the fracking pressure, and producers are often operated under the constant pressure to prevent the reservoir pressure from dropping below the bubble point pressure.

Considering flow within streamtubes under constant pressure boundaries, Johansen and Liu (2016) presented the analytical Riemann solution for two phase multi-component flow in three-dimensional streamtubes. In their work, the analytical Riemann solution requires the cross-sectional area of streamtube. The explicit geometric shape for streamtubes in three-dimension sometimes can be very complex. In this research thesis, the effective cross-sectional area is approximated along the central streamline in each streamtube. Based on this approximation, a semi-analytical Riemann approach for constant pressure boundary along streamlines is presented in this chapter. This approach maps the analytical Riemann solutions along streamlines in terms of time-of-flight.

The Cubic streamline tracing method combined with the semi-analytical Riemann approach constitutes the new streamline simulator for two phase flow presented in this chapter. More specifically, to simulate a two-phase immiscible displacement process under constant pressure boundaries, this new streamline simulator has mainly three steps to accomplish:

1. The pressure equation is solved using the finite-difference method with given boundary

conditions;

2. The streamlines are traced using the Cubic method assuming single phase flow;
3. The two-phase transport problem is solved along each streamline using the semi-analytical Riemann approach.

A series of waterflooding processes with different reservoir and fluid properties are modeled using the new streamline simulator and the commercial petroleum reservoir simulator Eclipse 100 (Copyright Schlumberger, 2009-2015). The accuracy and efficiency of this new streamline simulator is evaluated through comparing the simulation results and the CPU usage with Eclipse 100 for the same waterflooding processes.

In this chapter, the general steps of streamline methods to solve the immiscible two phase flow Riemann problems in a reservoir along streamlines/streamtubes are firstly introduced in section 6.1; secondly, a semi-analytical Riemann approach is introduced to solve the Riemann problem under constant differential pressure boundary along streamlines in section 6.2; then, calculation examples and comparisons of the streamline simulator with Eclipse100 is given in section 6.4; finally, the methodology and results are summarized in section 6.5.

6.1 Immiscible Two Phase Flow Problems in a Reservoir

The immiscible two phase flow problems for conservation laws in porous medium are frequently encountered in reservoir simulations. In analytical solutions, the capillary and gravity effects are assumed to be negligible. Both assumptions can be partially overcome by an operator splitting technique proposed by Bratvedt *et al.* (1996). This technique is used in commercial streamline simulators (Thiele *et al.* 2010).

Consider the immiscible displacement of waterflooding process in an quarter-five-spot reservoir as depicted in Figure 6.1. In this figure, two solid lines form a streamtube, and the

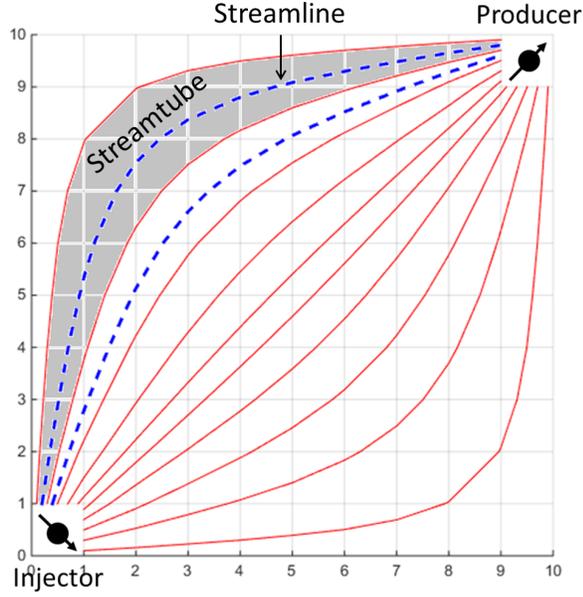


Figure 6.1: Solve waterflooding problems along streamlines/streamtube

dashed lines illustrate streamlines in two streamtubes. Generally speaking, immiscible flow problems in two- or three- dimensions are solved through three steps in streamline based methods: first, decouple the flow problem into one-dimensional transport problem along streamlines/streamtubes; then solve the transport problem along each streamline/streamtube analytically or numerically; finally, the solutions for entire reservoir is obtained by integrating the solutions from all streamlines/streamtubes.

For the transport problem in porous medium with constant initial and boundary conditions (usually called a Riemann problem), a Riemann approach can be applied to solve this problem by mapping the analytical Riemann solution along streamtubes/streamlines.

The mass conservation equation for a waterflooding Riemann problem in porous medium along a one-dimensional streamline/streamtube is (Buckley–Leverett, 1942) (the derivation of Eq. 6.1 is given in Appendix B.2),

$$\frac{\partial s}{\partial t} + \frac{q}{\phi A} \frac{\partial f}{\partial \xi} = 0, \quad (6.1)$$

$$q = -\lambda A(\xi) \frac{\partial P}{\partial \xi}; \quad (6.2)$$

with the boundary and initial conditions being

$$s(\xi, 0) = s_R, \text{ for } \xi \in [0, L], \quad (6.3)$$

$$s(0, t) = s_L, \text{ for } t \geq 0. \quad (6.4)$$

In the above equations, $s(\xi, t)$ is water saturation; t is time; ξ is arc-length of streamline; q is the total flow rate carried by a streamtube or its central streamline; ϕ is porosity; $A = A(\xi)$ is the cross sectional area of streamtube; $f(s) = \frac{\lambda_w}{\lambda}$ is the fractional flow function of water; $\lambda(\xi, s)$ is total mobility, *i.e.*,

$$\lambda = \lambda_w + \lambda_o; \quad (6.5)$$

where, $\lambda_w = \frac{Kk_{rw}}{\mu_w}$ and $\lambda_o = \frac{Kk_{ro}}{\mu_o}$ is the mobility for water and oil, respectively.

The analytical solutions for Riemann problems are available for two kinds of boundary conditions. They are the constant flow rate boundary condition, defined as

$$q(t) = q_L = \text{constant}, \text{ for } t \geq 0, \quad (6.6)$$

and the constant differential pressure boundary condition, defined as

$$\Delta P(t) = P_i(t) - P_o(t) = \text{constant}, \text{ for } t \geq 0; \quad (6.7)$$

where P_i and P_o is the pressure for injector and producer, respectively.

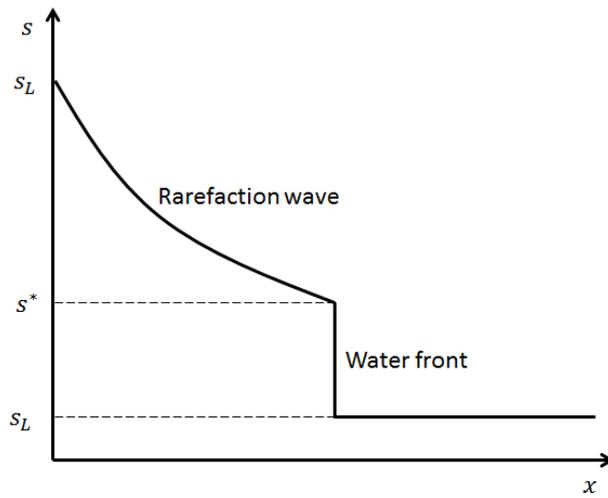


Figure 6.2: A typical saturation profile before water breakthrough

The Riemann problem under constant flow rate condition was solved by the Buckley-Leverett (1942) and is well described in literature. A detailed description and solution of this problem is given in Appendix B. As shown in Figure 6.2, before the water breakthrough, this solution consists of a leading shock with shock saturation s^* , and a rarefaction wave connecting s^* to s_L .

The one-dimensional Riemann problem along a streamtube under constant differential pressure condition was solved by numerical methods, until Johansen and Liu (2016) published the analytical Riemann solution. When the cross sectional area of a streamtube is known, the total flow rate $q(t)$ and the saturation profile $s(\xi, t)$ are determined analytically.

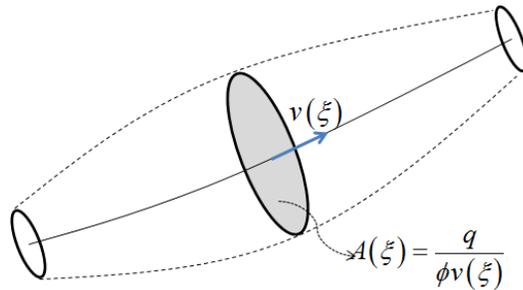


Figure 6.3: The relationship between a streamtube and its central streamline

Solving transport problems along streamlines are equivalent to streamtubes if one considers the relationship between a streamtube and its central streamline, as shown in Figure 6.3. The difference between these two methods is that the streamtube method gives explicit area $A(\xi)$; while the streamline method numerically determines $A(\xi)$ by the total flow rate q in a streamtube, and the total velocity at its central streamline $v(\xi)$, *i.e.*

$$A(\xi) = \frac{q}{\phi v(\xi)}. \quad (6.8)$$

In addition, the volume of a streamtube V from the launching point to location ξ along its central streamline is given by,

$$V(\xi) = \int_0^\xi \phi(\xi) A(\xi) d\xi. \quad (6.9)$$

Because of fluid incompressibility, the total flow rate q keeps constant as a function of ξ within a streamtube. Substituting the cross sectional area equation Eq. 6.8 into the above

equation gives,

$$V(\xi) = q \int_0^\xi \frac{1}{v(\xi)} d\xi. \quad (6.10)$$

Recall the mathematical definition of time-of-flight τ at the location ξ along the central streamline,

$$\tau(\xi) = \int_0^\xi \frac{1}{v(\xi)} d\xi. \quad (6.11)$$

Thus, the volume of a streamtube is related to its central streamline in terms of time-of-flight (Batycky, 1997),

$$V(\xi) = q\tau(\xi). \quad (6.12)$$

The advantages of using streamlines instead of streamtubes are that the total velocity $v(\xi)$ along streamlines is easier to obtain than the explicit cross sectional area $A(\xi)$ of streamtubes; and the time-of-flight τ along streamline is easier to calculate than the integration form of streamtube volume in Eq. 6.9. The trade off for using central streamlines is that the boundary and the cross-sectional area of streamtube are approximated numerically by ignoring the complicated geometry of streamtube, and it may lead to errors when the total number of streamlines is not sufficiently large. Since the cross sectional area is determined numerically, this Riemann approach along streamlines is semi-analytical.

The main objective of this chapter is to solve the Riemann problem under constant pressure boundary along streamline by using a semi-analytical Riemann solver.

6.2 A Semi-analytical Riemann Solver under Constant Pressure Boundary

As discussed above, streamline methods solve displacement problems along one-dimensional streamlines. Since the pressure distribution keeps fixed for a long time interval, the pressure

solution only needs to be updated a few times throughout a displacement process (Batycky, 1997). The geometry of streamlines is assumed to be fixed in a given time interval in a two-phase displacement process.

In a streamline simulation process, streamlines are first traced for the single phase flow under the same boundary conditions using the Cubic method introduced in Chapter 4. During this step, the following values are defined: the *time-of-flight* τ , the *velocity* along a streamline v , and the *reference flow rate* associated with a streamline q_r . The *time-of-flight* τ and the *velocity* v values are time independent and only correspond to the streamline arc-length ξ . The *reference flow rate* associated with a streamline q_r is a constant value. It is important to note that the *time-of-flight* τ refers to a spatial coordinate in the streamline, since it corresponds to a unique point on the streamline. These values (q_r , τ and v) simplify the process of mapping Riemann solutions along streamlines. When streamlines are required to be updated, these reference values are also updated together and used in the following calculations.

The Riemann problem under constant differential pressure along a one-dimensional streamline is described as Eq. 6.1 and 6.2,

$$\begin{aligned} \frac{\partial s}{\partial t} + \frac{q}{\phi A} \frac{\partial f}{\partial \xi} &= 0, \\ q(t) &= -\lambda(\xi, s) A(\xi) \frac{dP}{d\xi}; \end{aligned}$$

with constant conditions defined by Eq. 6.3, 6.4, and 6.7

$$\begin{aligned} s(\xi, 0) &= s_R, \text{ for } \xi \in [0, L], \\ s(0, t) &= s_L, \text{ for } t \geq 0; \end{aligned}$$

$$\Delta P(t) = P_i(t) - P_o(t) = \text{constant}, \text{ for } t \geq 0;$$

where, the cross sectional area is independent of time and is approximated by Eq. 6.8,

$$A(\xi) = \frac{q_r}{\phi v(\xi)}.$$

The main result of this section is that the Riemann problem has semi-analytical Riemann solutions for $q(t)$ and $s(\xi, t)$ in terms of time-of-flight. More specifically:

1) the total flow rate $q(t)$ along a certain streamline for the constant differential pressure Riemann problem described above is a function of time, given by,

$$q(t) = \frac{q_r \Delta P}{R(t)}; \quad (6.13)$$

where, q_r is the reference flow rate in the streamline; $R(t)$ is the total flow resistance of the streamline, defined as,

$$R(t) = \int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s)} d\tau; \quad (6.14)$$

where, τ^R is the time-of-flight at the outlet of streamline; v is the velocity along streamline;

2) the saturation $s(\xi, t)$ behind the water front (leading shock) is given by,

$$s(\xi, t) = f'^{-1} \left(\frac{q_r \tau(\xi)}{Q(t)} \right), \quad (6.15)$$

where, $\tau(\xi)$ is the time-of-flight of one location ξ in the streamline; f'^{-1} is the inverse function of f' ; $Q(t) = \int_0^t q(t) dt$ is the cumulative water volume injected to the streamtube at time t .

Detailed explanations for these results are given in the following sections.

6.2.1 Derivation of the total flow rate as a function of time

In the following paragraphs, the derivation of the total flow rate Eq. 6.13,

$$q(t) = \frac{q_r \Delta P}{R(t)}$$

is given.

Since transport problems in porous medium obey Darcy's law, the total flow rate q along a certain streamline is given by,

$$q(t) = -\lambda(\xi, s) A(\xi) \frac{\partial P}{\partial \xi}. \quad (6.16)$$

According to the chain rule, the above equation can be derived as,

$$q(t) = -\lambda(\xi, s) A(\xi) \frac{\partial P}{\partial \tau} \frac{\partial \tau}{\partial \xi}. \quad (6.17)$$

Since the cross sectional area is assumed to be independent with time, it can be approximated by Eq. 6.8,

$$A(\xi) = \frac{q_r}{\phi v(\xi)}.$$

In addition, based on the mathematical definition of the time-of-flight,

$$\tau = \int \frac{1}{v} d\xi, \quad (6.18)$$

we can obtain,

$$\frac{\partial \tau}{\partial \xi} = \frac{d\tau}{d\xi} = \frac{1}{v(\xi)}. \quad (6.19)$$

Substituting Eq. 6.8 and Eq. 6.19 into Eq. 6.17 yields,

$$q(t) = -\frac{\lambda(\xi, s) q_r}{\phi(\xi) v(\xi)^2} \frac{dP}{d\tau}. \quad (6.20)$$

Integrating the above equation from the inlet to the outlet of the streamline under the constant differential pressure boundary gives,

$$q(t) \int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s)} d\tau = -q_r \int_{P_i}^{P_o} dP; \quad (6.21)$$

and we can obtain Eq. 6.13,

$$q(t) = \frac{q_r \Delta P}{\int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s)} d\tau} = \frac{q_r \Delta P}{R(t)};$$

where ΔP is the total pressure drop of the streamline, and τ^R is the time-of-flight at the outlet of the streamline.

In the above equation, the numerator is a constant value, and the denominator is a function of time. Here, the denominator is denoted as the total flow resistance of the streamline $R(t)$, as given in Eq. 6.14,

$$R(t) = \int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s)} d\tau.$$

The physical meaning of the total flow resistance is the differential pressure required for keeping the flow rate at the reference value (q_r). We note that the resistance R introduced here is similar to the geometric resistance G defined by Higgins and Leighton (1962a), the flow rate is inversely proportional to both of them. The difference is, the resistance R accounts for the effects of total mobility and streamtube geometry; while the geometric resistance G in Higgins and Leighton (1962a) is a shape factor, it only accounts for the effect of streamtube geometry.

The parameters in Eq. 6.13 can be divided into two groups: either saturation dependent, or saturation independent. The saturation dependent parameters are determined later in this section when the saturation profile is known; the saturation independent parameters are determined when streamlines are traced during the single phase flow period. To begin with, the *velocity* and pressure distributions in x -, y - and z - directions are approximated using the Cubic method everywhere in the reservoir; the *time-of-flight* for streamlines and streamline locations are also determined at the intersection points between grid blocks and streamlines.

The streamline launching points can be uniformly distributed at the launching grid block interface, and the *reference flow rate* assigned to each streamline can be determined by surface integration,

$$\left\{ \begin{array}{l} q_{r,x} = \int \phi v_x dydz, \\ q_{r,y} = \int \phi v_y dx dz, \\ q_{r,z} = \int \phi v_z dx dy, \end{array} \right. \quad (6.22)$$

where $q_{r,x}$, $q_{r,y}$ and $q_{r,z}$ are the *reference total flow rate* of a streamline that launch from the grid block interface x , y and z ; v_x , v_y and v_z are *velocities* defined during single phase flow in x -, y - and z - directions.

The *total velocity* v is calculated using,

$$v = \sqrt{v_x^2 + v_y^2 + v_z^2}. \quad (6.23)$$

The total mobility $\lambda(\xi, s)$ is a function of saturation s and location along streamlines ξ , *i.e.*,

$$\lambda(\xi, s) = K(\xi) \left(\frac{k_{ro}(s)}{\mu_o} + \frac{k_{rw}(s)}{\mu_w} \right), \quad (6.24)$$

where k_{ro} and k_{rw} is the relative permeability for oil and water, respectively; μ_o and μ_w is the viscosity for oil and water, respectively; $K(\xi)$ is the directional permeability along streamline, given by,

$$K(\xi) = \mathbf{n}_v^T \mathbf{K} \mathbf{n}_v. \quad (6.25)$$

where \mathbf{n}_v is the unit vector of velocity vector, given by,

$$\mathbf{n}_v = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \begin{bmatrix} \frac{v_x}{v} \\ \frac{v_y}{v} \\ \frac{v_z}{v} \end{bmatrix} \quad (6.26)$$

Therefore, the directional permeability becomes,

$$\begin{aligned} K(\xi) &= \begin{bmatrix} n_x & n_y & n_z \end{bmatrix} \begin{bmatrix} K_x & 0 & 0 \\ 0 & K_y & 0 \\ 0 & 0 & K_x \end{bmatrix} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \\ &= \left(\frac{v_x}{v} \right)^2 K_x + \left(\frac{v_y}{v} \right)^2 K_y + \left(\frac{v_z}{v} \right)^2 K_z. \end{aligned} \quad (6.27)$$

6.2.2 Total flow resistance before and after the water breakthrough

In the following paragraphs, the method to determine the total flow resistance before and after the water breakthrough is introduced. The key is to integrate the flow resistance ahead of and behind a water front (shock front) separately.

Typical saturation profiles for this Riemann problem at different times are shown in Figure 6.4. As shown in this figure, at time t_1 and t_2 before water breakthrough, a smooth propagation of rarefaction wave is trailing a shock front of saturation s^* , and the saturation of unswept area in front of a shock keeps at the initial value s_R . The saturation of shock front

s^* is a constant value, while its location and *time-of-flight* (τ^*) varies with time. After the water breakthrough (t_3 in the Figure 6.4), the leading shock front has already arrived the outlet of streamtube; the rarefaction wave keeps on propagating towards the streamtube outlet.

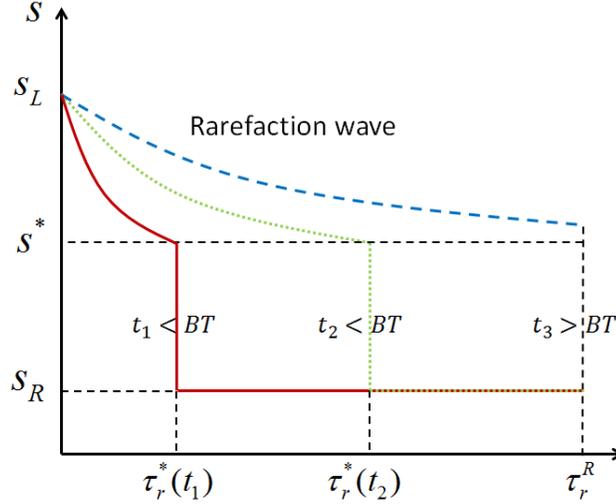


Figure 6.4: Typical saturation profiles for a Riemann problem at different time; where BT is the water break through time

Determination of the total flow resistance before the water breakthrough

According to the discussions made above, before the water breakthrough, the total flow resistance $R(t)$ is the sum of rarefaction wave flow resistance $R_-(t)$ behind a shock, and the flow resistance $R_+(t)$ ahead of a shock,

$$R(t) = R_-(t) + R_+(t), \quad (6.28)$$

or,

$$R(t) = \int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s(\xi, t))} d\xi = \int_0^{\tau^*} \frac{\phi(\xi) v(\xi)^2}{\lambda_-(\xi, s(\xi, t))} d\xi + \int_{\tau^*}^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda_+(\xi, s_R)} d\xi, \quad (6.29)$$

where τ^* is the time-of-flight of shock front location at time t ; $\lambda_-(\xi, s(\xi, t))$ is the total mobility of rarefaction wave; and $\lambda_+(\xi, s_R)$ is the total mobility of unswept area.

Since the reservoir and fluid properties of unswept area are still at the initial conditions, $\lambda_+(\xi, s_R)$ is a known function. The only unknown factor for solving $R(t)$ in Eq. 6.29 is the

total mobility of rarefaction wave λ_- . Based on its definition Eq. 6.24, λ_- is a function of saturation and location. It can be determined numerically once the saturation profile is obtained.

Determinations of the total flow resistance after the water breakthrough

After the water breakthrough, only a rarefaction wave contributes to the total flow resistance $R(t)$,

$$R(t) = R_-(t), \quad (6.30)$$

or,

$$R(t) = \int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda_-(\xi, s(\xi, t))} d\tau, \quad (6.31)$$

where $\lambda_-(\xi, s(\xi, t))$ is the total mobility of the rarefaction wave.

6.2.3 The shock front location, water breakthrough time, and saturation profile

In the following paragraphs, the saturation profile in the rarefaction wave, Eq. 6.15,

$$s(\xi, t) = f'^{-1} \left(\frac{q_r \tau(\xi)}{Q(t)} \right);$$

the *time-of-flight* at shock front, *i.e.*,

$$\tau^*(t) = \frac{f'(s^*) Q(t)}{q_r}; \quad (6.32)$$

and the water breakthrough time, *i.e.*,

$$BT = Q^{-1} \left(\frac{q_r \tau^R}{f'(s^*)} \right); \quad (6.33)$$

are derived. In the above equations, $Q(t)$ is the cumulative water volume injected to the streamtube at time t ; and Q^{-1} is the inverse function of $Q(t)$.

In the rarefaction wave, the propagation velocity of the fluids with a certain saturation s

along a streamline is given by (Buckley and Leverett, 1942),

$$\left. \frac{d\xi}{dt} \right|_s = \frac{q(t)}{\phi(\xi) A(\xi)} f'(s), \quad (6.34)$$

where, $f'(s) = \frac{df}{ds}$ is the derivative of water fractional flow with respect to water saturation.

Integrating equation Eq. 6.34 gives,

$$\int_0^\xi \phi(\xi) A(\xi) d\xi = f'(s) \int_0^t q(t) dt. \quad (6.35)$$

The left hand side of the equation Eq. 6.35 is the pore volume V of the streamtube.

According to Eq. 6.9 and Eq. 6.12,

$$\int_0^\xi \phi(\xi) A(\xi) d\xi = V = q_r \tau. \quad (6.36)$$

Substituting Eq. 6.36 into Eq. 6.35 yields,

$$q_r \tau = f'(s) \int_0^t q(t) dt. \quad (6.37)$$

Denote $Q(t)$ as the cumulative water volume injected to the streamtube at time t ,

$$Q(t) = \int_0^t q(t) dt; \quad (6.38)$$

then, Eq. 6.37 becomes,

$$q_r \tau = f'(s) Q(t). \quad (6.39)$$

Since $f'(s)$ is monotonically decreasing at the range of $s \in [s^*, s_L]$, it has an inverse function f'^{-1} . Thus, according to Eq. 6.39, the saturation profile for rarefaction wave is given by (Eq. 6.15),

$$s(\xi, t) = f'^{-1} \left(\frac{q_r \tau(\xi)}{Q(t)} \right).$$

At the shock front, $s = s^*$, Eq. 6.39 becomes,

$$q_r \tau^*(t) = f'(s^*) Q(t). \quad (6.40)$$

Therefore, the relationship between shock front time-of-flight τ^* and time t is revealed as

(Eq. 6.32),

$$\tau^*(t) = \frac{f'(s^*) Q(t)}{q_r}.$$

At water breakthrough, the shock front arrives the outlet of streamtube, $\tau^* = \tau^R$, and Eq. 6.32 becomes,

$$Q(BT) = \frac{q_r \tau^R}{f'(s^*)}; \quad (6.41)$$

where BT is the water breakthrough time.

Since $Q(t)$ is a monotonically increasing function of time, it has an inverse function Q^{-1} .

Thus, the water breakthrough time is given by (Eq. 6.33),

$$BT = Q^{-1}\left(\frac{q_r \tau^R}{f'(s^*)}\right).$$

6.2.4 The relationship between flow rate and time

In the following paragraphs, the total flow rate $q(t)$ is determined using given cumulative water volume injected dQ at each discretization stage. This is because the water saturation profile is given explicitly and analytically in terms of the cumulative water volume injection $Q(t)$.

The incremental water volume injected dQ can be given for every discretization stage $i = 1, 2, \dots, n, \dots$. Hence, the accumulated water injected volume at the discretization stage $i = n$ can be expressed as,

$$Q_n = \sum_{i=1}^n dQ_i. \quad (6.42)$$

In order to calculate the total flow rate along each streamline, all the parameters in Eq. 6.13, *i.e.*,

$$q(t) = \frac{q_r \Delta P}{\int_0^{\tau^R} \frac{\phi(\xi) v(\xi)^2}{\lambda(\xi, s)} d\tau} = \frac{q_r \Delta P}{R(t)}$$

must be known. The saturation independent parameters are determined using the Cubic

method during the single phase flow period. The saturation dependent parameters can be determined after the saturation profile is obtained. Then, the total flow rate along each streamline $q(t_n)$ at the current discretization stage $i = n$ can be determined using the saturation profile given by Eq. 6.15, *i.e.*,

$$s(\xi, n) = f'^{-1} \left(\frac{q_r \tau(\xi)}{Q_n} \right).$$

The incremental time Δt_n at any discretization stage $i = n$ can be determined by the arithmetic average flow rate of this time interval,

$$\Delta t_n = \frac{2dQ_n}{q(t_{n-1}) + q(t_n)}, \quad (6.43)$$

where $q(t_{n-1})$ is the total flow rate at previous discretization stage (when $n = 1$, $q(t_{n-1}) = q_r$); $q(t_n)$ is the flow rate at current discretization stage.

Finally, the displacement time at discretization stage $i = n$ can be determined,

$$t_n = \sum_{i=1}^n \Delta t_i. \quad (6.44)$$

Based on the above discussions, the saturation distributions along streamline s (Eq. 6.15) and the total flow rate $q(t)$ as a function of time (Eq. 6.13) can be calculated at each discretization stage. The incremental water volume injected at each stage dQ should be given with a reasonable value. One reasonable approach used in this research is,

$$dQ_i = q(t_{i-1})\Delta t_{i-1}; \quad (6.45)$$

where $q(t_{i-1})$ is the total flow rate at previous discretization stage (when $i = 1$, $q(t_{i-1}) = q_r$); Δt_{i-1} is the incremental time at previous discretization stage, when $i = 1$, Δt_{i-1} is the incremental *time-of-flight* in the first grid block.

In summary, an overall flow chart of simulating two phase flow using the semi-analytical Riemann approach along each streamline is shown in Figure 6.5.

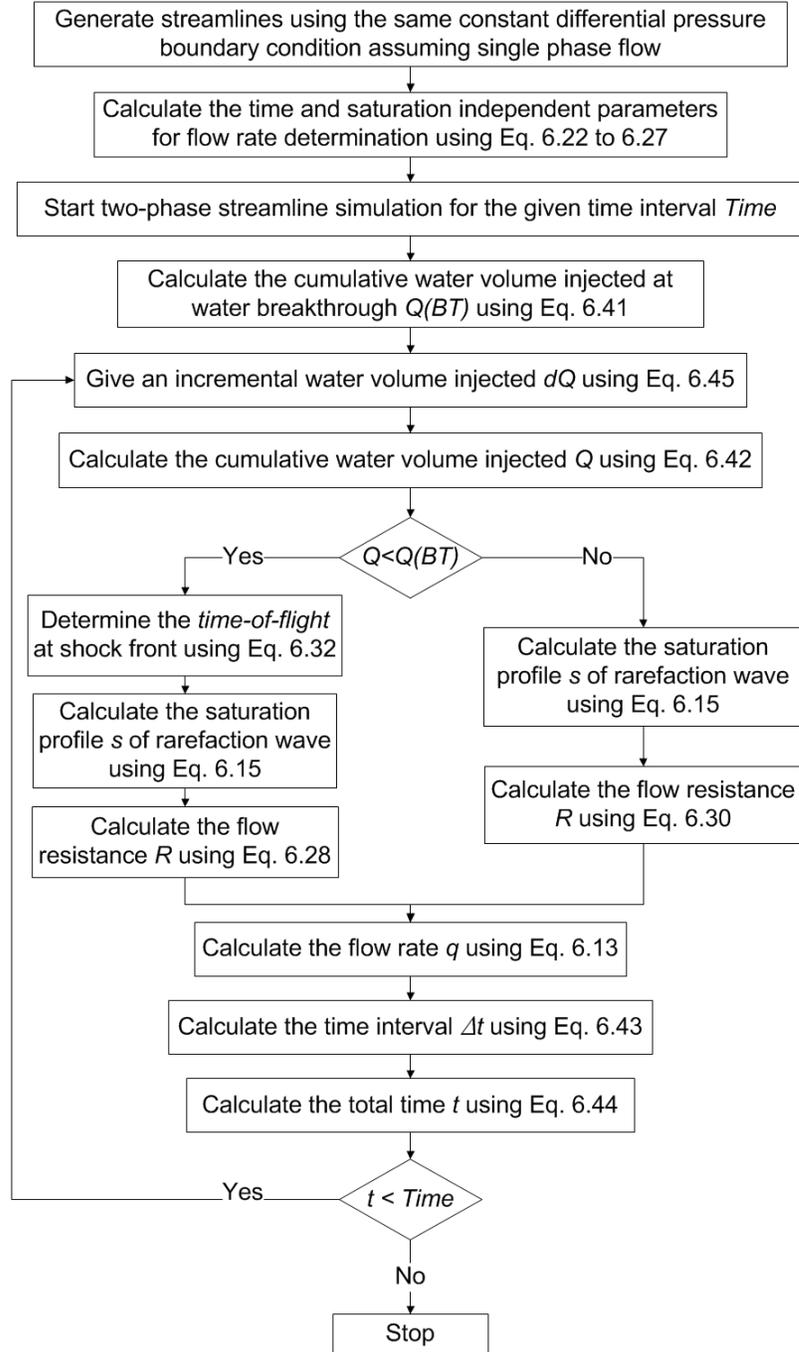


Figure 6.5: Flow chart of semi-analytical Riemann approach along streamline

The phase flow rate at the producer is given by summarizing the results from all streamlines at the producer grid block,

$$q_t = \sum_{l=1}^N q_l; \quad (6.46)$$

$$q_w = \sum_{l=1}^N q_l f(s_l); \quad (6.47)$$

where q_t is total production rate; q_w is water production rate; q_l is the total flow rate carried in a streamline l ; N is the total number of streamlines; f is water fractional flow; s_l is the saturation of a streamline l at its outlet.

6.3 The New Streamline Simulator

The new streamline simulator developed in this research thesis is a combination of the Cubic streamline tracing method and the semi-analytical Riemann solver developed in this section. As mentioned before, there are three steps for this new streamline simulator to simulate a two-phase immiscible displacement process. First, the pressure equations are solved using the first-order finite-difference method with given boundary conditions; Second, streamlines are traced using the Cubic method (introduced in Chapter 4); Finally, the transport problem is solved along streamlines using the semi-analytical Riemann approach (introduced in section 6.2).

In the last two steps, the new streamline simulator assumes streamline paths are fixed over time. The advantage of this assumption is that the pressure field is calculated only once, thus a high computational speed is achieved (Batycky 1997). This assumption may lead to large errors when the pressure distribution and velocity field vary significantly within a short period of time during the displacement process. This situation will occur when boundary conditions are suddenly changed and/or total fluid mobility change significantly over a short distance along streamlines/streamtubes (Batycky 1997). For all tracer tests, the total fluid mobility stays unchanged; for waterflooding displacements, the total fluid mobility changes gradually over a long rarefaction wave. Therefore this assumption is valid for all tracer tests, and some waterflooding processes.

The next section 6.4 presents some case studies with varying reservoir and fluid properties

for waterflooding under constant pressure boundaries. Through these case studies, the accuracy and efficiency of this new streamline simulator and the commercial reservoir simulator software Eclipse100 (Copyright Schlumberger 2009-2015) are compared.

6.4 Calculation Examples and Comparisons

To evaluate the accuracy and efficiency of the new streamline simulator, a series of two-phase immiscible flow calculations in quarter-five-spot patterns are performed. The commercial petroleum reservoir simulator Eclipse100 (Copyright 2009-2015 Schlumberger) is also applied to model the same processes. Through demonstrations and comparisons, the advantages and limitations of the new streamline simulator is concluded.

In this section, the calculation examples are designed to evaluate the new streamline simulator from the following four aspects: the ability to model homogeneous, heterogeneous, anisotropic and different end point mobility ratios (ranging from 0.5 to 50). Note that all the cases simulated in this section using both the new streamline simulator and Eclipse100 assume the effects of capillary pressure and gravity are negligible due to the limitation of the scope of this research thesis.

6.4.1 Homogeneous problems

In this section, the simulation results of two- and three- dimensional waterflooding problems in homogeneous porous medium using the streamline simulator are presented for three purposes:

1. To gain a better understanding of the concept of solving the transport problems along one-dimensional streamlines, the flow rate associated with streamlines, the total flow resistance for streamlines;

2. To demonstrate the engineering application of the new streamline simulator, the oil and water production rate, and its sensitivity to the number of streamlines;
3. To evaluate the accuracy and efficiency of the new streamline simulator, comparing with Eclipse100 in modeling the same waterflooding problem.

Table 6.1: Reservoir properties and well location for the two-dimensional homogeneous problem

Reservoir dimension		40 cm × 40 cm
Absolute permeability	K	100.0 mD
Porosity	ϕ	30%
Injector location		(2 cm, 2 cm)
Producer location		(39 cm, 39 cm)
Constant differential pressure	ΔP	9.5 atm
Number of streamlines		50
Simulation period	$Time$	2.0 hr
Water viscosity	μ_w	1.0 cP
Oil viscosity	μ_o	5.0 cP
Connate water saturation	S_{wc}	0.2
Residual oil saturation	S_{or}	0.3
Maximum water relative permeability	a_w	0.4
Maximum oil relative permeability	a_o	1.0
Exponent in Corey model for water	n_w	2.0
Exponent in Corey model for oil	n_o	2.0
End point mobility ratio	M	2.0

In this two-dimensional homogeneous model, the input parameters are summarized in Table 6.1. The fluid relative permeability relations in this model are given by the Corey's model (1954),

$$k_{rw}(S_w) = a_w \left(\frac{S_w - S_{wc}}{1 - S_{wc} - S_{or}} \right)^{n_w}; \quad (6.48)$$

$$k_{ro}(S_w) = a_o \left(\frac{1 - S_w - S_{or}}{1 - S_{wc} - S_{or}} \right)^{n_o}; \quad (6.49)$$

where, S_{wc} is connate water saturation; S_{or} is residual oil saturation; a_w and a_o , is the maximum relative permeability for water and oil, respectively; n_w and n_o is the Corey exponent for water and oil, respectively.

The end point mobility ratio M is given by,

$$M = \frac{\frac{a_w}{\mu_w}}{\frac{a_o}{\mu_o}}, \quad (6.50)$$

where μ_w and μ_o is the viscosity for water and oil, respectively.

With given parameters, streamlines are generated by using the Cubic method. The results are shown in Figure 6.6.

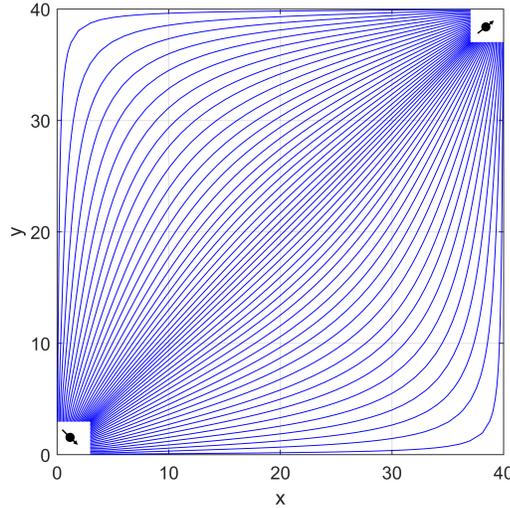


Figure 6.6: 40 streamlines in 2D homogeneous problem

Then, the transport problem is solved along a series of streamlines by using the semi-analytically Riemann approach (section 6.2). As shown in Figure 6.7, the flow rate associated with each streamline varies with time, since wells operate under a constant delta pressure. We can observe that the changes of flow rate is neither monotonic nor smooth. This behavior can be understood if one looks at the changing in the total flow resistance (R). For the streamline number 1 in Figure 6.7, the total flow resistance is given in Figure 6.8.

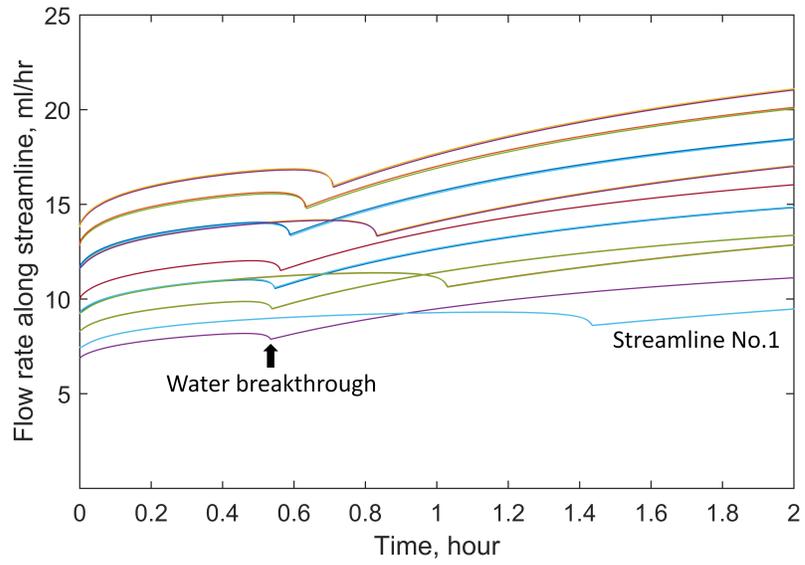


Figure 6.7: The flow rate associated with each streamline varies with time

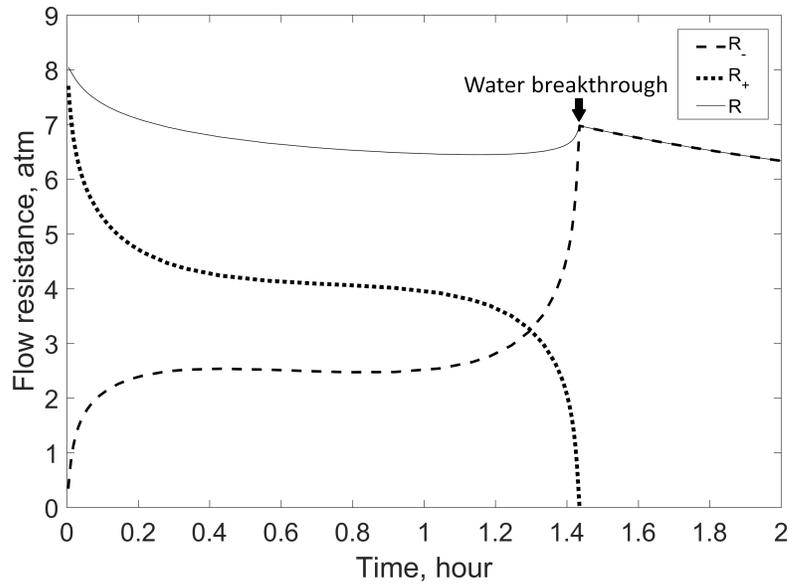


Figure 6.8: The flow resistance for the streamline No.1 in the Figure 6.7

Recall that the physical meaning of total flow resistance is the differential pressure required for keeping the flow rate at its reference value. Before water breakthrough, the total flow resistance is given by,

$$R = R_- + R_+; \tag{6.51}$$

where, R is total flow resistance; R_- is the rarefaction wave flow resistance behind shock; R_+ is the flow resistance ahead of shock. As shown in Figure 6.8, as the shock front approaches to the producer, the flow resistance ahead of shock R_+ is decreasing towards zero, since the area of unswept area is shrinking; while the rarefaction wave flow resistance R_- is gradually changing. The changes of R_- is a combined effects of the velocity field, the total fluid mobility, and the interval of integration from the inlet to the shock front along the streamline ($R_- = \int_0^{\tau^*} \frac{\phi(\xi)v(\xi)^2}{\lambda_-(\xi,s(\xi,t))} d\tau$). Therefore, the changes of rarefaction wave flow resistance R_- is not necessarily monotonic.

We can also observe in Figure 6.8 that, when the shock front travels close to the injector/producer (around 0.0 *hr* and 1.4 *hr*), the changes of both flow resistances (R_+ and R_-) are rapid. In these regions, the velocity is significantly higher than the other regions that are away from the injector/producer, thus, the increases or decreases of the integration interval becomes important, and it dominates the variation in flow resistance.

When the water reaches the outlet along the streamline, R_+ becomes zero, the total flow resistance of this streamline becomes, $R = R_-$. As can be observed in Figure 6.8, the changing trend of the rarefaction wave flow resistance R_- swifts from increasing to decreasing at the water breakthrough point. This is because the water is more mobile than oil, and the R_- is inversely proportional to the total fluid mobility (Eq. 6.31).

Generally speaking, due to the complex changing behavior of total flow resistance R , and the saturation discontinuous across the shock front, the changes of flow rate is continuous, but not necessarily monotonic nor smooth.

The oil and water flow rate at the producer can be obtained by integrating the results from all streamlines at the producer grid block. The simulation results are dependent on the number of streamlines. The simulation results obtained from 10, 50 and 150 streamlines are shown in Figure 6.9. Before water breakthrough, the results obtained from different number of streamlines are identical. After water breakthrough the difference can be observed. Because

streamlines are treated independently from each other and the water breakthrough time for each streamline are different, the production profiles generated by streamline simulator are not smooth. More smooth production profile can be obtained with more streamlines launched at the injector. Comparing these results we can observe that 10 streamlines are not sufficient to provide accurate results for this case; while the difference between 50 and 150 streamlines is very small. This comparison indicates that launching too many streamlines is unnecessary to obtain sufficiently accurate results; while, too few streamlines create erroneous results.

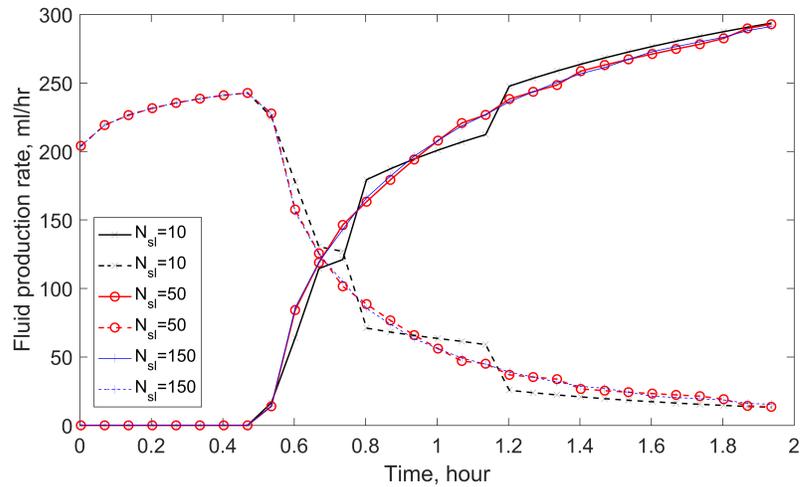


Figure 6.9: The sensitivity of the fluid production rates to the number of streamlines

The new streamline simulator can be validated by comparing the production profiles from 50 streamlines and Eclipse100. The difference between two results are quantified by calculating the *Relative Difference (RD)*,

$$RD = \frac{1}{N} \sum_{i=1}^N \frac{|q_{i,e} - q_{i,s}|}{\frac{q_{i,e} + q_{i,s}}{2}}, \quad (6.52)$$

where, $q_{i,e}$ and $q_{i,s}$ is the phase flow rate simulated by using the Eclipse100 and the new streamline simulator at the estimation point i , respectively; and N is the total number of the estimation points. The *Relative Difference* can be evaluated for both oil (RD_o) and water (RD_w) production rates. As shown in Figure 6.10, the agreement in production profile between streamline simulator and Eclipse100 is acceptable, while differ from each other.

The streamline method delivers more accurate results, as will be discussed later in the text.

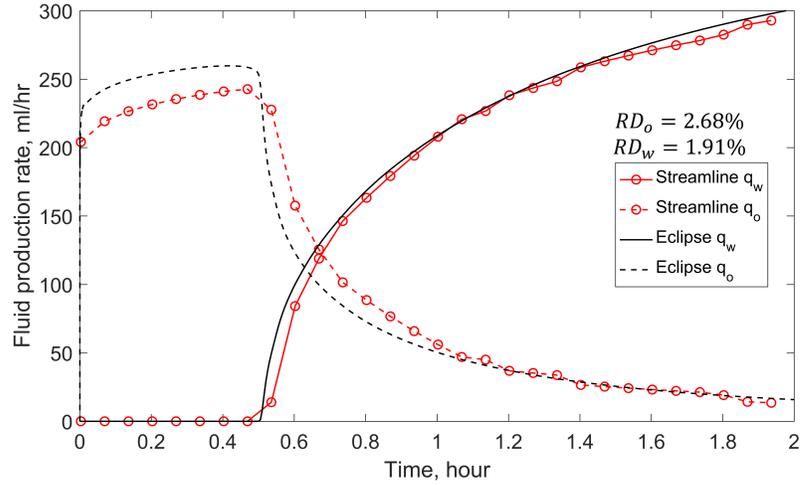


Figure 6.10: The production profile for 2D homogeneous problem

Table 6.2: Reservoir properties and well locations for the three-dimensional homogeneous problem

Reservoir dimension		40 cm × 40 cm × 20 cm
Absolute permeability	K	100.0 mD
Porosity	ϕ	30%
Injector location		(2 cm, 2 cm, 1 cm)
Producer location		(39 cm, 39 cm, 20 cm)
Constant differential pressure	ΔP	21.7 atm
Number of streamlines		75
Simulation period	$Time$	2.0 hr
Water viscosity	μ_w	1.0 cP
Oil viscosity	μ_o	5.0 cP
Connate water saturation	S_{wc}	0.2
Residual oil saturation	S_{or}	0.3
Maximum water relative permeability	a_w	0.4
Maximum oil relative permeability	a_o	1.0
Exponent in Corey model for water	n_w	2.0
Exponent in Corey model for oil	n_o	2.0
End point mobility ratio	M	2.0

In the three-dimensional quarter-five-spot case, the input parameters are given in Table 6.2. The injector and producer is located in the bottom and top layer, respectively. The streamline tracing results are given in Figure 6.11. The oil and water production profile given

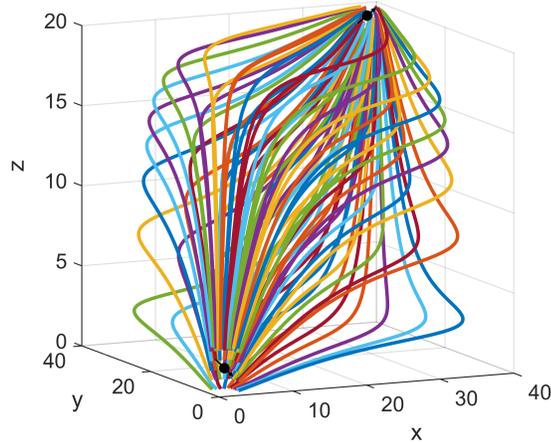


Figure 6.11: Streamlines in 3D homogeneous problem

by the new streamline simulator and Eclipse100 are shown in Figure 6.12. In 3D Eclipse simulations, the gravity effects are switched off by using the fluids with same density. The fluid production profiles for this three-dimensional case are also in acceptable agreement, while streamline method is more accurate, as will be explained later in the text.

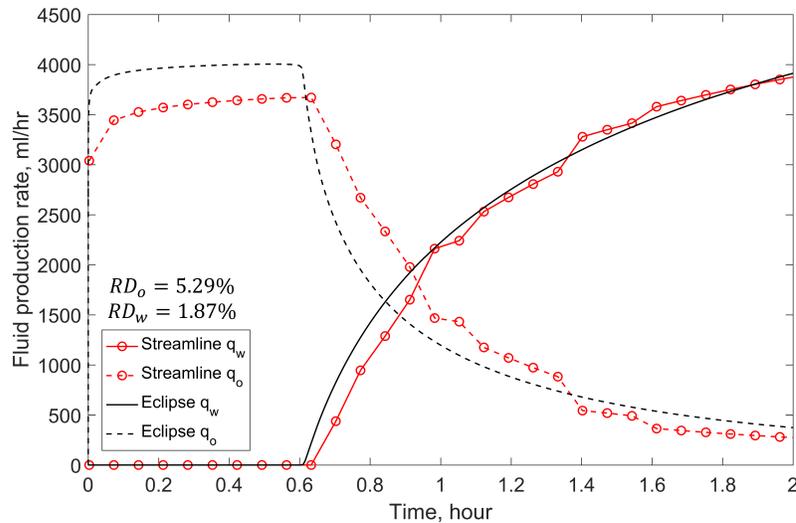


Figure 6.12: The fluids production rates in 3D homogeneous problem

Computational time comparison

Importantly, the new streamline simulator is more computationally efficient than Eclipse100. As shown in Figure 6.13, the CPU usage for the new streamline simulator and Eclipse100 are compared for the above two- and three- dimensional waterflooding simulations. As can be observed, even with a less powerful solver, the new streamline simulator is two times faster than Eclipse100. This advantage is more significant as the grid block number increases.

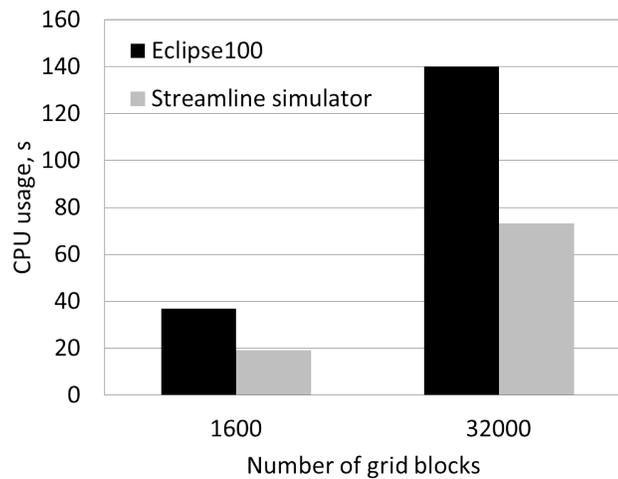


Figure 6.13: The CPU usages for the 2D and 3D waterflooding simulations

Comparing with a streamline simulator, there are three types of numerical error within Eclipse100 in simulating waterflooding, as discussed below.

Grid orientation effect

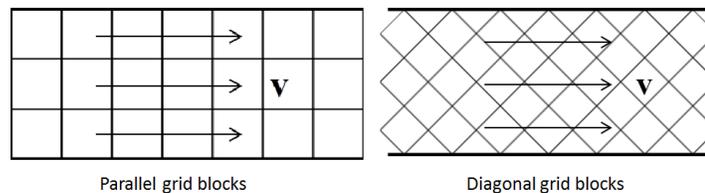


Figure 6.14: An illustration of parallel and diagonal grid blocks

The grid orientation effect is a phenomenon which leads to different solutions with different grid orientation (parallel or diagonal to the principal flow direction, as illustrated in Figure

6.14) and size. These differences do not vanish but can be reduced when finer grids are used.

Brand *et al.* (1991) compared saturation contours in a five-spot displacement, calculated with parallel and diagonal square grid blocks of different sizes at uniform mobility ratio ($M = 1$) in Figure 6.15. As can be observed, the diagonal grid blocks deliver late water breakthrough time at the producer. This situation will become worse when the mobility ratio is greater than one. On the other hand, the parallel grid blocks deliver more accurate results, as illustrated by similar results using finer grid blocks.

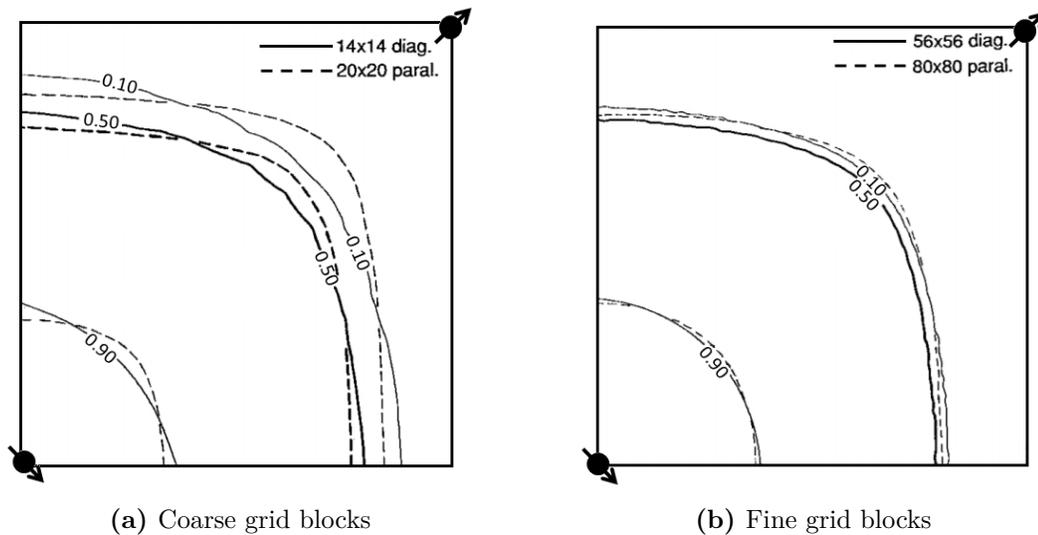


Figure 6.15: Comparison of saturation contours for parallel and diagonal grid block in a five-spot displacement, $M = 1$, (Brand *et al.*, 1991)

In these particular waterflooding cases, Eclipse100 solutions is influenced by the orientation effects of the underlying finite difference grid blocks. These grid blocks are diagonal to the line connecting injector and producer, and deliver late water breakthrough when the mobility ratio $M = 2$.

To the contrary, streamlines are accurately applied as curvilinear grid blocks parallel to the flow direction, thus, the grid orientation effects are significantly reduced.

Numerical dispersion

Eclipse100 obtain solutions to fluid flow problems by replacing derivatives with finite-difference approximations. The use of these approximations, derived by manipulating Taylor's series, introduces an error known as truncation error. For many problems, such as single phase flow problems, the error is small and the numerical solutions are sufficiently accurate. However, these truncation errors can cause significant solution inaccuracies for immiscible floods in which viscous forces are much larger than capillary forces (Fanchi, 1983). As shown in Figure 6.16, in Riemann problems, the finite-difference solution of Buckley-Leverett equation (Eq. 6.1) introduces truncation error that can smear the shock front as if additional physical dispersion were present. This smearing, is called numerical dispersion or numerical diffusion. The effects of numerical dispersion can be partly overcome by using finer grid blocks and smaller time-steps. Obviously, the trade-off is the increment in computational time.

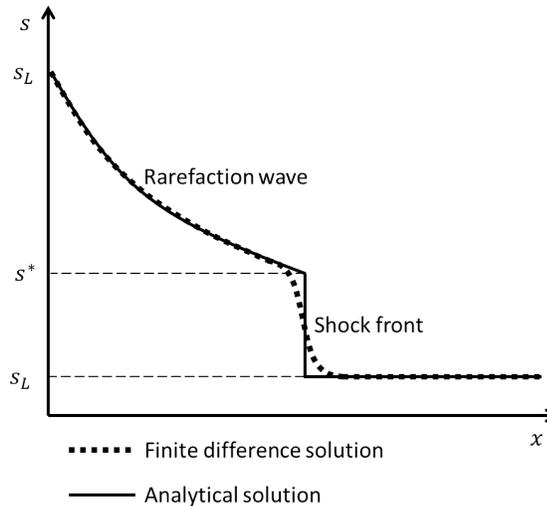


Figure 6.16: An illustration of the numerical dispersion in finite-difference approximations

In the new streamline simulator, instead of using a finite-difference approach, the analytical Riemann solution is directly mapped along streamlines in terms of time-of-flight. Thus, the numerical dispersion in numerical solutions of Buckley-Leverett equation is eliminated in the streamline simulator.

Well treatment

In Eclipse100, boundary conditions are defined by well controls, such as bottom hole pressure or flow rate targets. The flow between the well-bore and a single reservoir grid block is related by a *connection transmissibility factor* (Eclipse Technical Description, Schlumberger, Version 2014.1), which depends on the geometry of the connecting grid block, the well-bore radius, and the rock permeability. This well treatment is practical in many engineering applications, since we can only directly control wells but not grid blocks in oil fields, and it accounts for the effects of different size and completion methods of wells. However, when the boundary condition is defined in grid block level for research purposes, Eclipse100 cannot specific this boundary condition at grid block level but to specific the bottom hole pressure at wells.

In these particular waterflooding processes, the wells are not specifically defined, and the boundary condition is given by constant grid block pressures. Thus, Eclipse100 induces some errors in well treatments, which is illustrated by resulting different flow rate before water breakthrough in Figure 6.10. In the new streamline simulator, the boundary condition is defined by directly feeding the given grid block pressures. Therefore, the streamline simulator is more accurate than Eclipse100 in keeping the constant differential pressure for this particular case.

Summary of error discussions

Based on the above discussions, the streamline simulator gives more accurate results compared to Eclipse100 for this particular case, since the streamline method applied can significantly reduce the grid orientation and numerical dispersion in a finite difference approach (Eclipse100). In addition, the well treatment method applied in Eclipse100 will induce some errors when the wells are not specifically defined. The similar conclusions are also reported in Batycky (1997) when comparing streamline simulation results with Eclipse100 for waterflooding problems in quarter-five-spot reservoir.

6.4.2 Heterogeneous problems

In this section, the ability of the new streamline simulator to model heterogeneous problems is tested in both two- and three-dimensional quarter-five-spot cases.

For the two-dimensional case, the permeability distribution is depicted in Figure 6.17 and 6.18. The permeability is normally distributed ranging from $36.23mD$ to $249.24mD$. The constant differential pressure is $12.6atm$. The same parameters given in Table 6.1 are applied.

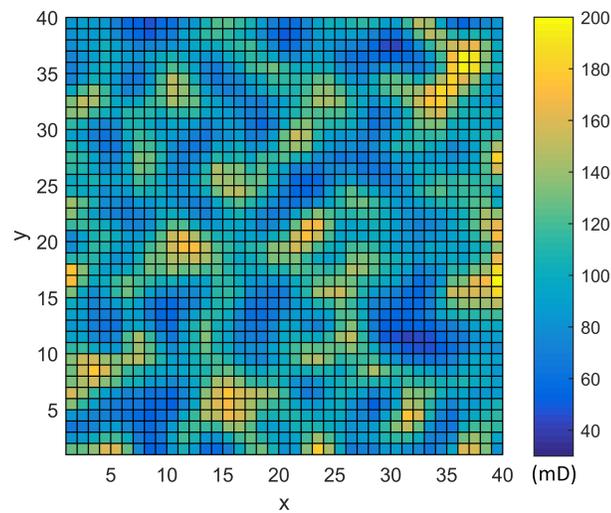


Figure 6.17: The permeability distribution in the 2D heterogeneous problem

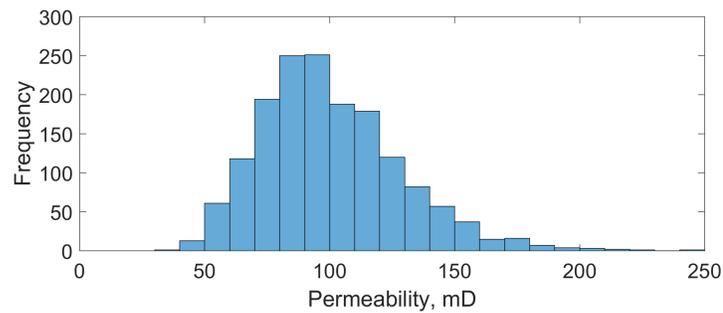


Figure 6.18: The histogram of the permeability distribution in 2D heterogeneous problem

With the given information, the streamlines generated for this case are shown in Figure 6.19. As can be observed, the streamlines become closer where the permeability is relatively

large, reflecting the fact that the velocity is higher in these regions.

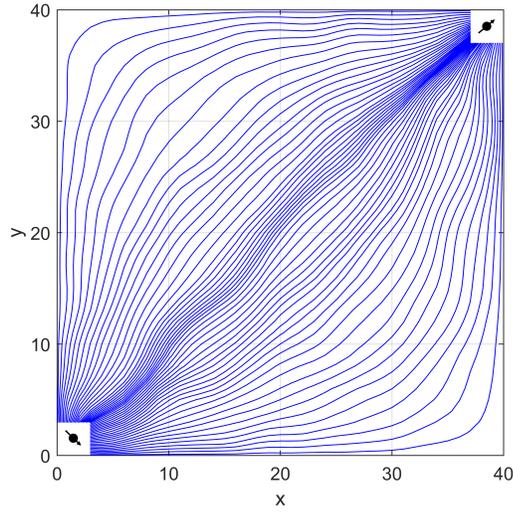


Figure 6.19: Streamlines in 2D heterogeneous problem

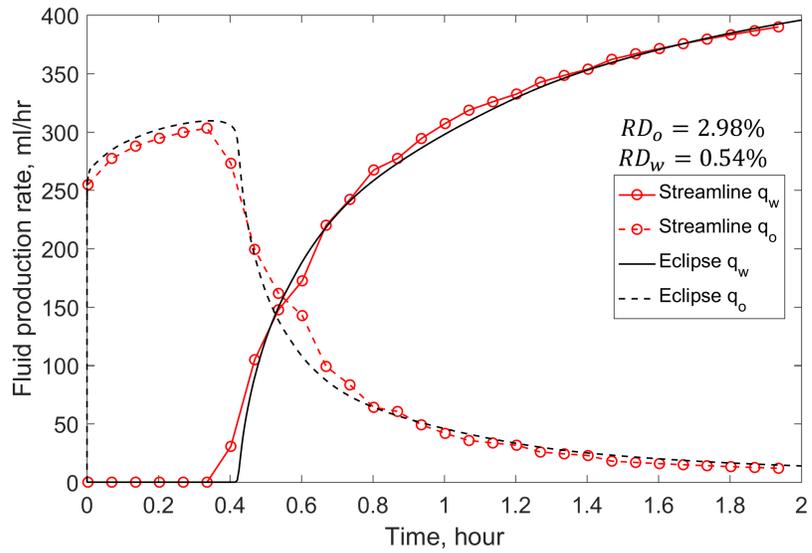


Figure 6.20: The fluids production rates in 2D heterogeneous problem

The production profiles generated by the new streamline simulator and Eclipse100 for this case are shown in Figure 6.20. The effects of permeability heterogeneity on the production profile are clearly reflected in streamline results, while they can hardly be observed in Eclipse100's results. These effects are smeared by the numerical diffusion in Eclipse100. Ex-

cept for this difference, the agreement on the production profile between the two simulators is excellent.

In the three-dimensional case, the permeability distribution is depicted in Figure 6.21, and 6.22. The permeability is normally distributed ranging from 53.87 mD to 187.22 mD . The constant differential pressure is 20.7 atm . The same parameters given in Table 6.2 are also applied here.

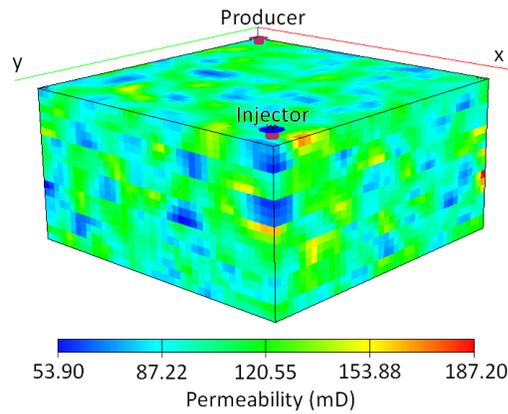


Figure 6.21: The permeability distribution in 3D heterogeneous problem

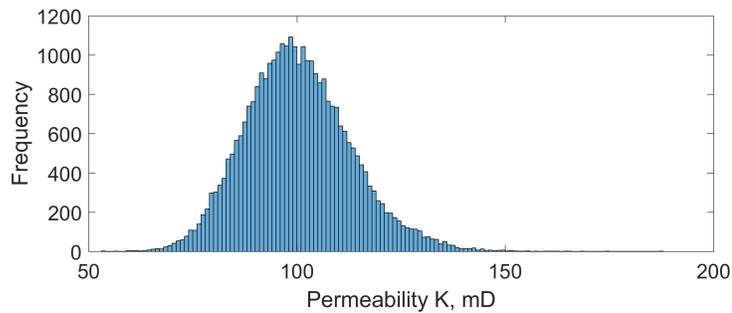


Figure 6.22: The histogram of the permeability distribution in 3D heterogeneous problem

The three-dimensional streamlines for this case are shown in Figure 6.23. The streamlines are not as smooth as in the homogeneous case (Figure 6.11) due to the heterogeneity effects.

With the given information above, the production profiles determined by the new streamline simulator and Eclipse100 are given in Figure 6.24. The same observations found in the two-dimensional case are also shown here. The effects of permeability heterogeneity on the

production profile are obvious on streamline results, but they are smeared by the numerical diffusion in Eclipse100. Overall, the agreement on the production profile between the two simulators is good.

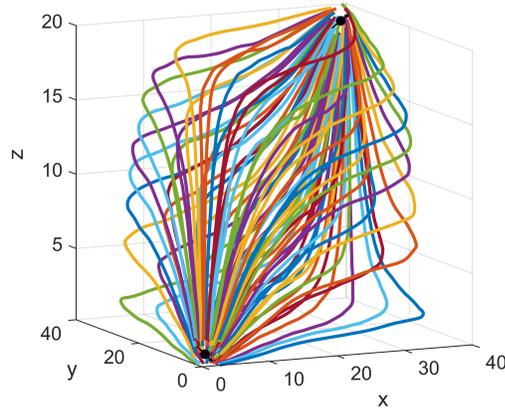


Figure 6.23: Streamlines in 3D heterogeneous problem

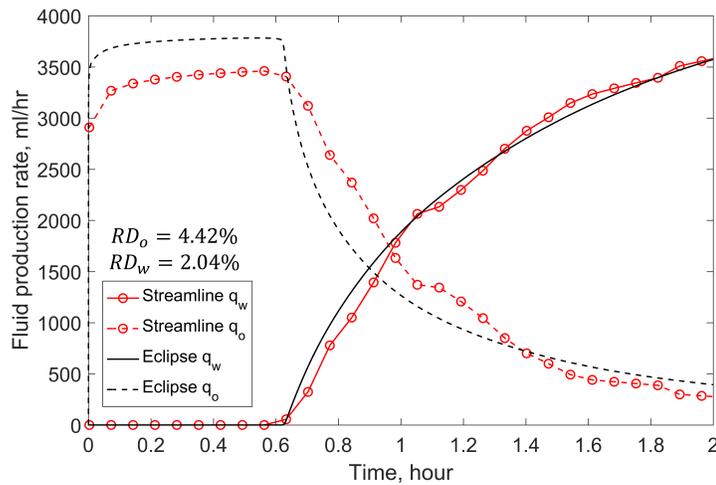


Figure 6.24: The fluids production rates in 3D heterogeneous problem

In summary, the new streamline simulator can give sufficiently accurate results for both two- and three-dimensional heterogeneous cases, it is shown by the good agreement with Eclipse100 in production profiles. The heterogeneous effects can be better simulated by the new streamline simulator than Eclipse100, since they can hardly be observed in Eclipse100's results.

6.4.3 Anisotropic problems

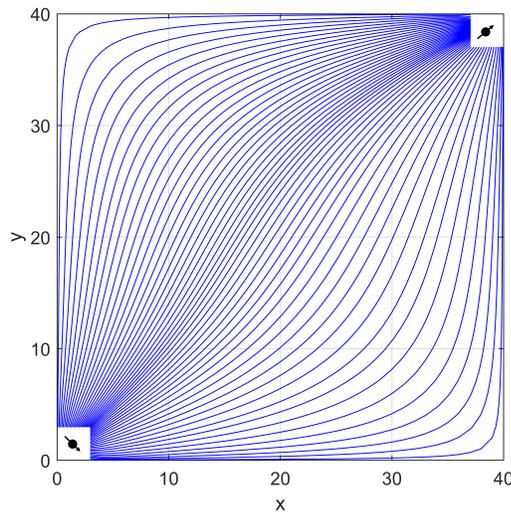


Figure 6.25: Streamlines in 2D anisotropic problem

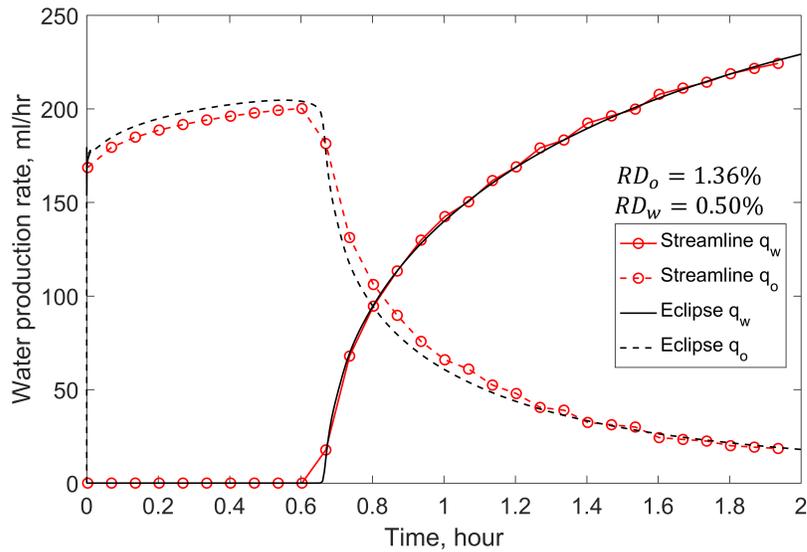


Figure 6.26: The fluids production rates in 2D anisotropic problem

In this section, the ability of the new streamline simulator to model anisotropic problems is tested in both two- and three-dimensional quarter-five-spot cases.

In the two-dimensional case, $k_x = 100 \text{ mD}$, and $k_y = 50 \text{ mD}$. The constant differential pressure is 11.4 atm . The same parameters given in Table 6.1 are applied. With the given

information above, the streamlines generated for this two-dimensional case are shown in Figure 6.25.

The fluid production profiles determined by the new streamline simulator and Eclipse100 for this two-dimensional anisotropic case are shown in Figure 6.26. As can be observed, the agreement on the production profile between the two simulators is excellent.

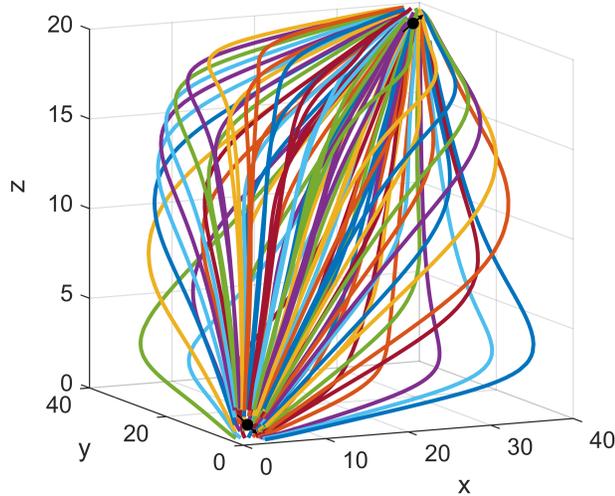


Figure 6.27: Streamlines in 3D anisotropic problem

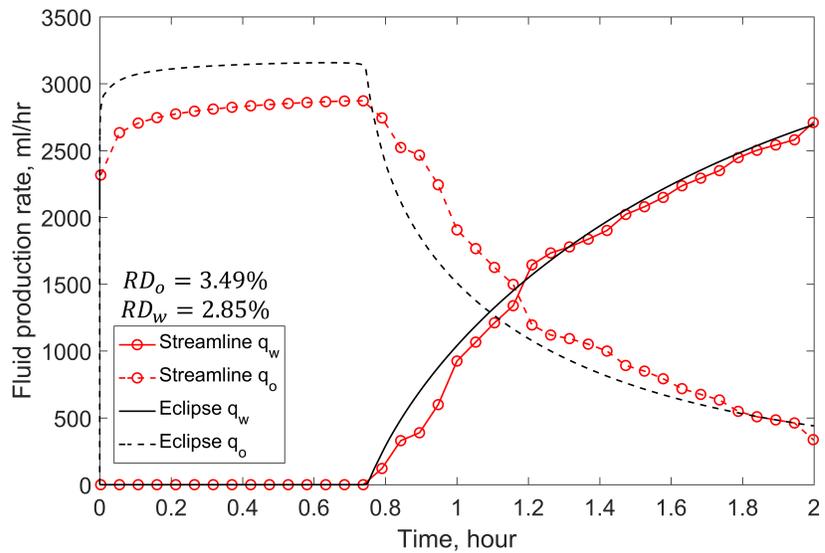


Figure 6.28: The fluids production rates in 3D anisotropic problem

In the three-dimensional case, $k_x = 100 \text{ mD}$, $k_y = 60 \text{ mD}$, $k_z = 30 \text{ mD}$. The constant differential pressure is 32.6 atm . The parameters in Table 6.2 are still applied. With the given information above, the streamlines generated for this case are shown in Figure 6.27.

The fluid production profiles determined by the new streamline simulator and Eclipse100 for this three-dimensional anisotropic case are given in Figure 6.28. The agreement on the production profile between the two simulators is good.

Based on the above discussions and comparisons, the new streamline simulator can give sufficiently accurate results for both two- and three-dimensional anisotropic cases.

6.4.4 Varying mobility ratio problems

As mentioned in the section 6.2, the assumption of fixed streamlines may lead to large errors when the pressure distribution and/or velocity field vary significantly within a short period of time during a displacement process. When the total fluid mobility varies significantly in a short distance along streamlines/streamtubes, the velocity field distribution will become less smooth. This situation may occur when the end point mobility ratio is too large or too small.

In this section, four waterflooding cases with different mobility ratios and relative permeability curves are modeled. The objective is to evaluate the accuracy of the new streamline simulator in the testing end point mobility ratio (M in Table 6.3) range from 0.5 to 50. The same reservoir parameters given in Table 6.1) are applied. The fluid properties, relative permeability parameters, mobility ratios, and constant differential pressures for these four cases are summarized in Table 6.3. In this table, the parameters in Corey relative permeability model (1954) (Eq. 6.48 and 6.49), *i.e.*,

$$k_{rw}(s_w) = a_w \left(\frac{s_w - S_{wc}}{1 - S_{wc} - S_{or}} \right)^{n_w} ;$$

$$k_{ro}(s_w) = a_o \left(\frac{1 - s_w - S_{or}}{1 - S_{wc} - S_{or}} \right)^{n_o} ;$$

are defined to represent the relative permeability relations.

Table 6.3: A summary of fluid properties and boundary conditions

Case number	μ_w (cP)	μ_o (cP)	S_{wc}	S_{or}	a_w	n_w	a_o	n_o	M	ΔP (atm)
No. 1	1.0	5.0	0.2	0.3	0.1	2.0	1.0	2.0	0.5	18.5
No. 2	1.0	5.0	0.2	0.3	0.4	2.0	1.0	2.0	2.0	11.2
No. 3	1.0	10.0	0.2	0.3	1.0	2.0	1.0	2.0	10.0	6.7
No. 4	1.0	100.0	0.2	0.3	0.5	2.0	1.0	2.0	50.0	25.3

Based on the fluid properties for oil and water, the fractional flow function, *i.e.*,

$$f = \frac{\lambda_w}{\lambda} = \frac{\frac{k_{rw}(s_w)}{\mu_w}}{\frac{k_{rw}(s_w)}{\mu_w} + \frac{k_{ro}(s_w)}{\mu_o}} \quad (6.53)$$

can be plotted as a function of water saturation. The fractional flow function plots for these four cases are shown in Figure 6.29. The s^* given in this figure is the shock front water saturation for each case. The determination algorithm for s^* is given in Appendix B.

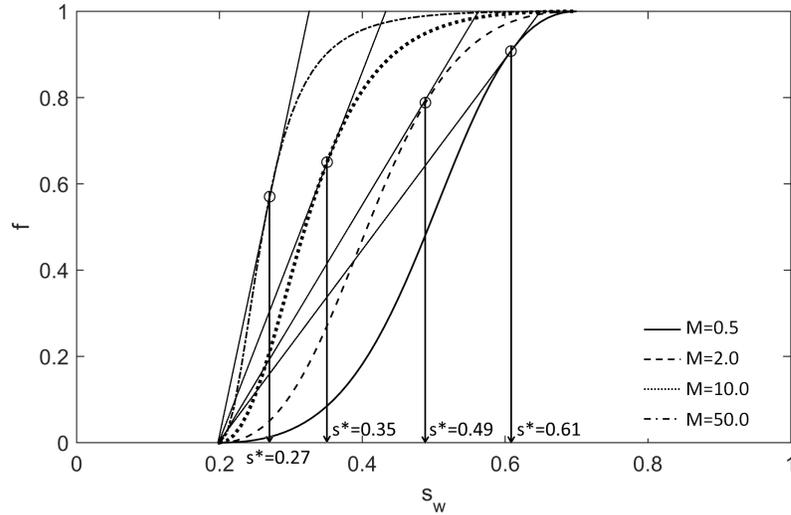


Figure 6.29: The fractional flow curves, and the shock front water saturations for the four cases

The oil and water production profiles simulated for different end point mobility ratios are given in Figures 6.30 to 6.33.

As shown in Figure 6.30, the difference between the new streamline simulator and Eclipse100

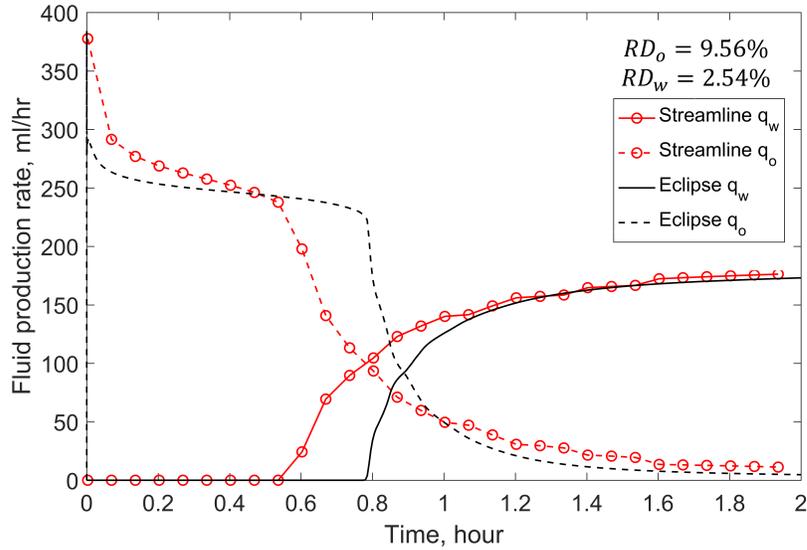


Figure 6.30: The production profiles results from the new streamline simulator and Eclipse100 for the first case ($M = 0.5$)

is large for this favorable mobility case ($M = 0.5$). As Martin *et al.* (1973) argued, the fundamental assumption of fixed streamtube/streamline is not valid when the mobility ratio is less than one, and the streamtube/streamline approach underestimates the volume of produced oil for this case. This is because the effective permeability of water is lower than oil, the most of pressure drop occurs in the rarefaction wave. Moreover, the water saturation of shock front is $s^* = 0.61$, which is much higher than the water saturation ahead of the shock ($s_R = S_{wc} = 0.2$), thus, the total mobility and the velocity field vary significantly across the shock. Therefore, the streamlines in rarefaction waves are almost independent of the unswept area, the assumption of fixed streamlines is not valid. Martin *et al.* (1973) suggested that a better solution can be obtained by updating streamlines several times as the displacement progresses.

Figure 6.31 indicates that the agreement between the new streamline simulator and Eclipse100 is excellent for $M = 2.0$. The saturation difference between shock front ($s^* = 0.49$) and unswept area ($s_R = S_{wc} = 0.2$) is relatively small. The changing of the distributions in pressure and velocity field take a long time in this case. Therefore, the fixed-streamline

assumption is valid for a large time-step. The new streamline simulator achieves computational high efficiency and accuracy simultaneously.

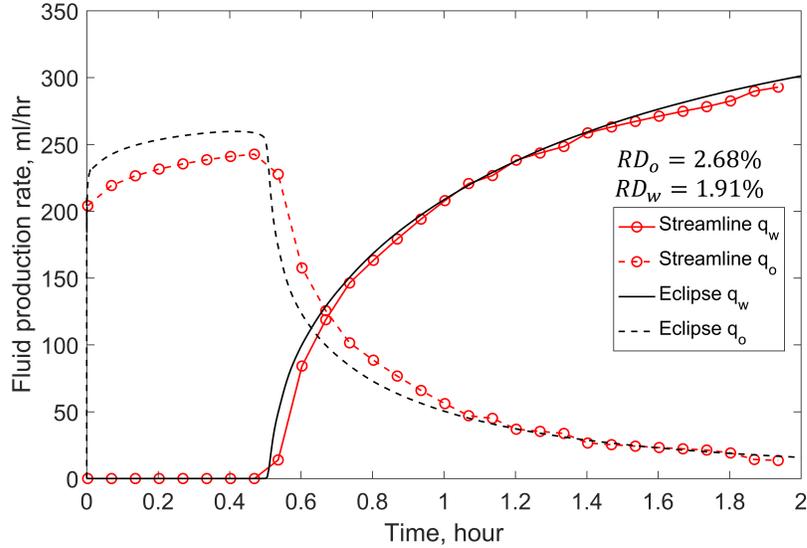


Figure 6.31: The production profiles results from the new streamline simulator and Eclipse100 for the second case ($M = 2$)

Figure 6.32 shows that the simulation results are in good agreement between the new streamline simulator and Eclipse100 for $M = 10$. Even though the end point mobility ratio is relatively large ($M = 10$), the difference in water saturation between shock front ($s^* = 0.35$) and unswept area ($s_R = S_{wc} = 0.2$) is small. The changing of the distributions in pressure and velocity field still takes a relatively long time in this case. Therefore, the fixed-streamline assumption is valid.

Figure 6.33 indicates that the simulation results show fair agreement between the new streamline simulator and Eclipse100 for $M = 50$. Although the end point mobility ratio ($M = 50$) is large, the saturation difference between leading shock ($s^* = 0.27$) and unswept area ($s_R = S_{wc} = 0.2$) is small. The changing of distributions in pressure and velocity field is faster than the situation in third case, where $M = 10$. Updating streamlines periodically can give more accurate results.

On the other hand, the grid orientation effects in Eclipse100 become more significant when

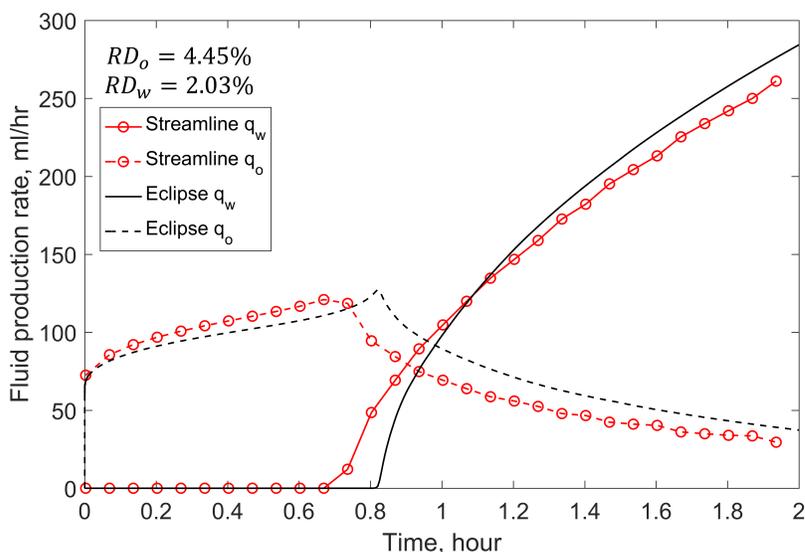


Figure 6.32: The production profiles results from the new streamline simulator and Eclipse100 for the third case ($M = 10$)

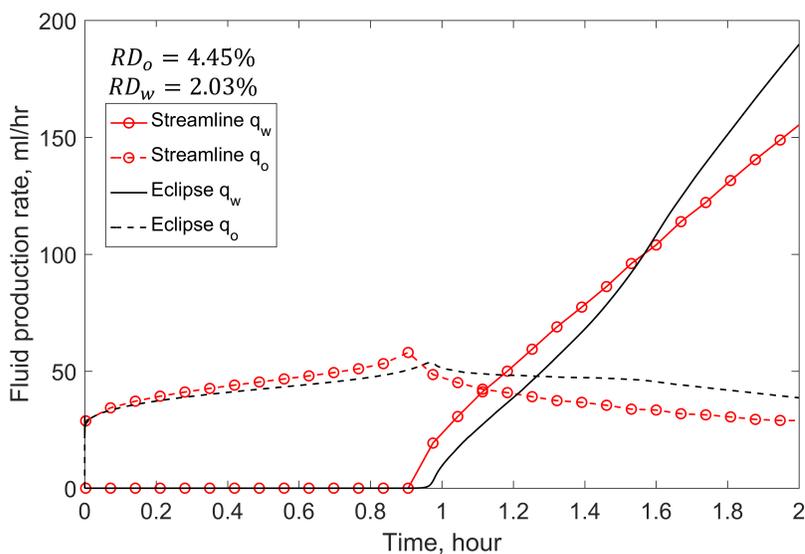


Figure 6.33: The production profiles results from the new streamline simulator and Eclipse100 for the last case ($M = 50$)

the total mobility ratio increases. This situation gradually occurs when the less viscous fluid is injected to reservoir. Ideally, to reduce the gradually increasing grid orientation effects, grid blocks in Eclipse100 should be refined. Comparing to refine grid blocks, updating streamlines are much computational effective. In brief, by mapping the saturation back to

finite difference grid blocks, recalculating the pressure in grid blocks using a finite-difference method, and then streamlines can be re-generated and updated.

In the four cases discussed above, the new streamline simulator can successfully simulate unfavourable waterflooding processes ($M > 1.0$). The advantage of the fixed streamlines assumption is the pressure equation is solved only once, thus the computational speed is fast. The trade-off of this assumption is that it may lead to large errors when the end point mobility ratio is too large ($M > 100$), or too small ($M < 1.0$) when the mobility field and pressure distribution changed significantly during the displacement. This drawback can be overcome by periodically updating streamlines. Due to the limitation of the research scope in this thesis, periodically updating streamlines was not considered.

6.5 Chapter Summary

In summary, this chapter introduced a semi-analytical Riemann solver, which maps the analytical solutions along streamlines. The applications of this semi-analytical Riemann approach are extended from constant flow rate to constant pressure boundaries. When certain conditions apply, this approach saves considerable time in simulation and gives more accurate results than the purely numerical methods.

The new streamline simulator is then developed by combining the Cubic method (Chapter 4) with the semi-analytical Riemann solver. The ability of this new streamline simulator is demonstrated through modeling different waterflooding displacement processes. It has been demonstrated that the new streamline simulator can give sufficiently accurate results in modeling two- and three-dimensional homogeneous, heterogeneous, and anisotropic waterflooding problems. It gives more accurate simulation results than Eclipse100 when the number of streamlines is sufficiently large and the fixed streamline assumption is valid. This is because the grid orientation effects, the numerical dispersion for solving Riemann problems, and the well treatment errors in Eclipse100 have been significantly reduced or

eliminated in the new streamline simulator. Moreover, the calculation speed of the new streamline simulator is much faster than Eclipse100. This advantage becomes more significant when more grid blocks are used in a reservoir model.

The limitations of this new streamline simulator are that it relies on the fixed streamline assumption; and it does not account for gravity and capillary effects. These limitations could be partly eliminated in the future by periodically updating streamlines with additional CPU time costs; and using the concept of operator splitting to account for gravity and capillary effect.

This new streamline simulator has a potential to simulate real waterflooding problems in real fields efficiently, due to its high computational speed and accuracy of the new streamline simulator. In the next chapter, examples of using the new streamline simulator to history match, simulate and study the lab-scale waterflooding displacement processes are presented to validate and demonstrate its ability of modeling of real physical problems.

Chapter 7

Experimental Studies and Simulations for Waterflooding under Constant Differential Pressure Boundaries

7.1 Introduction

Waterflooding is the most commonly used secondary oil recovery method to increase the production from oil reservoirs. In order to maintain the reservoir pressure above the bubble point pressure and favor oil recovery, injectors and producers can be operated under constant pressure. In this chapter, the laboratory waterflooding visualization experiments are designed and performed under different constant differential pressure boundaries to mimic real waterflooding processes in a reservoir.

The experimental procedures and observations for two waterflooding experiments are re-

ported. These experiments are performed under constant pressure boundaries and in a two-dimensional, heterogeneous glass-bead unconsolidated macro-model (James, 2012). The cumulative fluid production and displacement front are recorded as a function of time for further analysis.

The new streamline simulator developed in this research thesis is applied to model these experiments for three purposes. Through comparisons between the experimental and simulation results, the new streamline simulator is independently validated, the ability of the new streamline simulator to model physical problems is demonstrated, and the physical displacement processes under constant pressure boundaries in a heterogeneous porous medium is well understood.

In most oil reservoirs, permeability and porosity vary from location to location. These variations are known as heterogeneity and occur at all scales from kilometers down to microns. If a simulator fails to capture the fluid flow affected by the reservoir heterogeneity, it will lead to erroneous results. Therefore, the numerical and experimental studies of displacement processes in heterogeneous porous medium are important.

In this Chapter, the waterflooding experimental setup, porous medium and fluid properties, procedures, and results are first introduced. Then, the history matching and direct simulation of the physical displacement processes using the new streamline simulator is discussed.

7.2 Waterflooding Experiments

7.2.1 Experimental setup and apparatus

The main purpose of a waterflooding process is to create pressure gradient in a reservoir, and therefore drive the oil in place to the producer. Laboratory waterflooding experiments on a two-dimensional, glass-bead unconsolidated macro-model are performed to mimic the real displacement processes in a reservoir. These experiments visually reveal the physical

displacement process under constant differential pressure boundaries for the first time.

The schematic of the experimental setup is shown in Figure 7.1, and the equipment list is given in Table 7.1. The main components of the apparatus are syringe pump, oil and water accumulators, macro-model (porous medium), pressure transducers, graduated cylinders and camera. During a waterflooding experiment, the syringe pump is used to inject water under a constant pressure into the model. The producer is open to atmosphere. Two pressure transducers are used to record the pressure profile of the injector and producer. A set of graduated cylinders at the outlet is used to measure cumulative oil and water production during each time interval. The digital camera settled up above the model is used to record the displacement front movement every 30 seconds.

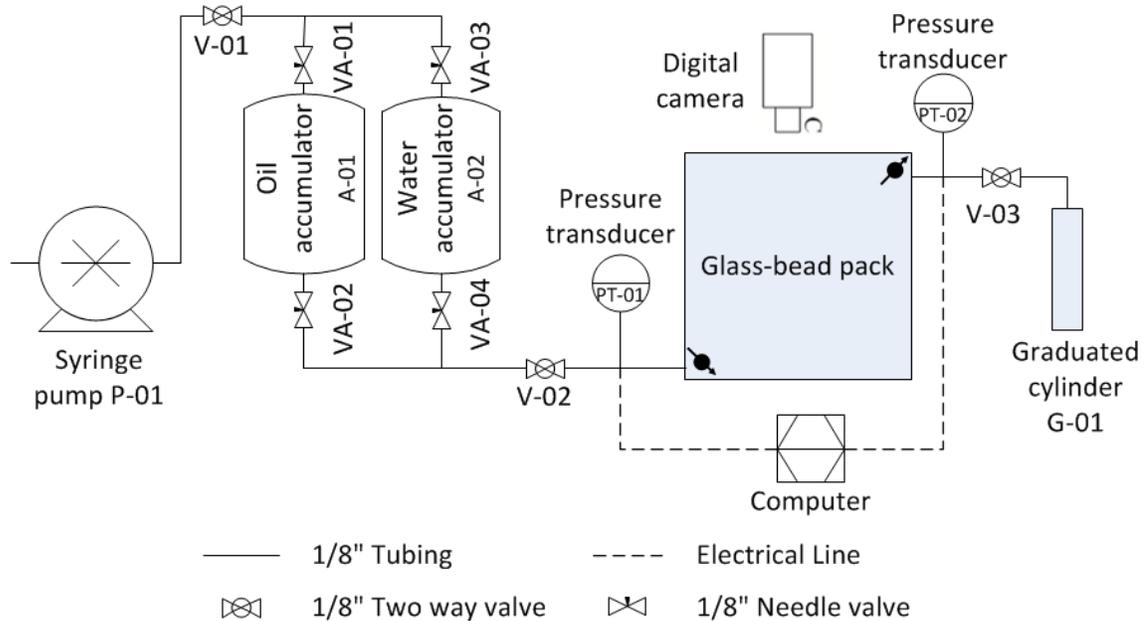


Figure 7.1: The schematic of the waterflooding experiments

Table 7.1: The equipment list

Description	Type	Range	Accuracy	Quantity
Syringe Pump	ISCO 500D	0 - 204 <i>ml/min</i>	0.1 <i>ml</i>	1
Accumulator	Custom Made	0 - 2 <i>L</i>	N/A	2
Pressure transducer	OMEGA PX409-100AUSB	0 - 100 <i>psia</i>	0.001 <i>psia</i>	2
Digital Camera	Canon Rebel XS	N/A	N/A	1
Computer	IBM Think Station	N/A	N/A	1
Light Box	Custom Made	N/A	N/A	1
Graduated Cylinder	20 <i>ml</i>	0 - 20 <i>ml</i>	0.1 <i>ml</i>	7

7.2.2 The macro-model and fluid properties

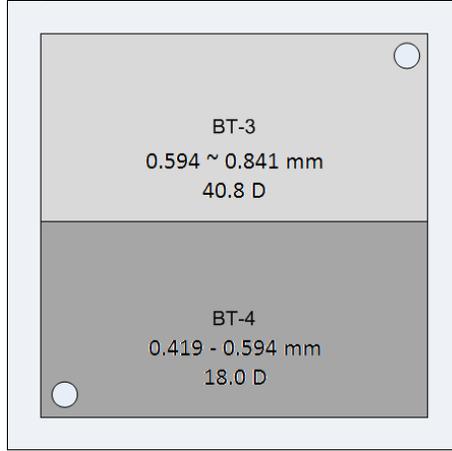
In this section, the macro-model and fluid properties of waterflooding experiments performed in this research are introduced.

The macro-model properties and well locations

A two-dimensional heterogeneous glass-bead pack macro-model is used to perform waterflooding experiments in this research. Figure 7.2 shows a schematic figure of the heterogeneous glass-bead pack macro-model (inner volume 30 *cm* × 30 *cm* × 1 *cm*), where the wells are located diagonally in the model.

As shown in Figure 7.2, uniformly sized glass beads are packed to give different properties in the model. The main reason for choosing glass-bead packs is to provide visualization; moreover, the glass beads are stable and have no chemical reactions with the fluids used in experiments. The detailed procedures of the glass-bead macro model fabrication is given in Appendix F.

The Plexiglas box is filled with two different uniformly sized glass beads: BT-3 (diameter, 0.594 - 0.841 *mm*) and BT-4 (diameter, 0.419 - 0.594 *mm*) (products from *International Surface Preparation Canada*). The two sizes of glass beads form the higher and lower



Dimension, $30\text{ cm} \times 30\text{ cm} \times 1\text{ cm}$

Figure 7.2: Heterogeneous glass-bead pack macro-model

permeable regions in the model. Due to the material and shape similarity for BT-3 and BT-4 glass beads, the relative permeability relations for oil and water in the higher and lower permeability regions are assumed to be the same.

The absolute permeability and porosity of BT-3 and BT-4 volumes are assumed to be constant when the same packing method is applied. Therefore, the absolute permeability and porosity of the higher and lower region can be measured outside the Plexiglas box using different uniform glass-bead packs, which are packed using the same method. In this research, a falling head method and a fluid saturation method is applied to measure the absolute permeability and porosity respectively. Detailed procedures of the permeability and porosity measurements are given in Appendixes G and H, respectively. The measurement results are given in Table 7.2.

Table 7.2: Properties for BT-3 and BT-4 glass beads

Glass beads	Diameter range, mm	Absolute permeability, D	Porosity
BT-3	0.594 - 0.841	40.8 ± 0.6	0.44 ± 0.01
BT-4	0.419 - 0.594	18.0 ± 0.5	0.44 ± 0.01

The macro-model and fluid compressibility

The porous medium and fluid applied are assumed to be incompressible. This assumption is valid since the pressure at the producer is less than 16 *psia* (1.10×10^5 *Pa*), pressure at the injector is less than 23 *psia* (1.58×10^5 *Pa*). The effects of porous medium and fluid compressibility are negligible within this pressure range. The validation of this assumption also indicates that the new streamline simulator developed in this research thesis is applicable for modeling these waterflooding processes. This is because this simulator assumes the velocity field is divergence, and this assumption is only valid when the porous medium and fluid are incompressible.

Fluid properties

The light mineral oil (Drakeol® 10B LT MIN OIL NF, from *Penreco*) and deionized (DI) water are applied to perform waterflooding experiments.

The viscosity and specific gravity of the reservoir fluids given in their Material Safety Data Sheets (MSDS) are summarized in Table 7.3. For good visualization results, a water-based blue dye is added to the water and an oil-based red dye is added to the light mineral oil. The effects of the dyes on fluid properties are assumed to be negligible.

Table 7.3: Fluid properties in the waterflooding experiments

Fluid	Viscosity	Specific gravity
Light mineral oil	20.2 <i>cP</i>	0.9
Deionized water	1.0 <i>cP</i>	1.0

7.2.3 Experimental procedures

Before the experiment begins, a series of safety tests is performed to ensure the experiment is safe when the maximum pressure on the macro-model is 25 *psia* (1.72×10^5 *pa*).

The experiment has three main stages:

1. The initial water imbibition (water displaces air);
2. The water drainage (oil displaces water);
3. The waterflooding (water displaces oil).

During the waterflooding stage, the pressures of injector and producer, the water injection rate, the cumulative oil and water production, and the displacement front are recorded as a function of time. The specific procedures are described below with more details.

Initial Water Imbibition

During the initial water imbibition, water is injected into the model to exhaust air initially present in the model. After the air is completely displaced from the model by water, it mimics the stage when the pores were completely saturated with water before the oil generated from the source rock occupies the pores. This process happens during the glass-bead pack macro-model fabrication when water is injected into the Plexiglas box to displace air and aid the compaction of glass beads (a detailed description is given in Appendix F). The initial water imbibition process is completed when the model fabrication process is finished. At this stage, no visual air bubbles can be observed and no more glass beads can be added in to the model.

The pore volume PV in the model can be calculated by knowing the porosity (ϕ) and the bulk volume (BV),

$$PV = \phi \times BV = 0.44 \times 30 \text{ cm} \times 30 \text{ cm} \times 1 \text{ cm} = 396 \text{ ml}. \quad (7.1)$$

Water Drainage

During the water drainage stage, oil is injected into the model to displace water. This process mimics the stage when the oil generated from source rocks begins to occupy reservoir pores. Water saturation is reduced as more oil is injected. After water reaches the connate water

saturation level, no more water can be displaced from the pores.

The specific procedures for the water drainage process are:

1. Start the water drainage process after the initial water imbibition process is finished;
2. Fill the pump with DI water;
3. Fill the oil accumulator A-01 in Figure 7.1 with red dyed silicone oil;
4. Expel the air from tubes before connecting the macro-model into the system;
5. Install the experimental apparatus according to Figure 7.1;
6. Open the following valves V-01, VA-01, VA-02, V-02, and V-03 (in Figure 7.1);
7. Start the pump and then inject oil from the oil accumulator at a constant injection rate (8 ml/min has been applied in these experiments);
8. Continue the constant flow rate oil injection after oil break-through;
9. Stop the pump when no more water is produced from the outlet, record the total water volume produced $V_{w,out}$;
10. Close all valves when the pressure at injector drops to the atmosphere pressure;
11. Calculate the original oil in place $OOIP$,

$$OOIP = V_{w,out}; \quad (7.2)$$

12. Calculate the connate water saturation S_{wc} ,

$$S_{wc} = \frac{PV - V_{w,out}}{PV}; \quad (7.3)$$

where PV is the pore volume of the macro-model.

13. To distribute the connate water saturation uniformly in the model, the model is placed stable and horizontal for at least 24 hours.

The values for $OOIP$, S_{wc} are reported later in section 7.2.4

Waterflooding under Constant Pressure Boundaries

During the waterflooding stage, water is injected into the model to displace oil. In a real oil field, waterflooding is a widely applied process to increase or maintain the reservoir pressure and thereby stimulate oil production.

Two waterflooding experiments are performed in this research. For these two experiments, the reservoir and fluid properties are kept the same, while the constant differential pressure between the injector and producer is designed to be 4 *psi* (2.7×10^4 *Pa*) and 5 *psi* (3.4×10^4 *Pa*), respectively. In these experiments, the syringe pump used in these experiments cannot automatically keep a constant injection pressure, thus, the producer is open to atmosphere pressure and the injection rate is manually adjusted during the entire displacement processes to keep the differential pressure between the injector and producer at the designed constant value. The pressures at the injector and producer, the water injection rate, cumulative oil and water production, and the displacement front are recorded as a function of time.

The specific procedures for waterflooding process are:

1. Start the waterflooding process after the water drainage process is finished;
2. Fill the pump with DI water;
3. Fill the water accumulator A-02 in Figure 7.1 with blue dyed DI water;
4. Expel air from the tubes before connecting the macro-model into the system;
5. Install the experimental apparatus according to Figure 7.1;
6. Open the following valves V-01, VA-03, VA-04, V-02; while keep the outlet valve V-03 closed (in Figure 7.1);
7. Use a camera to take photos of the model every 30 seconds;
8. Record the pressures of injector and producer automatically every 5 seconds;
9. Start the pump and then inject the blue dyed DI water from the water accumulator at a constant rate of 20 *ml/min*;
10. Read the pressure value from the pressure transducer PT-01 as it gradually increases;

11. Open the outlet valve V-03 when the injector pressure reaches the designed value;
12. Adjust the water injection rate throughout the waterflooding processes to keep the differential pressure at a constant value, record the water injection rate and its corresponding time interval;
13. As the produced fluids accumulate in the graduated cylinder, change the graduated cylinder every 1 or 2 minutes to keep the fluid volume within the measurement range of the graduated cylinder, record the time interval and the order of graduated cylinders;
14. Stop the constant pressure boundary experiment and change the graduated cylinder when the differential pressure cannot be kept constant at the constant value;
15. Continue to inject water at a constant flow rate (for example 8 *ml/min*), stop the pump and camera until no more oil can be displaced;
16. Close all the valves when the injector pressure drops to the atmosphere pressure;
17. Determine the total oil $V_{o,out}$ produced, and calculate the residual oil saturation S_{or} ,

$$S_{or} = \frac{OOIP - V_{o,out}}{PV}; \quad (7.4)$$

18. Read the oil and water volume in each graduated cylinder during the constant pressure boundary period;
19. Calculate the cumulative water and oil production during the constant pressure boundary period by summing the fluid volumes in graduated cylinders in order,

$$Q_w(n) = \sum_{i=1}^n V_w(i); \quad (7.5)$$

$$Q_o(n) = \sum_{i=1}^n V_o(i); \quad (7.6)$$

where, $Q_w(n)$ and $Q_o(n)$ is the cumulative water and oil production when the graduated cylinder number n stops collecting fluid, respectively; $V_w(i)$ and $V_o(i)$ is the water and oil volume in the graduated cylinder number i , respectively. The corresponding time is determined by,

$$t(n) = \sum_{i=1}^n \Delta t(i); \quad (7.7)$$

where, $\Delta t(i)$ is the time interval for the graduated cylinder number i collects the

produced fluid.

20. Calculate the cumulative water injected during the constant pressure boundary period, by summing up the injection volume at each constant injection rate interval in sequence,

$$Q_{inj}(n) = \sum_{i=1}^n q(i)\Delta t_{inj}(i) \quad (7.8)$$

where, $Q_{inj}(n)$ is the cumulative water injected at interval n ; $q(i)$ is the injection rate at interval i ; $\Delta t_{inj}(i)$ is the time interval at i . The corresponding time is calculated using,

$$t_{inj}(n) = \sum_{i=1}^n \Delta t_{inj}(i); \quad (7.9)$$

The recorded values are reported later in section 7.2.4.

By following the experimental procedures introduced above, two waterflooding visualization experiments under constant differential pressure boundaries are performed. The results of these two experiments are reported in the next subsection.

7.2.4 The experimental results

In this subsection, the pressure data and the production and injection profiles for two waterflooding visualization experiments are reported. These results are mainly applied to validate the applications of the new streamline simulator developed in this research thesis.

Experimental pressure profiles

The pressure profile for the two experiments are given in Figure 7.3 and 7.4. These data are automatically collected by the pressure transducers next to the injector and producer. The average differential pressure for the first experiment is 4.4 *psi*, with a standard deviation of 0.21; the average differential pressure for the second experiment is 5.2 *psi*, with a standard deviation of 0.35. The difference between two experiments are caused by their different constant pressure boundary conditions.

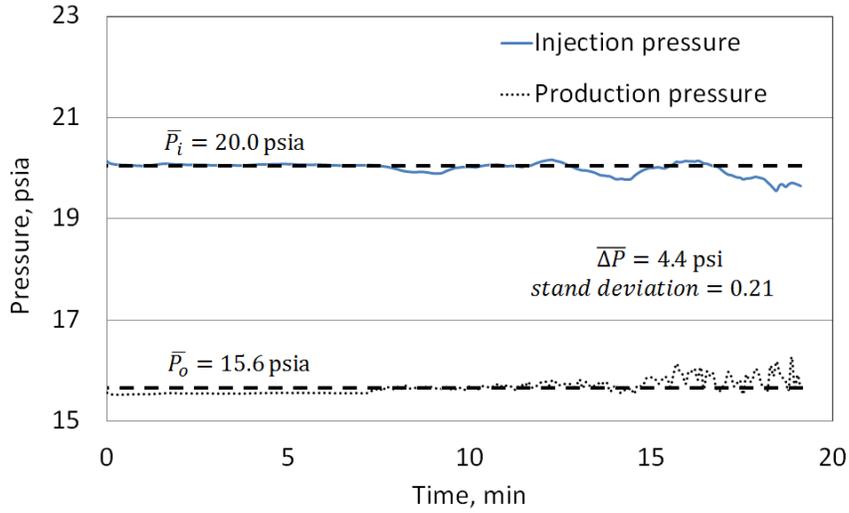


Figure 7.3: The pressure profile for the first experiment, $\Delta P = 4.4 \text{ psi}$

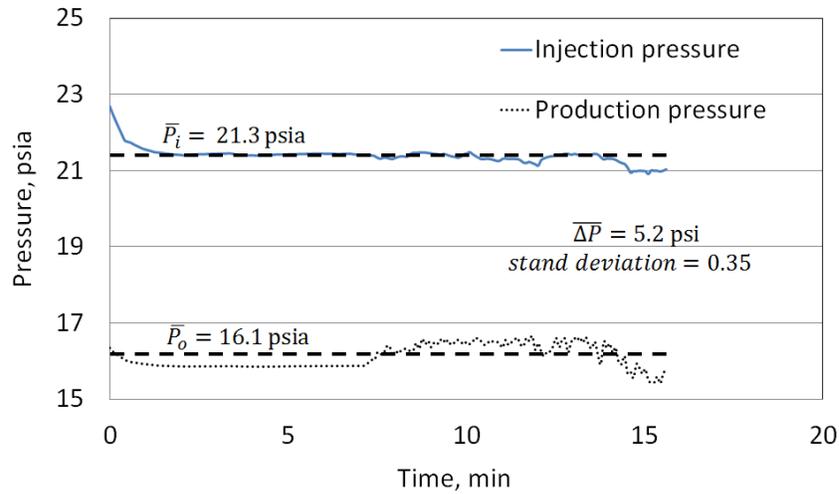


Figure 7.4: The pressure profile for the second experiment, $\Delta P = 5.2 \text{ psi}$

Observed cumulative production and injection profiles

The cumulative production and injection profiles are shown in Figure 7.5. As we can observed in this figure, the cumulative production and injection profiles for these two experiments are different. Comparing these two experiments, when the differential pressure is larger, the water breakthroughs at earlier time, and the cumulative injection and production profiles are steeper. They both indicate that the phase flow rates (for both oil and

water) is increased when the differential pressure ΔP is increasing.

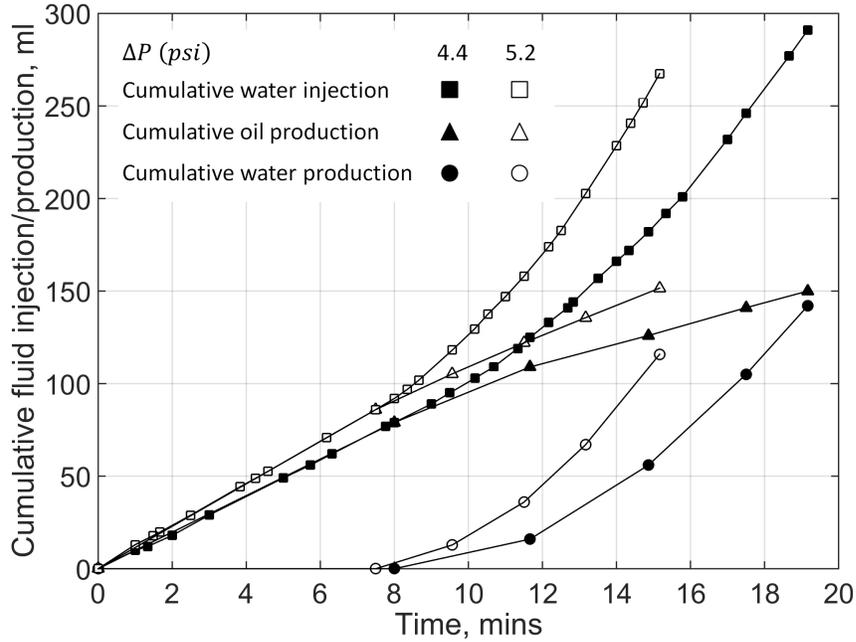


Figure 7.5: The experimental cumulative injection and production profiles

Recorded connate water and residual oil saturation

The recorded original oil in place, the total oil produced, the connate water saturation and the residual oil saturation are summarized in Table 7.4.

Table 7.4: The connate water saturation and residual oil saturation for the two experiment

	$\Delta P = 4.4 \text{ psi}$	$\Delta P = 5.2 \text{ psi}$
Original oil in place (<i>OOIP</i>)	299 ml	296 ml
Total oil produced ($V_{o,out}$)	227 ml	230 ml
Connate water saturation (S_{wc})	0.24	0.25
Residual oil saturation (S_{or})	0.18	0.17

Comparing the values in Table 7.4, the results for connate water (S_{wc}) and residual oil (S_{or}) saturations in these two experiments are consistent. The assumption of the same relative permeability relations for both experiments is valid.

The main driving force in the experiments

The main driving force can be determined by evaluating the Capillary number Ca , *i.e.*,

$$Ca = \frac{\mu v}{\gamma}; \quad (7.10)$$

where, $\mu = 1.0 \text{ cP}$ is the displacing phase (water) viscosity, v is the fluid flow velocity, and $\gamma = 0.049 \text{ N/m}$ is the interfacial tension between silicone oil and water (Stan *et.al*, 2009). It represents the relative effect of viscous forces versus interfacial tension acting across an interface between oil and water. For high capillary numbers greater than 10^{-5} , flow in porous medium is dominated by viscous forces (Lake, 1989).

In these experiments, the minimum velocity for a viscous dominated flow can be determined by substituting $Ca = 10^{-5}$ into Eq. 7.10,

$$v_{min} = \frac{\gamma Ca}{\mu_o} = \frac{0.049 \times 1 \times 10^{-5}}{1.0 \times 10^{-3}} = 4.9 \times 10^{-4} \text{ m/s} = 2.94 \text{ cm/min}. \quad (7.11)$$

The viscous forces dominates the water displacement processes if the flow velocity is greater than 2.94 cm/min .

The water front movement velocity can be roughly estimated using the pictures taken by the camera before the water breakthrough. As shown in Figure 7.6 and 7.7, in these two experiments, the water and oil phases are both continuous, the velocity of the water front is around 2 cm/min to 3 cm/min , and it is very close to 2.94 cm/min . In this case, the capillary effects may be neglected on this macroscopic level for reservoir simulations, since the combined effects of viscous and capillary forces are lumped into the relative permeability relations (Cense and Berg, 2009). In the following sections, the new streamline simulator is applied to model the waterflooding processes. If the agreement between the simulated and observed results is acceptable, the above statement about capillary effects will be proven to be true.

In Figure 7.6, the water movement in the bottom of this model is fast, this is edge effect caused by defective artificial model fabrication.

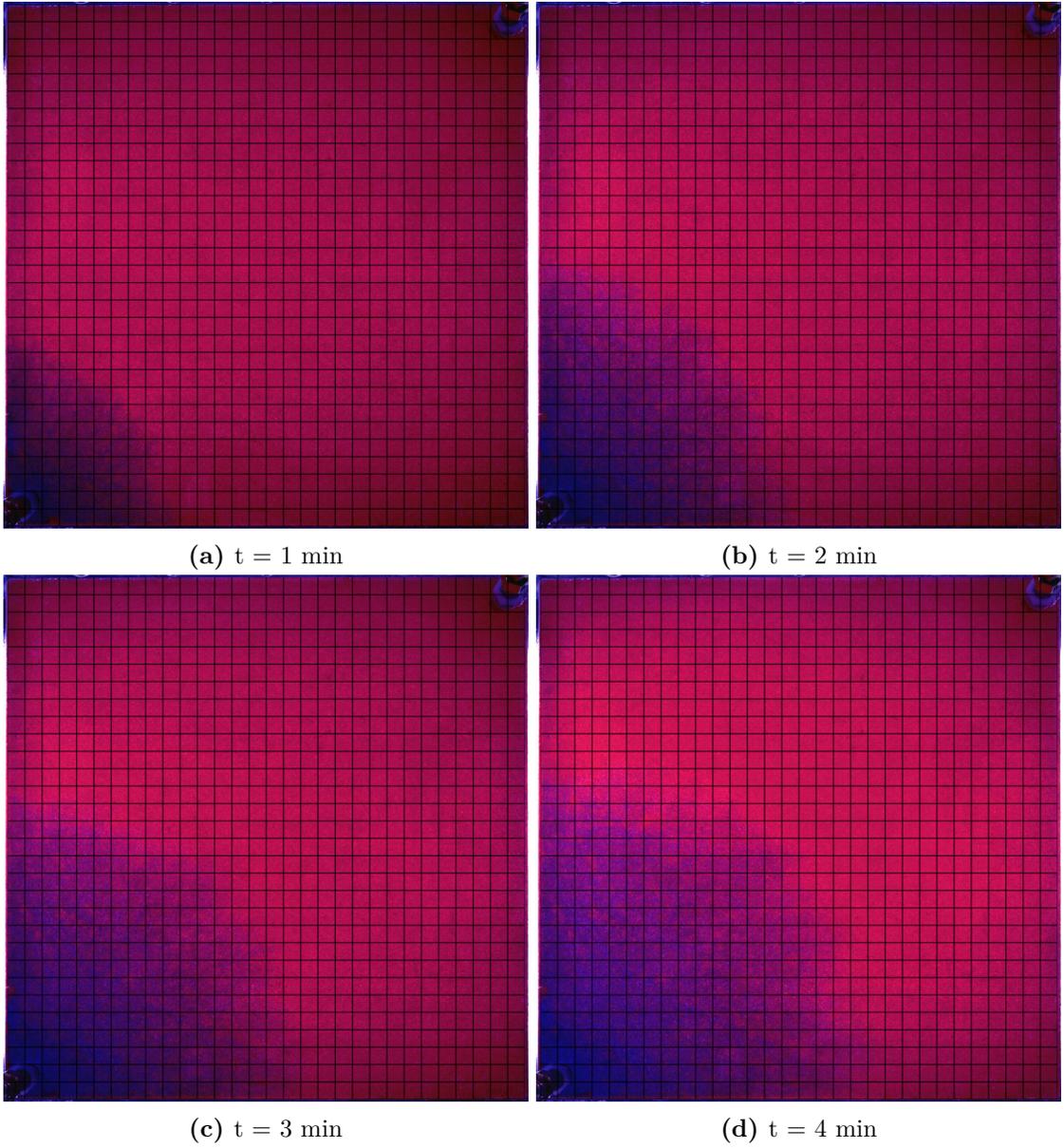


Figure 7.6: Displacement front movement before water breakthrough, $\Delta P = 4.4$ *psi*

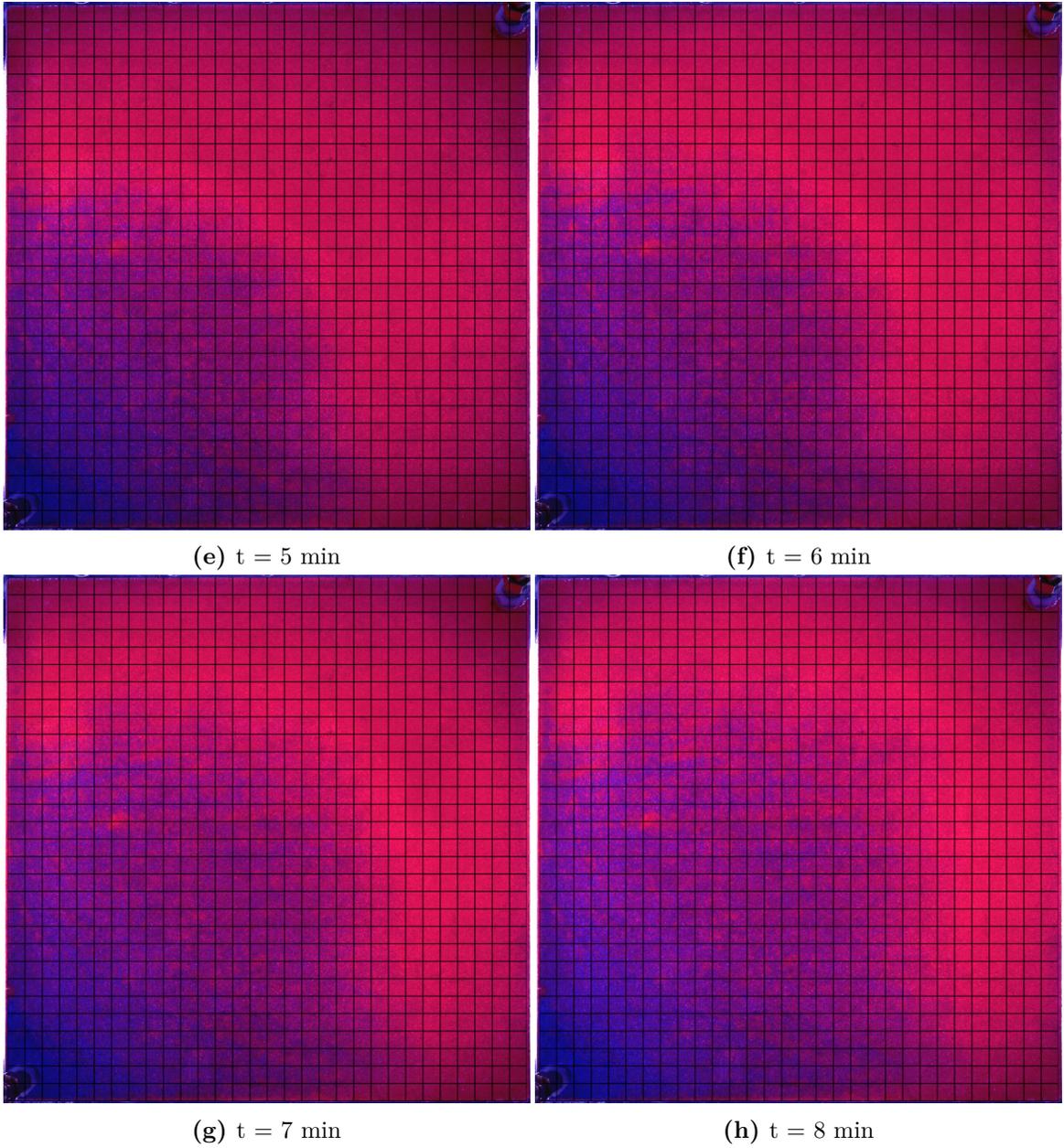
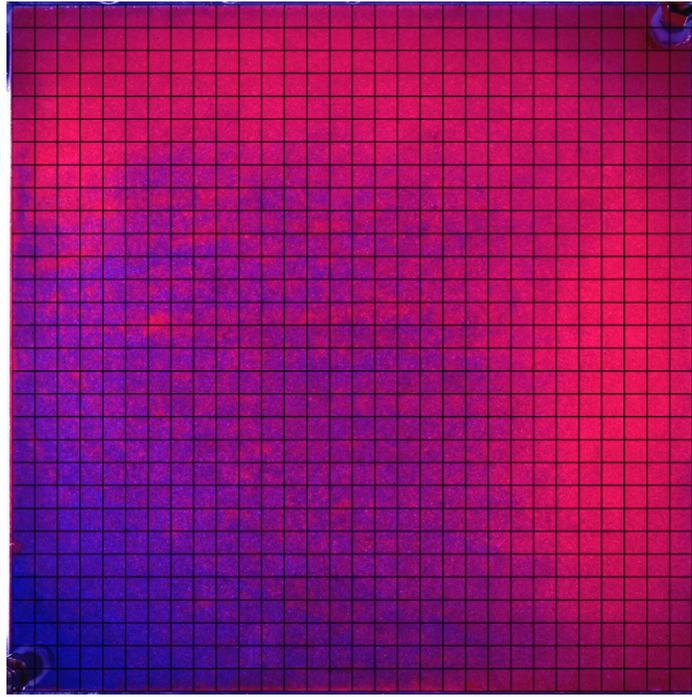


Figure 7.6: Displacement front movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)



(i) $t = 9 \text{ min}$

Figure 7.6: Displacement front movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)

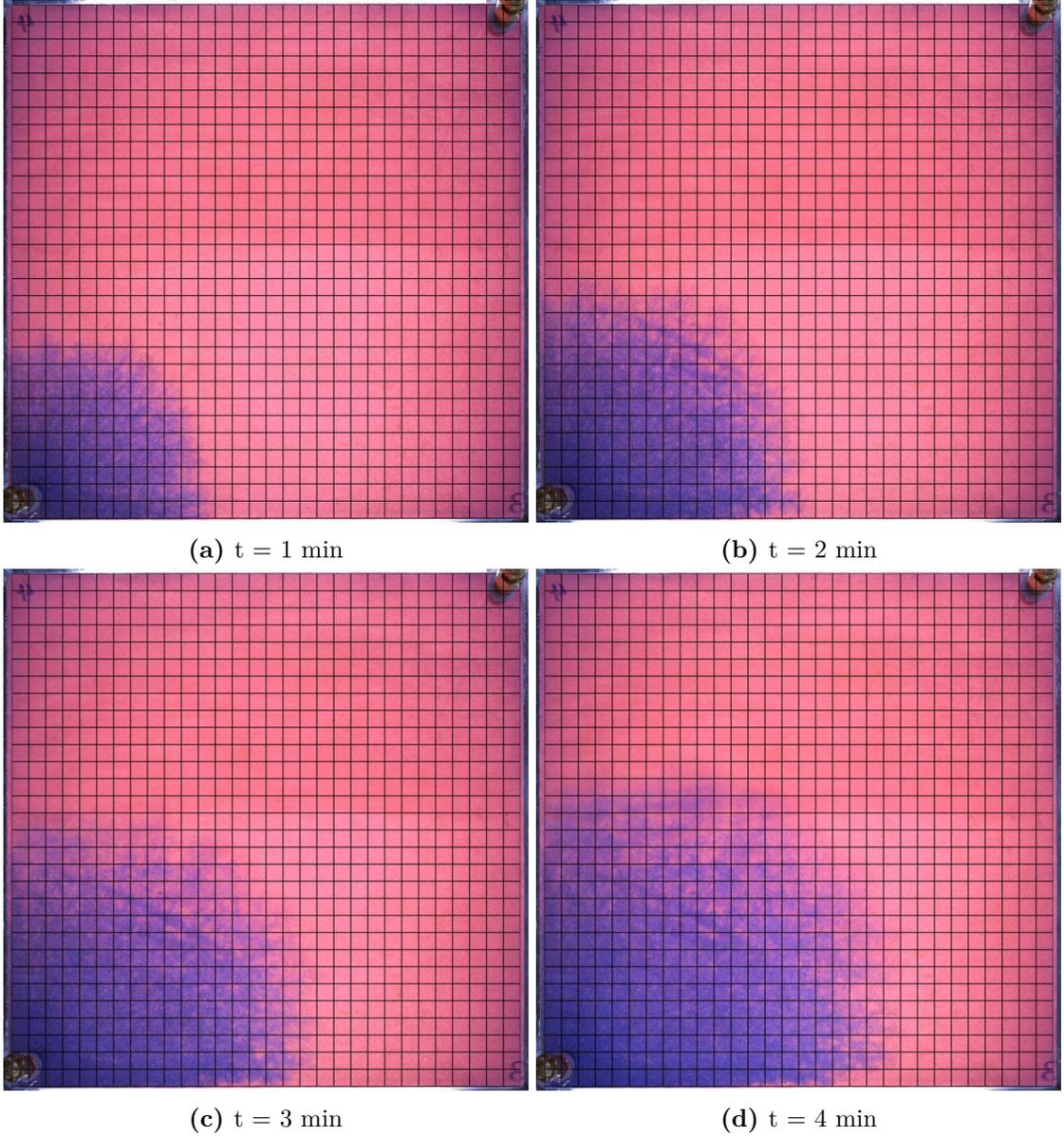
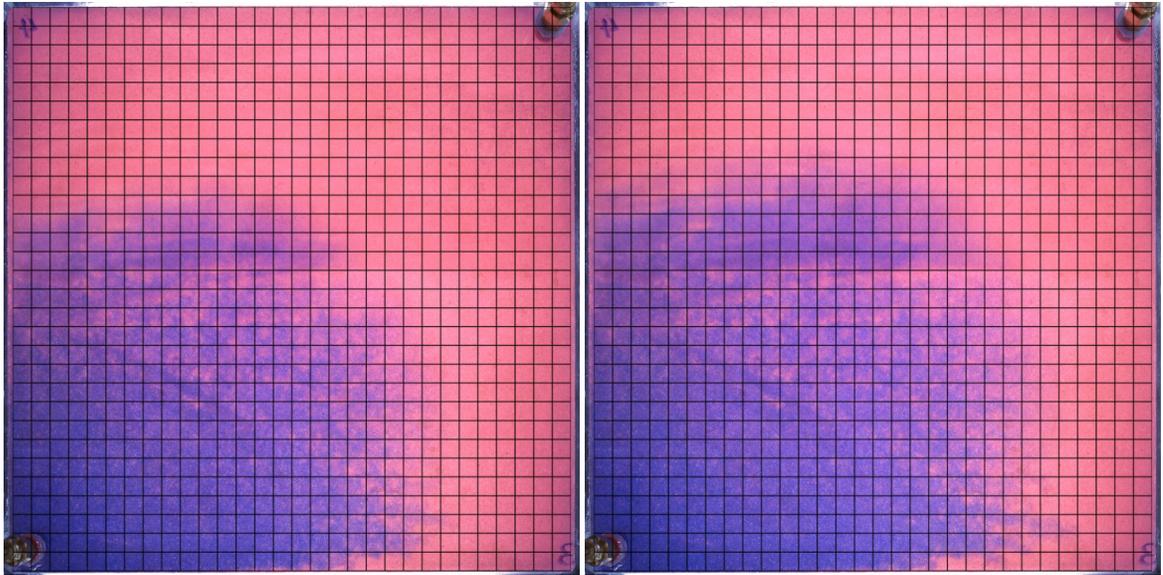
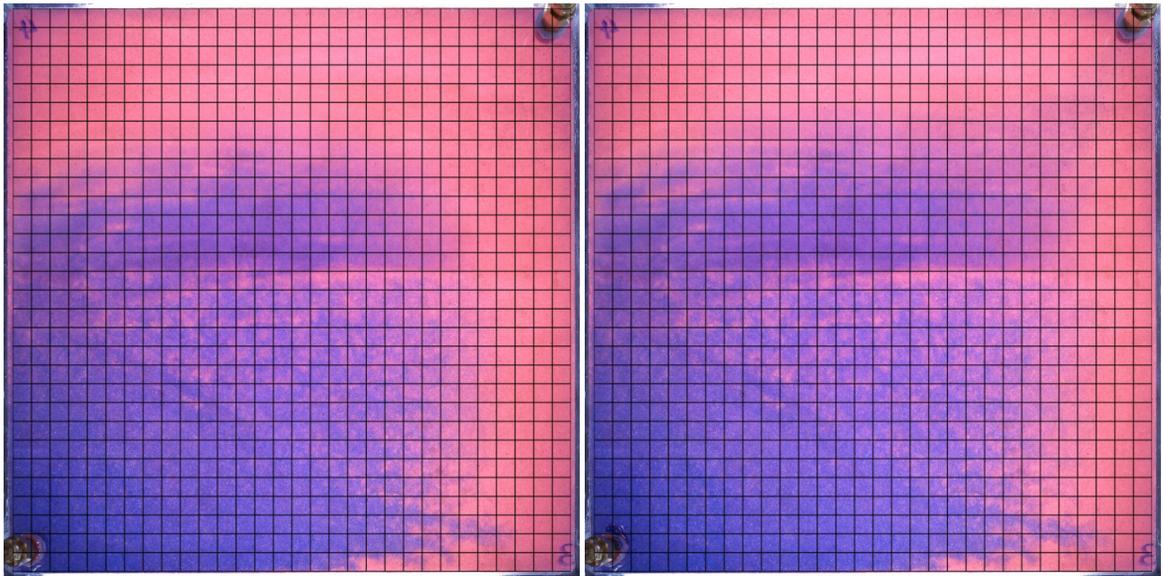


Figure 7.7: Displacement front movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$



(e) $t = 5 \text{ min}$

(f) $t = 6 \text{ min}$



(g) $t = 7 \text{ min}$

(h) $t = 8 \text{ min}$

Figure 7.7: Displacement front movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$ (cont.)

7.3 Streamline Simulation Results

Reservoir simulators are widely used engineering tools for making decisions on the development of oil fields. One of the major application of a reservoir simulator is to accurately model the flow performance in a reservoir. This is usually achieved by two methods: first, by adjusting the uncertain parameters in a reservoir model, so the simulator can successfully match the historical behavior of fluid flow processes; second, by directly feeding the measured/calculated parameters into the simulator, and then model displacement processes. In this research thesis, the ability of the new streamline simulator is validated and demonstrated by following similar methods.

More specifically, in this section, the relative permeabilities of oil and water are determined through history matching the observed cumulative injection and production profile of the first experiment ($\Delta P = 4.4 \text{ psi}$); secondly, the new streamline simulator is independently validated by successfully modeling the second experiment ($\Delta P = 5.2 \text{ psi}$) using the same relative permeabilities; finally, the applications of the new streamline simulator is demonstrated by analyzing the waterfront movements for both experiments using simulated results, and the physical displacement processes can be well understood.

7.3.1 History matching waterflooding (first experiment), $\Delta P = 4.4 \text{ psi}$

History matching is the process of adjusting the uncertain parameters in the numerical reservoir model to achieve a reasonable agreement between the simulated and observed historical field/well behavior. The main objective of the history match is to build a reservoir model that can accurately mimic the historical and other possible flow performances in the reservoir.

In this section, the new streamline simulator is applied to history match the cumulative production and injection profile for the first experiment ($\Delta P = 4.4 \text{ psi}$) shown in Figure

7.5. The uncertain parameters adjusted in the history matching process are the relative permeabilities of oil and water.

The history matching performed in this section is a trial-and-error approach. In brief, the procedures are given as follows:

1. Decide the criteria for a successful match;
2. Input the reservoir properties, fluid properties, and well conditions to the new streamline simulator (given in Table 7.5);
3. Decide and input the initial trial data for relative permeabilities of oil and water into the new streamline simulator;
4. Run simulation for the historical period;
5. Compare the simulated results to the observed cumulative injection and production profiles;
6. Adjust the relative permeability parameters of oil and water to improve the match;
7. Continue with Steps 4 through 6 until the criteria established in Step 1 are met.

Alternatively, a non-linear regression could be employed. Since the relationship between the relative permeabilities and production profiles is highly non-linear (Eq. 6.13), model it with a linear approximation cannot be used. Furthermore, the non-linear regression is not considered in this research since it cannot deliver better history match results than the trial-and-error approach in this particular case (as shown later in the text). Besides, a non-linear regression history marching is a complex process, and it is beyond the scope this research.

The quality of the history matching results can be evaluated by the average relative error between the observed and predicted data (Elsayed *et al.*, 1993). In this case, since the objective of this history match process is to fit the cumulative injection and production profile, the average relative error is defined as,

$$e = \frac{\sum_{i=1}^N \left| \frac{Q_{predicted}(i) - Q_{observed}(i)}{Q_{observed}(i)} \right|}{n}; \quad (7.12)$$

where, $Q_{predicted}$ and $Q_{observed}$ is the predicted and observed cumulative injection or production, respectively; i is the observation point; N is the total number of observation points.

The criterion for the successive history matching process defined in Step 1 is the average relative error for the cumulative injection and production profile for both the oil and water phase. The average relative error should be less than 10%.

The input parameters in Step 2 are summarized in Table 7.5.

Table 7.5: Input parameters for the history match

BT-3 region dimension	15 cm × 15 cm × 1 cm
BT-4 region dimension	15 cm × 15 cm × 1 cm
Absolute Permeability for the BT-3 region	40.83 D
Absolute Permeability for the BT-4 region	18.03 D
Connate water saturation	0.24
Residual oil saturation	0.18
Porosity	44%
Injector location	(2 cm, 2 cm)
Producer location	(29 cm, 29 cm)
Water viscosity	1.0 cP
Oil viscosity	20.2 cP
Constant differential pressure	4.4 psi

Following the procedures and criteria introduced above, the relative permeability relations that can successfully match the experimental results are obtained. These relations are presented in Table 7.6 and Figure 7.8. In this Table, a_w , n_w , a_o , and n_o are the parameters for relative permeability relations in Corey's model (1954), Eq. 6.48 and 6.49,

$$k_{rw}(s_w) = a_w \left(\frac{s_w - S_{wc}}{1 - S_{wc} - S_{or}} \right)^{n_w}; \quad k_{ro}(s_w) = a_o \left(\frac{1 - s_w - S_{or}}{1 - S_{wc} - S_{or}} \right)^{n_o};$$

where, S_{wc} and S_{or} is connate water and residual oil saturation, respectively; a_w and a_o are maximum relative permeabilities for water and oil; n_w and n_o are exponents for water and oil.

Table 7.6: The relative permeability parameters in Corey’s model for a successful history match result

	a_w	n_w	a_o	n_o
Initial trial	1.0	2.0	1.0	2.0
Final result	0.39	2.0	1.0	1.2

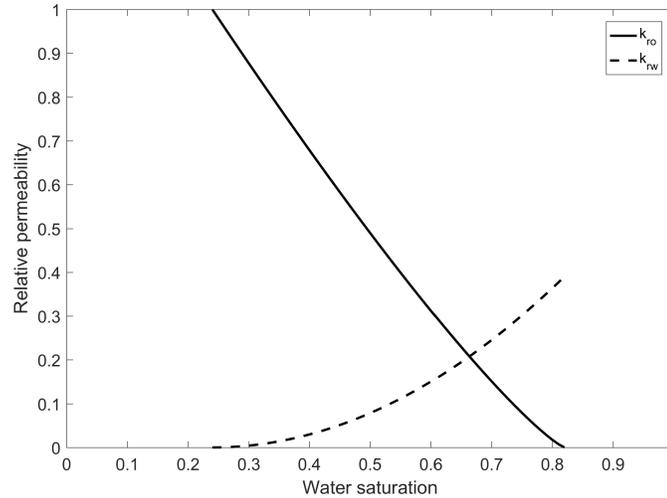


Figure 7.8: The relative permeability curves for a successful history match result

As shown in Figure 7.9 and 7.10, the predicted cumulative oil and water injection and production profiles are matched with the observation results when the relative permeability relations for oil and water given in Table 7.6 are applied. The average relative errors for matching the injection and production profiles are given in Table 7.7. Based on the criterion made for a successive history match ($e < 10\%$ in Eq. 7.12), this match using a trial-and-error approach is successful.

Table 7.7: The average relative errors for matching the cumulative injection and production profiles in the first experiment, $\Delta P = 4.4 \text{ psi}$

Parameter	Average relative error
Cumulative water injection	4.65%
Cumulative water production	2.81%
Cumulative oil production	2.27%

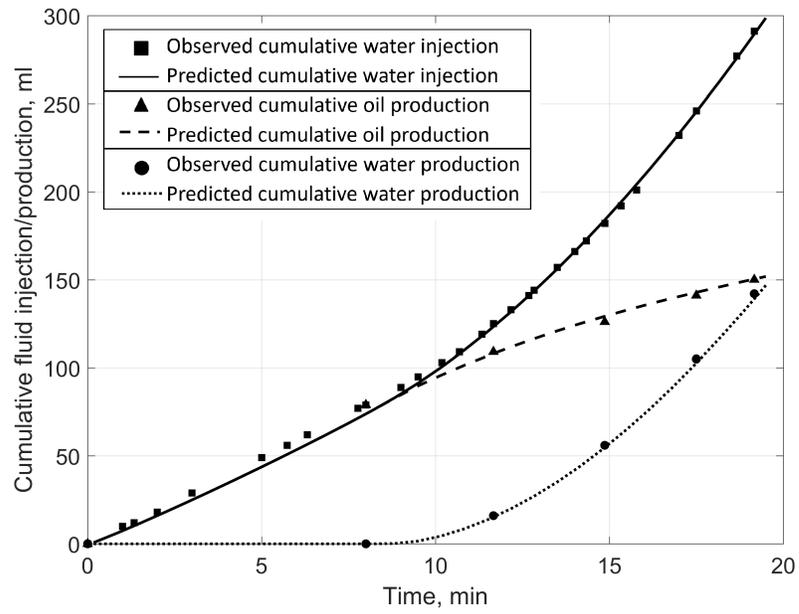


Figure 7.9: Experimental and history match results for the experiment, $\Delta P = 4.4 \text{ psi}$

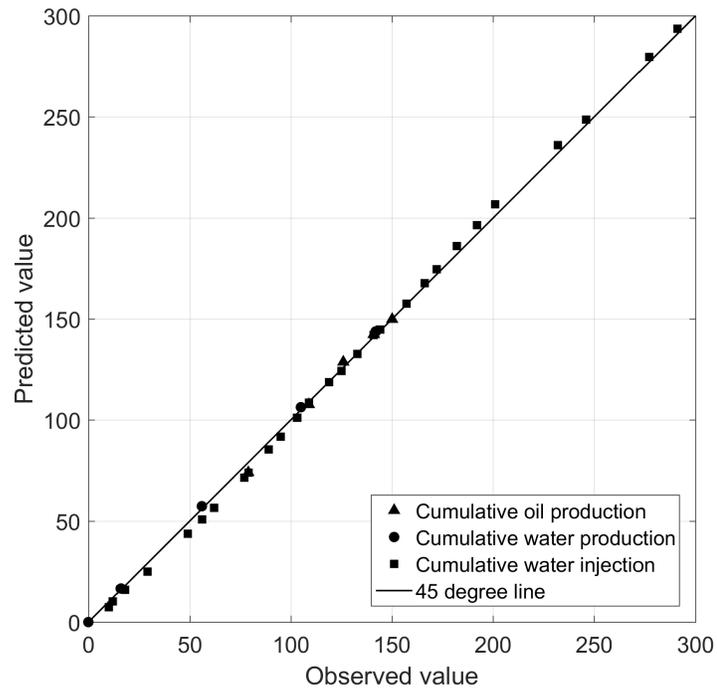


Figure 7.10: Observed vs Simulated results for the experiment, $\Delta P = 4.4 \text{ psi}$

The objectives of history matching are not limited to achieve a good match between the predicted and observed results. Moreover, history matching is also aiming to build a reservoir model that can provide accurate predictions for other possible flow performances in the reservoir. In this research thesis, the ability of the new streamline simulator in modeling physical displacement problems are also demonstrated by comparing the simulated results with the observed results for the second waterflooding experiment ($\Delta P = 5.2 \text{ psi}$).

7.3.2 Waterflooding simulation (second experiment), $\Delta P = 5.2 \text{ psi}$

In this section, the second waterflooding experiment is simulated by directly feeding parameters into the new streamline simulator. The agreement between the simulated and the observed results for the cumulative injection and production profiles is evaluated, and is used to validate the simulated results.

Table 7.8: Input parameters to directly simulate for waterflooding at $\Delta P = 5.2 \text{ psi}$

BT-3 region dimension	$15 \text{ cm} \times 15 \text{ cm} \times 1 \text{ cm}$
BT-4 region dimension	$15 \text{ cm} \times 15 \text{ cm} \times 1 \text{ cm}$
Absolute Permeability for the BT-3 region	40.83 D
Absolute Permeability for the BT-4 region	18.03 D
Connate water saturation	0.24
Residual oil saturation	0.18
Porosity	44%
Injector location	$(2 \text{ cm}, 2 \text{ cm})$
Producer location	$(29 \text{ cm}, 29 \text{ cm})$
Water viscosity	1.0 cP
Oil viscosity	20.2 cP
Constant differential pressure	5.2 psi
Maximum water relative permeability (a_w)	0.39
Maximum oil relative permeability (a_o)	1.0
Relative permeability exponent for water (n_w)	2.0
Relative permeability exponent for oil (n_o)	1.2

Comparing the first and the second waterflooding experiments, the only difference is the boundary pressures. Therefore, the same input parameters (given in Table 7.5), relative permeability relations (Table 7.6), with a different boundary pressure (5.2 *psi*) are used in the new streamline simulator to model the displacement process in the second experiment. The input parameters are summarized in Table 7.8.

The displacement process at the experimental time frame can be simulated by using the new streamline simulator. The predicted and observed cumulative injection and production profiles are given in Figure 7.11 and 7.12. The agreement between the prediction and the observation is evaluated by calculating the relative errors (Eq. 7.12) in predicted injection and production profiles; and its results are given in Table 7.9. According to the criteria for a successive match, the relative errors for cumulative injection and production volume are less than 10%, the streamline simulator can deliver sufficiently accurate simulation results for the second waterflooding experiment.

Based on the above discussions, the new streamline simulator can accurately model the physical waterflooding displacement processes in heterogeneous porous method through history matching or directly feeding relabel parameters when the underlying assumptions are met.

Table 7.9: Average relative error in simulated and observed cumulative rates in the second experiment, $\Delta P = 5.2$ *psi*

Parameter	Average relative error
Cumulative water injection	6.40%
Cumulative water production	2.52%
Cumulative oil production	2.37%

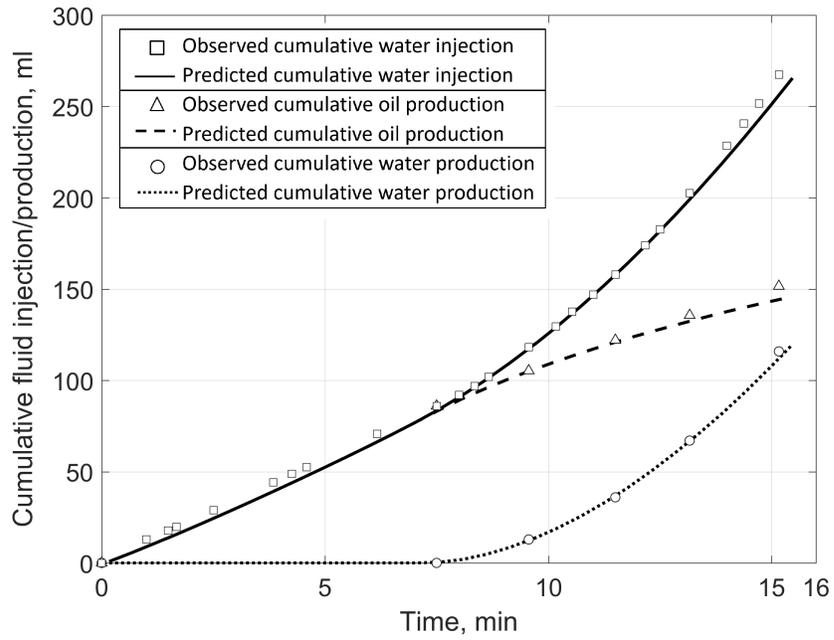


Figure 7.11: Experimental and simulated results for the experiment, $\Delta P = 5.2 \text{ psi}$

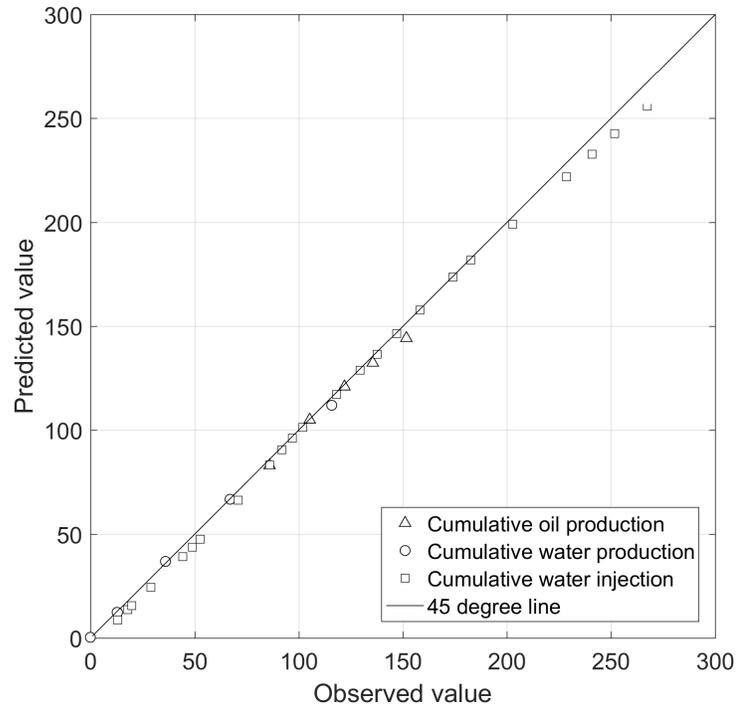


Figure 7.12: Experimental and simulated results for the experiment, $\Delta P = 5.2 \text{ psi}$

7.3.3 Analyzing the visualization results

In this section, the new streamline simulator is applied to study the displacement front movement from both waterflooding experiments and to demonstrate the applications of the simulator. During the waterflooding experiments, the water displacement front was captured at every 30 seconds before the water breakthrough. Figure 7.15 and 7.16 show some of the displacement fronts recorded in these two experiments every minute. In the same picture, the simulated displacement front and streamlines generated using the new streamline simulator are also given. These waterfront are estimated as a function of both time and space. In these streamlines, five streamlines are evenly selected and numbered as 1 to 5 (from left to right).

As shown in Figure 7.15 and 7.16, the simulated displacement fronts can accurately capture the movement of waterfront before water breakthrough. Therefore, the simulation results can be applied to quantify some physical effects, which cannot be evaluated by the experimental results only. For example, the capillary effects cannot be quantified in terms of capillary number using the visualization results only. This is because the fluid flow velocity is the variable in calculating capillary number ($Ca = \frac{\mu v}{\gamma}$), and it cannot be evaluated without the assists from streamlines. The fluid velocity changes instantaneously in value and direction, but the flow direction cannot be visually estimated without streamlines. Nevertheless, the capillary number can be accuracy as a function of both time and space using the new streamline simulator. In these two waterflooding experiments, following the movement of the waterfront along different streamlines (1 to 5 shown in Figure 7.15 and 7.16), the capillary number results are given in Figure 7.13 and 7.14. In the same Figure, some experimental results for capillary number along the streamlines 1 and 3 are also evaluated based on the flow directions indicated by streamlines (a detailed description is given in Appendix I). These values are calculated using the averaged waterfront movement speed along streamlines, which is determined using the results from Figure 7.15 and 7.16. The experimental and simulated results for capillary number again shows good agreement.

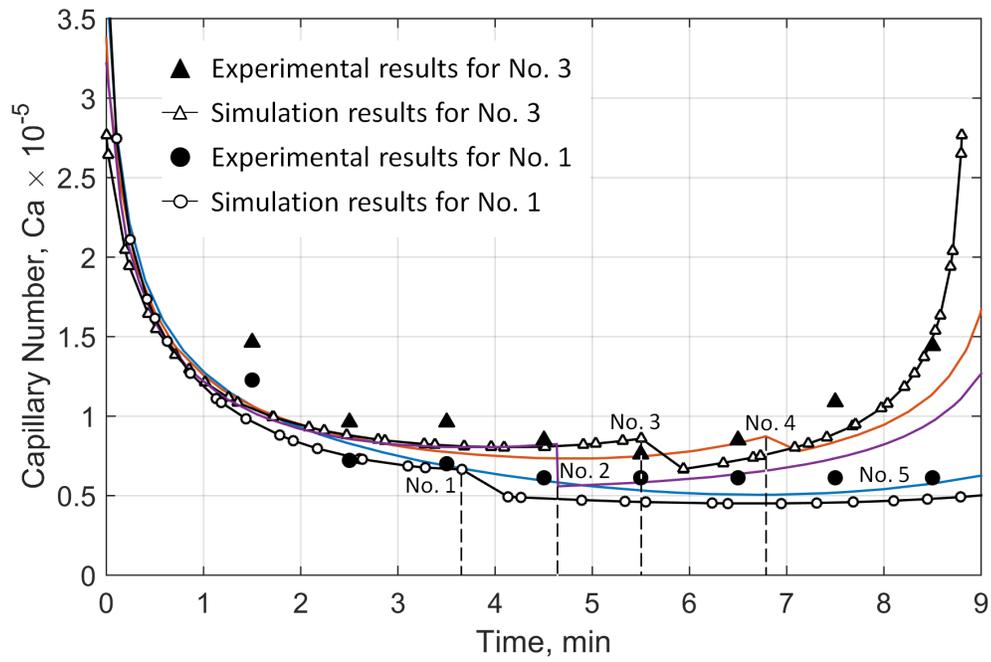


Figure 7.13: Capillary number at the waterfront for different streamlines, $\Delta P = 4.4 \text{ psi}$

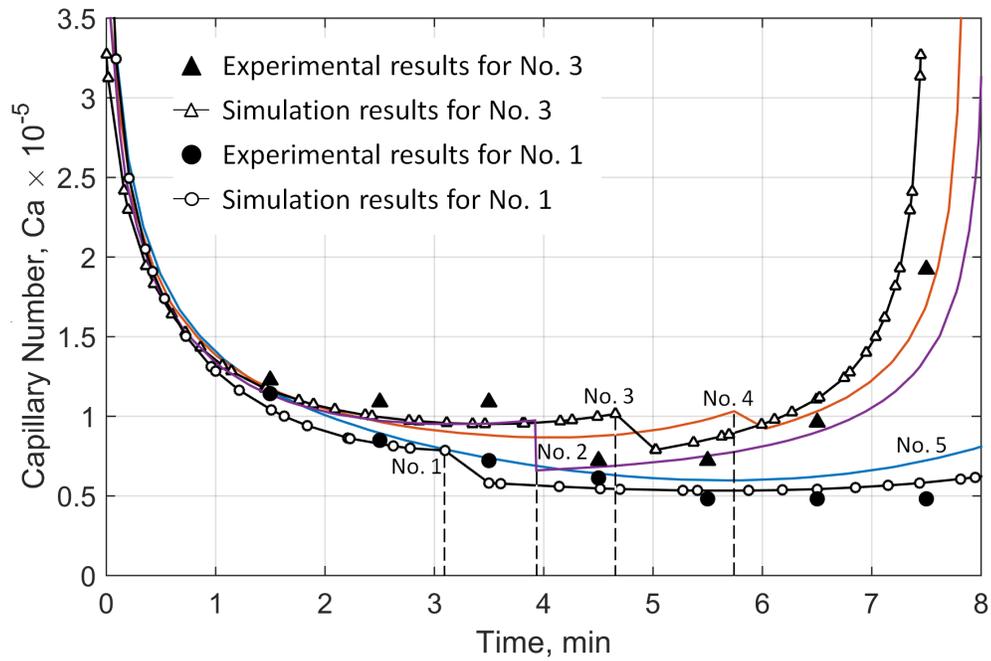


Figure 7.14: Capillary number at the waterfront for different streamlines, $\Delta P = 5.2 \text{ psi}$

In general, the capillary number varied in the range of 5×10^{-6} to 3×10^{-5} for the selected streamlines, this observation is consistent with the previous conclusion made for the capillary number in the section 7.2.4. The capillary number is larger for the second experiment ($\Delta P = 5.2 \text{ psi}$) than the first experiment ($\Delta P = 4.4 \text{ psi}$). This can be easily understood since the capillary number is proportional to the fluid velocity ($Ca = \frac{\mu v}{\gamma}$), and the fluid velocity increases with increasing boundary differential pressure.

We can also observe from Figure 7.13 and 7.14, that the capillary number profiles for streamlines 2, 3, and 4 are relatively large at the beginning and ending time frame. During these time, the fluid velocity at the waterfront is relatively large since it flows closely to the injector/producer (as shown in Figure 7.15i and 7.16h), and the capillary number increases with increasing fluid velocity. On the other hand, the capillary number remains small for streamlines 1 and 5 at later times, since the waterfront at these streamlines are still far from the producer (as shown in Figure 7.15i and 7.16h), therefore, the fluid velocity and the capillary number remain small.

Additionally, the capillary number decreases non-smoothly for streamlines 1 to 4. This phenomena occurs when the fluid flow velocity changes rapidly. In these two particular experiments, the capillary number drops significantly when the waterfront travels from the high permeable region ($K = 40.8 D$) to the low permeable region ($K = 18.0 D$), and the fluid velocity decreases.

Overall, the simulated displacement fronts can accurately capture the overall shape of waterfronts and the displacement front movement from the two experiments is well understood. In these two macroscopic waterflooding experiments, the capillary effects can be ignored in streamline simulations.

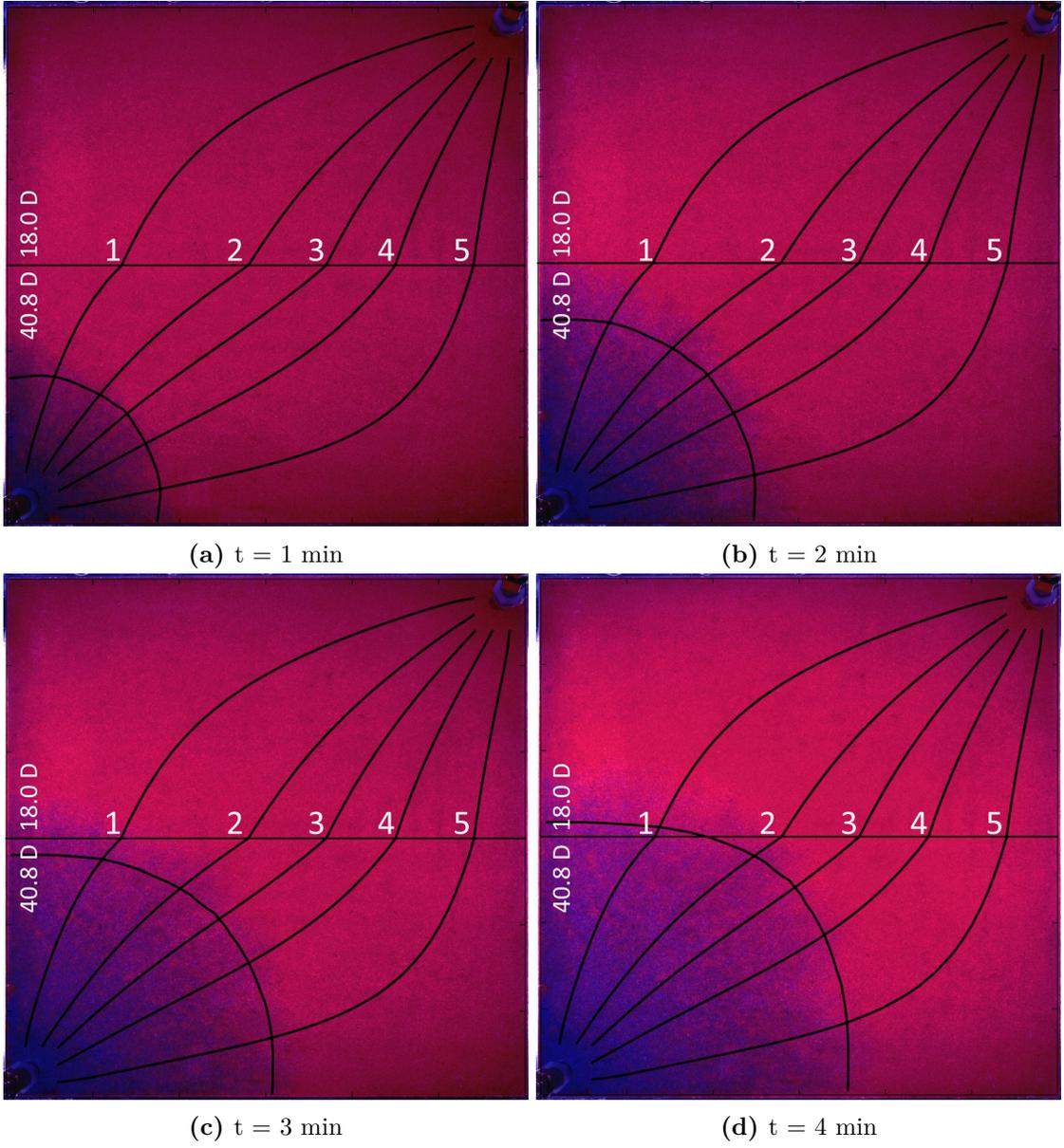
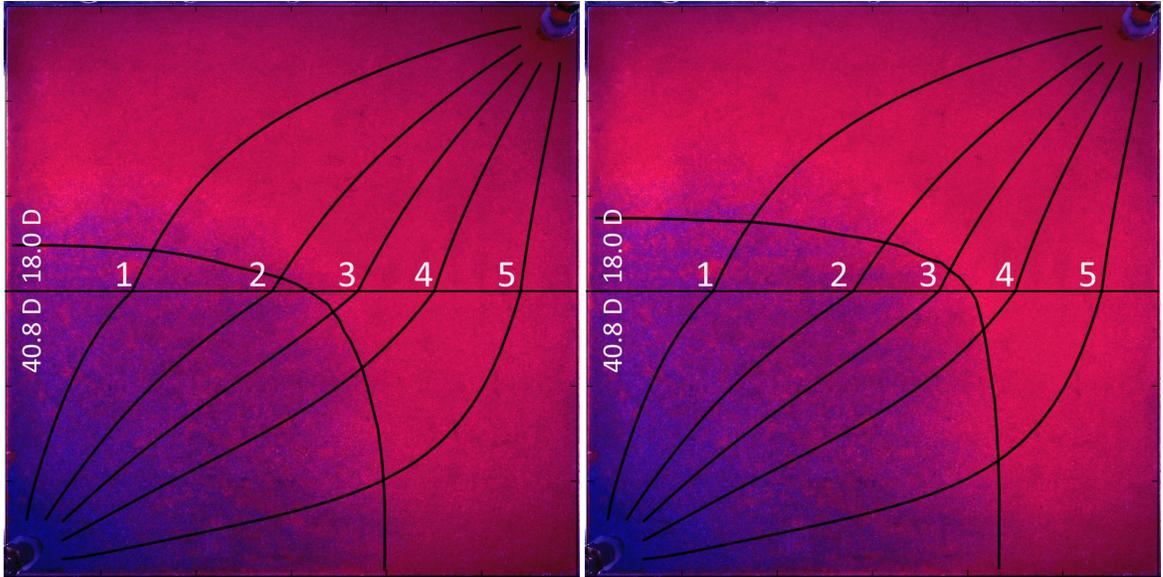
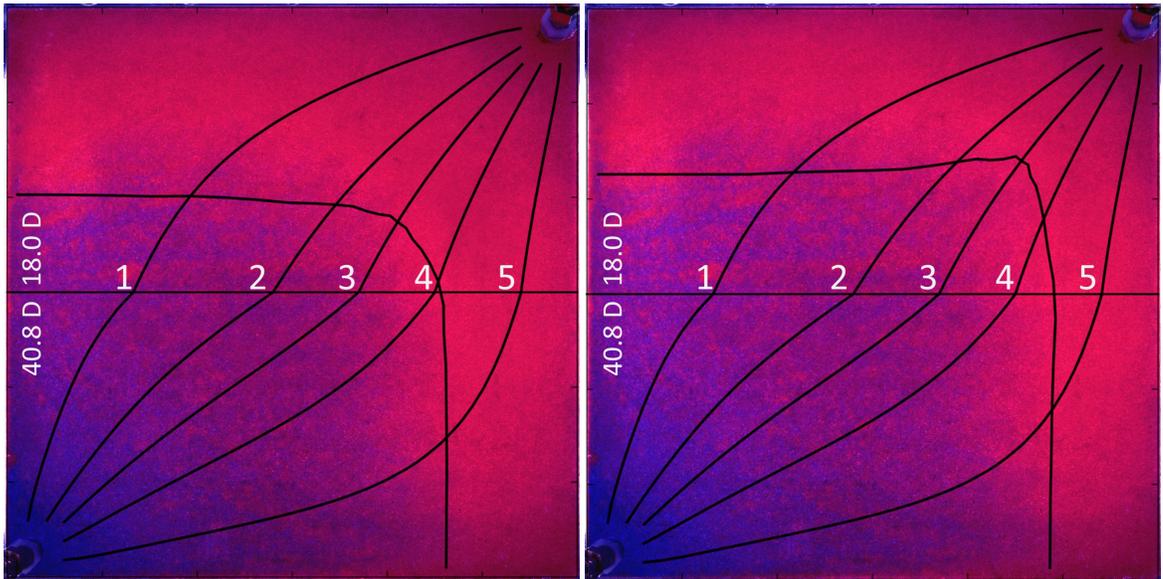


Figure 7.15: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4$ psi



(e) $t = 5 \text{ min}$

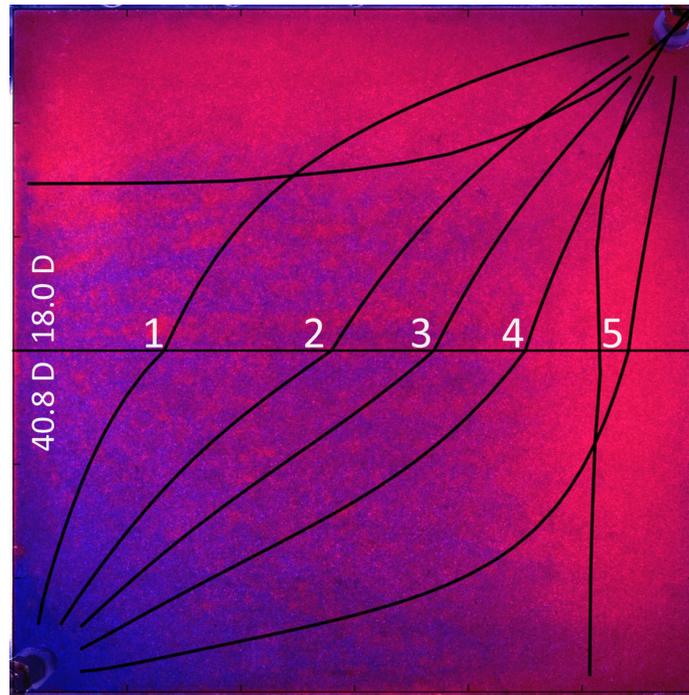
(f) $t = 6 \text{ min}$



(g) $t = 7 \text{ min}$

(h) $t = 8 \text{ min}$

Figure 7.15: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)



(i) $t = 9 \text{ min}$

Figure 7.15: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 4.4 \text{ psi}$ (cont.)

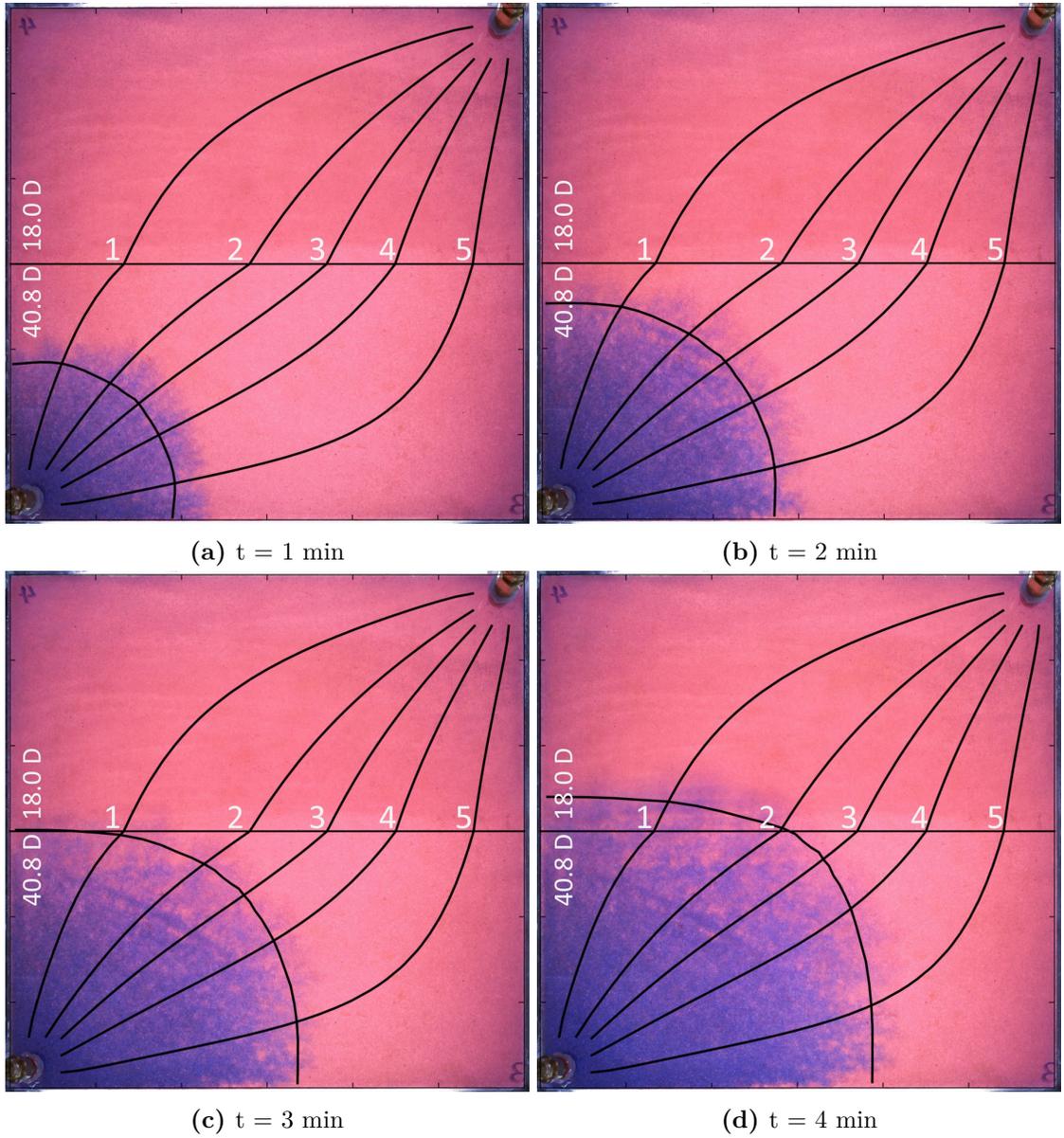
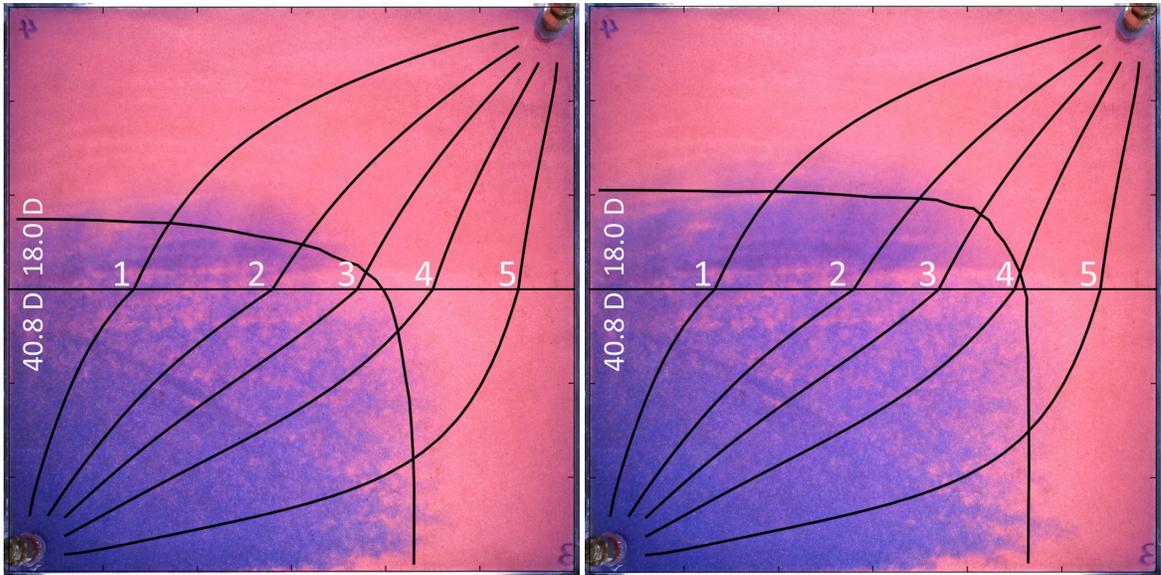
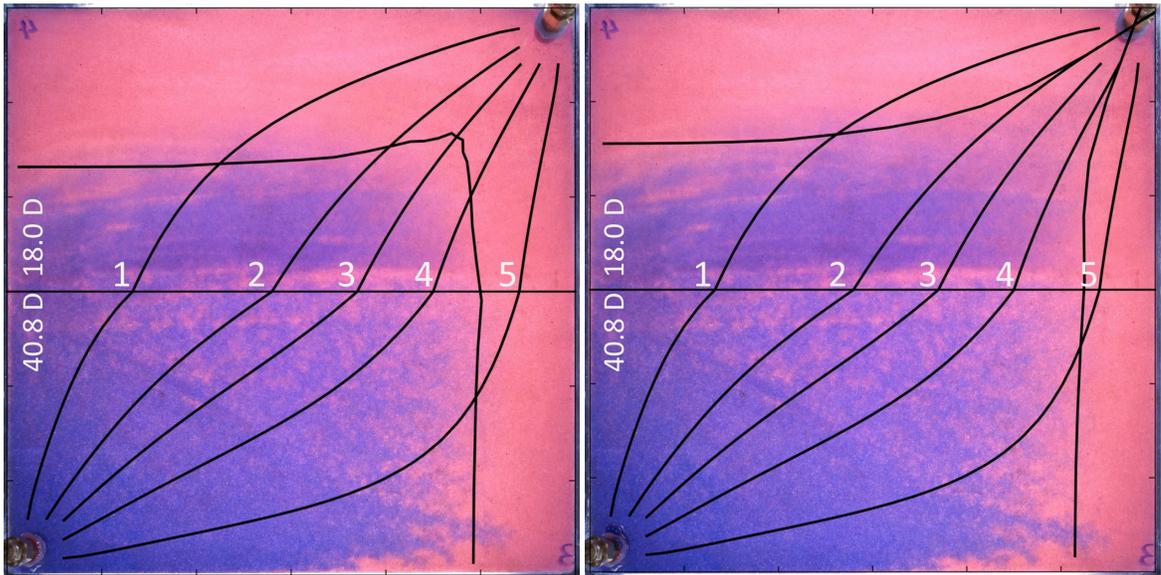


Figure 7.16: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$



(e) $t = 5 \text{ min}$

(f) $t = 6 \text{ min}$



(g) $t = 7 \text{ min}$

(h) $t = 8 \text{ min}$

Figure 7.16: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psi}$ (cont.)

7.4 Chapter Summary

In this research, two waterflooding visualization experiments are performed in the two-dimensional squared heterogeneous glass-bead unconsolidated macro-model. These visualization experiments are performed to mimic the real displacement process under constant differential pressure boundaries in an oil field for the first time.

The ability of the new streamline simulator to model physical problems is validated and demonstrated through history matching and direct simulation of the experimental displacement processes. The relative permeability relations between oil and water are determined by trial and error during the history matching process, then the relative permeabilities are input directly into the simulator to model the displacement process in the second experiment. The relative error between the observed and simulated results for both experiments is less than 6.5%, and the numerical simulations are proven to be successful.

Moreover, through analyzing the visualization results by determining the displacement fronts as a function of both time and space, the simulated displacement fronts can basically capture the overall shape of waterfronts, and the physical displacement processes are well understood.

Overall, through comparisons between the experimental and simulation results, the new streamline simulator is independently validated, the ability of the new streamline simulator to model physical problems is demonstrated, and the physical displacement processes under constant pressure boundaries in a heterogeneous porous medium is well understood.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

8.1.1 The New Streamline Tracing Methods

The new streamline tracing methods are developed in this research to give more accurate simulation results in determining pressure distribution, velocity field, streamline location and time-of-flight. The schemes of these new methods are based on piece-wise polynomial pressure distribution approximations by considering that pressure gradient is a main driving force for fluid flow in a porous medium, the pressure distribution approximations are important for approaching realistic velocity fields and accurate streamlines. Through approximating the pressure distribution at each grid block by a polynomial, the velocity field that directly defines streamlines can be determined. There are four requirements which the approximation polynomials must obey to give accurate solutions for a streamline:

1. The velocity approximation function must obey the velocity field divergence free condition;
2. The velocity approximation function is derived from the pressure function through

Darcy's Law;

3. The pressure approximation function must obey the Laplace equation; and
4. The pressure and velocity functions are given by explicit functions.

Based on these requirements, the Bilinear, Trilinear, and Cubic methods are developed and introduced.

The Bilinear and Trilinear methods

The Bilinear (for two-dimensional problems) and Trilinear (for three-dimensional problems) streamline tracing methods are developed by acknowledging the fact that the pressure distribution in a reservoir is always continuous. A full pressure continuity can be achieved everywhere in a reservoir by assuming pressures are continuous at sub-cell interfaces, and they vary linearly along sub-cell interfaces.

The Bilinear method is sufficiently accurate for pressure and velocity approximations. It is more accurate in approximations at inner grid blocks than boundary grid blocks. Compared to Pollock's method, the Bilinear method gives more accurate streamlines in inner grid blocks for all tested cases in this research, including the homogeneous, heterogeneous and anisotropic cases. This advantage is more obvious when a reservoir has heterogeneous properties. In some special cases, the heterogeneous reservoir will cause the streamlines to enter and exit in the same interface of a grid block. In this condition, only the Bilinear method can deliver accurate results. Moreover, the computational efficiency of the Bilinear method is high, being only slightly slower than Pollock's method. One limitation of the Bilinear method is that it may lead to errors for tracing streamlines in boundary grid blocks when the grid resolution is low.

The Trilinear method is also sufficiently accurate to trace streamlines in three-dimensional grid blocks. When the reservoir is relatively large, the Bilinear and Trilinear methods have a large potential to reduce the computational time by using coarse grid blocks in the inner

reservoir without losing much accuracy. This is achieved by using one coarse grid block where one bilinear/trilinear function can accurately approximate the pressure distribution in this region.

The Cubic method

The Cubic method assumes the pressure distributions are third-degree polynomials. It determines the velocity field by Darcy's Law using the velocity approximation function derived from the third-degree pressure functions. The Cubic method is adaptable to a first-order finite difference method by using the velocity interpretation method that interprets the normal velocity at grid block interfaces.

Compared to the Bilinear and Pollock methods, the Cubic method more closely approximates the analytical pressure and velocity distributions under the same grid resolution. This improvement for the velocity field approximation is sometimes crucial when the velocity field varies significantly in two or three coordinate directions. Additionally, it is also important for the Riemann approach along streamlines introduced in Chapter 6, since the cross sectional area of the streamtube is determined using the velocity.

Additionally, the Cubic method delivers more accurate streamlines than Pollock's method. This advantage is significant in all tested cases in this research, including homogeneous, heterogeneous and anisotropic cases. The trade off is that the computational cost is increased compared to Pollock's method. Nevertheless, high computational efficiency is still achieved.

The Cubic method is also very accurate to trace streamlines in three-dimensional grid blocks. The streamlines generated by the Cubic method can accurately illustrate the effects of different layers in three-dimensional reservoirs even with low grid resolutions.

One limitation of the Cubic method is that, the Pollock method gives better results when streamlines traveling very close to no flow boundaries. However, this disadvantage is insignificant for most cases, since only few streamlines traveling very close to boundaries.

Moreover, it becomes less important when the reservoir has heterogeneous properties or the grid resolution increases.

8.1.2 The Semi-analytical Riemann Solver

A semi-analytical Riemann solver along streamlines is introduced in this research thesis. This approach isolate the effects of geological heterogeneity in finite-difference grid blocks and the complex geometry of streamtubes from the fluid transport calculations. The effective cross-sectional area of a streamtube is approximated along its central streamline. This semi-analytical Riemann approach maps the analytical solutions along streamlines in terms of time-of-flight. The flow rate, saturation profile and corresponding time are determined with given cumulative water injection at each discretization stage. This semi-analytical Riemann solver achieve high efficiency and are orders of magnitude faster than conventional finite-difference simulation methods. It also extends the applications of Riemann approach along streamlines from constant flow rate boundaries to constant pressure boundaries.

8.1.3 The Three-dimensional New Streamline Simulator

The new three-dimensional streamline simulator for two phase flow introduced in this research thesis is a combination of the Cubic streamline tracing method and the semi-analytical Riemann solver. More specifically, to simulate a two-phase immiscible displacement process under constant pressure boundaries, this new streamline simulator has mainly three steps to accomplish:

1. The pressure equation is solved using the finite-difference method with given boundary conditions;
2. The streamlines are traced using the Cubic method assuming single phase flow;
3. The two-phase transport problem is solved along each streamline using the semi-

analytical Riemann approach.

The ability of this new streamline simulator is demonstrated through modeling different waterflooding displacement processes. It has been proven that the new streamline simulator can give sufficiently accurate results in modeling two- and three-dimensional homogeneous, heterogeneous, and anisotropic waterflooding problems. It gives more accurate simulation results than Eclipse100 when the number of streamlines is sufficiently large and the fixed streamline assumption is valid. This is because the grid orientation effects, the numerical dispersion for solving Riemann problems, and the well treatment errors in Eclipse100 have been significantly reduced or eliminated in the new streamline simulator. Moreover, the calculation speed of the new streamline simulator is much faster than Eclipse100. This advantage becomes more significant when more grid blocks are used in a reservoir model.

The limitations of this new streamline simulator are that it relies on the fixed streamline assumption; and it does not account for gravity and capillary effects. These limitations could be partly eliminated in the future by periodically updating streamlines; and using the concept of operator splitting to account for gravity and capillary effect.

8.1.4 Experimental and Numerical Studies of Waterflooding Experiments under Constant Differential Pressure Boundaries

In this research, the laboratory waterflooding visualization experiments are designed and performed to mimic real waterflooding processes under different constant differential pressure boundaries in a heterogeneous reservoir for the first time.

The ability of the new streamline simulator to model physical problems is validated and demonstrated through history matching and direct simulation of the two waterflooding experiments. The relative error between the observed and simulated results for both experiments are less than 6.5%, and the numerical simulations are proven to be successful.

The application of the new streamline simulator is also demonstrated through analyzing the visualization results. The waterfront movement and capillary number are successfully estimated as a function of both time and space. Overall, the simulated displacement fronts can basically capture the overall shape of waterfronts, and the physical displacement processes are well understood.

8.2 Future Work

The new streamline simulator is developed to deliver more accurate simulations for waterflooding problems. However, the applications of this new streamline simulator is still limited. These are potential areas of future research. To actually apply the new streamline simulator for engineering purposes, the streamline simulator needs to relax the assumptions made, and solve multi-phase multi-component transport problems. Several potential research directions are given below.

Couple a near well-bore streamline method

The new streamline simulator is a reservoir simulation tool, however, well conditions and near-well-bore regions are barely considered in this simulator. Since the pressure varies exponentially in the radial direction in a near well-bore region, the polynomials applied in this research cannot give close pressure approximations in this case. Therefore, when streamlines traveling very close to wells, the results may contain large errors.

This drawback can be potentially overcome by coupling a near-well-bore streamline method with the field streamline tracing method. Similar to the applications of hybrid grid blocks in reservoir simulations, a technique can be developed to improve the streamline tracing results by using Polar grid blocks in near-well-bore regions and Cartesian grid blocks elsewhere in the reservoir.

Update streamlines

One limitation of the new streamline simulator developed in this research is it relies on the fixed streamline assumption. When the mobility field varies significantly from its initial conditions, the updates of the pressure distribution, velocity field, and streamlines are necessary for obtaining accurate simulation results. One way to update streamlines is to map the saturation profile back to finite-difference grid blocks and then re-solve the pressure (Laplace) equation. Since large time steps can be taken to update streamlines, the calculation speed of the streamline simulator is still relatively fast compared to grid block based simulators.

In addition, since Eclipse100 is very robust in constructing geology models and solving pressure equation, the streamline simulator can be coupled with Eclipse100 in modeling large reservoir models. Communications between the streamline simulator and Eclipse100 could be improved to give more accurate simulation results and save computational time.

Solve multi-component problems

Multi-component problems are not investigated in this research. However, the analytical Riemann solutions for two-phase multi-component flow under constant pressure boundaries exist (Johansen and James, 2016). The semi-analytical Riemann approach developed in this research can be easily extended to solve these problems by mapping the component concentrations along streamlines. Numerical compositional solutions can also be mapped to streamlines in a similar way.

Model complex flow mechanisms

This new streamline simulator assumes that the fluid flows along streamlines. This assumption is no longer valid when the gravity and/or capillary effects are significant. The streamline results can be modified to incorporate capillary and gravity effects. A streamline simulator with capillary and gravity effects considered can also be implemented using

published methods. The governing equations should be modified to account for the gravity and the capillary forces. Modification of pressure equation changes the locations of the streamlines. Moreover, modification of saturation equation allows description of crossflow effects by means of an operator splitting technique. The similar technique can be applied in the new streamline simulator to model the fluid flow with complex mechanisms.

Bibliography

Aarnes, J.E., Gimse, T., and Lie, K.A. (2007). An introduction to the numerics of flow in porous media using Matlab. In *Geometric modelling, numerical simulation, and optimization* (pp. 265-306). Springer Berlin Heidelberg.

Agarwal, B., and Blunt, M.J. (2003). Streamline-based method with full-physics forward simulation for history-matching performance data of a North Sea field. *Society of Petroleum Engineers*, 8(02), 171-180.

Batycky, R.P. (1997). A three-dimensional two-phase field scale streamline simulator. Doctoral dissertation, Stanford University.

Batycky, R.P., Thiele, M.R., Baker, R.O., and Chugh, S. (2008). Revisiting reservoir flood-surveillance methods using streamlines. *SPE Reservoir Evaluation & Engineering*, 11(02), 387-394.

Bear, J. (1972). Dynamics of fluids in porous media. American Elsevier.

Berenblyum, R.A., Shapiro, A.A., Jessen, K., Stenby, E.H., and Orr Jr, F.M. (2003). Black oil streamline simulator with capillary effects. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.

Bommer, P.M., and Schechter, R.S. (1979). Mathematical modeling of in-situ uranium leaching. *Society of Petroleum Engineers*. 19(06), 393-400.

- Brand, C.W., Heinemann, J.E., and Aziz, K. (1991). The grid orientation effect in reservoir simulation. *SPE Symposium on Reservoir Simulation*. Society of Petroleum Engineers.
- Bratvedt, F., Bratvedt, K., Buchholz, C.F., Gimse, T., Holden, H., Holden, L., and Risebro, N.H. (1993). Frontline and Frontsim, two full scale, two-phase, black oil reservoir simulators based on front tracking. *Surveys on Mathematics for Industry*, 3, 185-215.
- Bratvedt, F., Gimse, T., and Tegnander, C. (1996). Streamline computations for porous media flow including gravity. *Transport in Porous Media*, 25(1), 63-78.
- Brock, D.C., and Orr Jr, F.M. (1991). Flow visualization of viscous fingering in heterogeneous porous media. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.
- Buckley, S.E., and Leverett, M. (1942). Mechanism of fluid displacement in sands. *Transactions of the AIME*, 146(01), 107-116.
- Caers, J., Krishnan, S., Wang, Y., and Kovscek, A.R. (2002). A geostatistical approach to streamline-based history matching. *Society of Petroleum Engineers*, 7(03), 250-266.
- Cense, A.W., and Berg, S. (2009). The viscous-capillary paradox in 2-phase flow in porous media. *International Symposium of the Society of Core Analysts held in Noordwijk, The Netherlands*, pp. 27-30.
- Christie, M.A., and Blunt, M.J. (2001). Tenth SPE comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Evaluation & Engineering*, 4(04), 308-317.
- Cinar, Y., Jessen, K., Berenblyum, R., Juanes, R., and Orr, F.M. (2006). An experimental and numerical investigation of crossflow effects in two-phase displacements. *Society of Petroleum Engineers*. 11(02), 216-226.
- Cordes, C., and Kinzelbach, W. (1992). Continuous groundwater velocity fields and path lines in linear, bilinear, and trilinear finite elements. *Water resources research*, 28(11),

2903-2911.

Corey, A.T. (1954). The interrelation between gas and oil relative permeabilities. *Producers monthly*, 19(1), 38-41.

Datta-Gupta, A., and King, M.J. (1995). A semianalytic approach to tracer flow modeling in heterogeneous permeable media. *Advances in Water Resources*, 18(1), 9-24.

Datta-Gupta, A., and King, M.J. (2007). Streamline simulation: theory and practice. Richardson, TX: Society of Petroleum Engineers.

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.

Dyes, A.B., Caudle, B.H., and Erickson, R.A. (1954). Oil production after breakthrough as influenced by mobility ratio. *Journal of Petroleum Technology*, 6(04), 27-32.

Elsayed, S.A., Baker, R., Churcher, P.L., and Edmunds, A.C. (1993). Multidisciplinary reservoir characterization and simulation study of the Weyburn unit. *Journal of Petroleum Technology*, 45(10), 930-973.

Ehlig-Economides, C.A. (1979). Well test analysis for wells produced at a constant pressure. Doctoral dissertation, Stanford University.

Fanchi, J.R. (1983). Multidimensional numerical dispersion. *Society of Petroleum Engineers Journal*, 23(01), 143-151.

Fay, C.H., and Prats, M. (1951). The application of numerical methods to cycling and flooding problems. *Proceedings of the 3rd World Petroleum Congress* (pp. 101-112).

Gautier, Y., Blunt, M.J., and Christie, M.A. (1999). Nested gridding and streamline-based simulation for fast reservoir performance prediction. *Computational Geosciences*, 3(3-4),

295-320.

Glimm, J., Isaacson, E., Marchesin, D., and McBryan, O. (1981). Front tracking for hyperbolic systems. *Advances in Applied Mathematics*, 2(1), 91-119.

Glimm, J., Lindquist, B., McBryan O., and Padmanabhan L. (1983). A front tracking reservoir simulator, five-spot validation studies and the water coning problem. *The Mathematics of Reservoir Simulation*, 1, 107-136.

Glimm, J., Grove, J.W., Li, X.L., Shyue, K M., Zeng, Y., and Zhang, Q. (1998). Three-dimensional front tracking. *SIAM Journal on Scientific Computing*, 19(3), 703-727.

Gunasekera, D., Cox, J., and Lindsey, P. (1997). The generation and application of K-orthogonal grid systems. In *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers.

Hægland, H., Dahle, H.K., Eigestad, G.T., Lie, K.A., and Aavatsmark, I. (2007). Improved streamlines and time-of-flight for streamline simulation on irregular grids. *Advances in Water Resources*, 30(4), 1027-1045.

Hauber, W.C. (1964). Prediction of waterflood performance for arbitrary well patterns and mobility ratios. *Journal of Petroleum Technology*, 16(01), 95-103.

Heinemann, Z.E., Brand, C.W., Munka, M., and Chen, Y.M. (1991). Modeling Reservoir Geometry With Irregular Grids. *SPE Reservoir Engineering*, 6(02), 225-232.

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(06), 679-683.

Higgins, R.V., and Leighton, A.J. (1962). Computer prediction of water drive of oil and gas mixtures through irregularly bounded porous media three-phase flow. *Journal of Petroleum Technology*, 14(09), 1-048.

- Higgins, R.V., Boley, D.W., and Leighton, A.J. (1964). Aids to forecasting the performance of water floods. *Journal of Petroleum Technology*, 16(09), 1-076.
- Huang, Y., Ringrose, P.S., and Sorbie, K.S. (1995). Capillary trapping mechanisms in water-wet laminated rocks. *SPE Reservoir Engineering*, 10(4).
- Lagrange, J.L. (1781). Mémoire sur la théorie du mouvement des fluides.
- Lake, L.W. (1989). Enhanced oil recovery. Prentice Hall.
- Lantz, R.B. (1971). Quantitative evaluation of numerical diffusion (truncation error). *Society of Petroleum Engineers Journal*, 11(03), 315-320.
- Ingebrigtsen, L., Bratvedt, F., and Berge, J. (1999). A streamline based approach to solution of three-phase flow. *SPE Reservoir Simulation Symposium*, pp. 253-260. James, L.A. (2012). Personal communication about experimental design.
- Johansen, T.E. (2010). A new semi-analytical method for streamline simulation. Unpublished manuscript, (www.petreng-thormod.ca).
- Johansen, T. E., and James, L. A. (2016). Solution of multi-component, two-phase Riemann problems with constant pressure boundaries. *Journal of Engineering Mathematics*, 96(1), 23-35.
- Johansen, T.E., and Liu, X. (2016). Solution of Riemann problems for hyperbolic systems of conservation laws modeling two phase flow in general streamtube geometries. Accepted by *Journal of engineering mathematics*.
- 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.
- Martin, J.C., and Wegner, R.E. (1979). Numerical solution of multiphase, two-dimensional incompressible flow using streamtube relationships. *Society of Petroleum Engineers*, 19(5),

313-323.

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.

Matringe, S.F., and Gerritsen, M.G. (2004). On accurate tracing of streamlines. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.

Matringe, S.F., Juanes, R., and Tchelepi, H.A. (2006). Robust streamline tracing for the simulation of porous media flow on general triangular and quadrilateral grids. *Journal of Computational Physics*, 219(2), 992-1012.

Muskat, M., and Wyckoff, R.D. (1934). A theoretical analysis of water-flooding networks. *Transactions of the AIME*, 107(01), 62-76.

Muskat, M., and Wyckoff, R.D. (1937). The flow of homogeneous fluids through porous media (Vol. 12). New York: McGraw-Hill.

Nédélec, J.C. (1980). Mixed finite elements in \mathbf{R}^3 . *Numerische Mathematik*, 35(3), 315-341.

Parsons, R.W. (1972). Directional permeability effects in developed and unconfined five-spots. *Journal of Petroleum Technology*, 24(04), 487-494.

Pedrosa Jr, O.A., and Aziz, K. (1986). Use of a hybrid grid in reservoir simulation. *SPE Reservoir Engineering*, 1(06), 611-621.

Pollock, D.W. (1988). Semianalytical computation of path lines for finite difference models. *Groundwater*, 26(6), 743-750.

Prats, M., Matthews, C.S., Jewett, R.L., and Baker, J.D. (1959). Prediction of injection rate and production history for multifluid five-spot floods. *Society of Petroleum Engineers*.

Prevost, M., Edwards, M.G., and Blunt, M.J. (2002). Streamline tracing on curvilinear

- structured and unstructured grids. *Society of Petroleum Engineers*, 7(02), 139-148.
- Roti, Ø., and Dawe, R.A. (1993). Modelling fluid flow in cross-bedded sections. *Transport in porous media*, 12(2), 143-159.
- Samier, P., Quettier, L., and Thiele, M. (2002). Applications of streamline simulations to Reservoir Studies. *SPE Reservoir Evaluation & Engineering*. 5(04), 324-332
- Sheldon, J.W., and Dougherty, E.W. (1964). A numerical method for computing the dynamical behavior of fluid-fluid interfaces in permeable media. *Society of Petroleum Engineers*, 4(02), 158-170.
- Sobolewski, M. (2005). Various methods of the measurement of the permeability coefficient in soils-possibilities and application. *Electronic Journal of Polish Agricultural Universities. Series Civil Engineering*, 8(2).
- Stan, C.A., Tang, S.K., and Whitesides, G.M. (2009). Independent control of drop size and velocity in microfluidic flow-focusing generators using variable temperature and flow rate. *Analytical chemistry*, 81(6), 2399-2402.
- Thiele, M.R. (1994). Modeling multiphase flow in heterogeneous media using streamtubes. Doctoral dissertation, Stanford University.
- Thiele, M.R., and Batycky, R.P. (2006). Using streamline-derived injection efficiencies for improved waterflood management. *SPE Reservoir Evaluation & Engineering*, 9(02), 187-196.
- Thiele, M.R., Batycky, R.P., and Fenwick, D.H. (2010). Streamline simulation for modern reservoir-engineering workflows. *Journal of Petroleum Technology*, 62(01), 64-70.
- Thiele, M.R., Batycky, R.P., Blunt, M.J., and Orr Jr, F.M. (1996). Simulating flow in heterogeneous systems using streamtubes and streamlines. *SPE Reservoir Engineering*, 11(1).

Thiele, M.R., Gerritsen, M.G., and Blunt, M.J. (2011). Streamline simulation. Society of Petroleum Engineers.

Wang, Y., and Kovscek, A.R. (2000). Streamline approach for history matching production data. *Society of Petroleum Engineers*, 5(04), 353-362.

Zhang, Y., King, M.J., and Datta-Gupta, A. (2012). Robust streamline tracing using intercell fluxes in locally refined and unstructured grids. *Water Resources Research*, 48(6).

Zhou, D., Fayers, F.J., and Orr Jr, F.M. (1997). Scaling of multiphase flow in simple heterogeneous porous media. *SPE Reservoir Engineering*, 12(03), 173-178.

Appendixes

Appendix A: Finite-Difference Method

The streamline tracing methods introduced in this research thesis are based on solving first for the pressure values at grid blocks and then approximating the pressure and velocity field. Here, the pressure equations for single phase flow used in the streamline tracing methods are solved using a finite difference method.

Recall the governing equations for the single phase fluid flow in the porous media with the following assumptions: the fluid and the porous media are incompressible, gravity and capillary effects are negligible, and flow is steady-state.

The mass conservation equation is,

$$\nabla \cdot (A\mathbf{u}) = q; \tag{A.1}$$

where, A is area; \mathbf{u} is the Darcy velocity; q is the volumetric outflow/inflow rate at the injector/producers.

The fluid flow in the porous media obeys Darcy's Law, *i.e.*,

$$\mathbf{u} = -\lambda\nabla P, \tag{A.2}$$

where \mathbf{u} is Darcy velocity, λ is the mobility and P is pressure.

Combing the mass conservation law (Eq. A.1) and Darcy's law (Eq. A.2) gives the Laplace

equation,

$$\nabla \cdot (A\lambda\nabla P) = q. \quad (\text{A.3})$$

This Laplace equation is the pressure equation we want to solve.

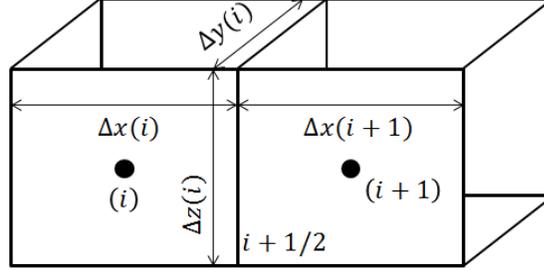


Figure A.1: An illustration of the finite difference method in grid block

As an example shown in Figure A.1, the gradient $\frac{\partial P}{\partial x}$, point out from the grid block (i) to (i+1) in the finite difference method is now replaced with,

$$\frac{\partial P}{\partial x} = \frac{2(P_{i+1} - P_i)}{\Delta x_i + \Delta x_{i+1}}. \quad (\text{A.4})$$

The λ on the interface (i+1/2) is defined by the harmonic mean,

$$\lambda_{i+1/2} = (\Delta x_i + \Delta x_{i+1}) \left(\frac{\Delta x_i}{\lambda_i} + \frac{\Delta x_{i+1}}{\lambda_{i+1}} \right)^{-1}. \quad (\text{A.5})$$

Hence, for orthogonal grid blocks, the out normal flux from the grid block (i) to (i+1) can be approximated by,

$$A_{i+1/2} u_{i+1/2} = 2\Delta y_i \Delta z_i \left(\frac{\Delta x_i}{\lambda_i} + \frac{\Delta x_{i+1}}{\lambda_{i+1}} \right)^{-1} (P_i - P_{i+1}); \quad (\text{A.6})$$

and the out normal velocity v from the grid block (i) to (i+1) is,

$$v_{i+1/2} = \frac{u_{i+1/2}}{\phi} = \frac{2(P_i - P_{i+1})}{\phi \left(\frac{\Delta x_i}{\lambda_i} + \frac{\Delta x_{i+1}}{\lambda_{i+1}} \right)}. \quad (\text{A.7})$$

Finally, summing over all interfaces to the particular grid block, we get an approximation to the $\nabla \cdot A\mathbf{v}$ in the mass conservation Eq. A.1.

To simplify the above expressions, we define the transmissibility in the interface (i+1/2),

$$T_{i+1/2} = 2\Delta y_i \Delta z_i \left(\frac{\Delta x_i}{\lambda_i} + \frac{\Delta x_{i+1}}{\lambda_{i+1}} \right)^{-1}. \quad (\text{A.8})$$

Thus by substituting the expression for the transmissibility into Eq.A.1, the grid block wise constant pressure $P_{i,j,k}$ satisfy the following equation,

$$\begin{aligned}
& T_{i+1/2,j,k} (P_{i,j,k} - P_{i+1,j,k}) + T_{i-1/2,j,k} (P_{i,j,k} - P_{i-1,j,k}) \dots \\
& + T_{i,j+1/2,k} (P_{i,j,k} - P_{i,j+1,k}) + T_{i,j-1/2,k} (P_{i,j,k} - P_{i,j-1,k}) \dots \\
& + T_{i,j,k+1/2} (P_{i,j,k} - P_{i,j,k+1}) + T_{i,j,k-1/2} (P_{i,j,k} - P_{i,j,k-1}) = q_{i,j,k}; \quad (\text{A.9})
\end{aligned}$$

where the k index is in the z coordinate direction, the j index is in the y coordinate direction, and the i index is in the x coordinate direction.

A published Matlab function for the implementation of Eq. A.9 on a uniform Cartesian grid (Aarnes *et al.*, 2007) is given below.

Code A-1: Finite difference method pressure solver

```

function [P,U,dp]=FD_pressure(Grid,Q,Mu,pi)
% Steady state incompressible single phase flow solver with given
% Grid block (Grid), permeability (Grid.K), fluid viscosity (Mu),
% constant well flow rate (Q), and initial reservoir pressure (pi).
% div (A u) = div[-AK/mu grad_p] = q
% SI Unit systems
% K: m^2; mu: Pa*s; P: Pa; q: m^3/s; U:m/s;

% Number of grid blocks and the grid block length
Nx=Grid.Nx; Ny=Grid.Ny; Nz=Grid.Nz; N=Nx*Ny*Nz;
hx=Grid.hx; hy=Grid.hy; hz=Grid.hz;
% Boundary conditions: five spot well parten with constant flow rate
q=zeros(N,1); q([1 N])=[Q -Q];
% Compute the transmissibilities TX, TY, TZ by harmonic averaging.
L = Grid.K.^(-1);
tx = 2*hy*hz/hx; TX = zeros(Nx+1,Ny,Nz);
ty = 2*hx*hz/hy; TY = zeros(Nx,Ny+1,Nz);
tz = 2*hx*hy/hz; TZ = zeros(Nx,Ny,Nz+1);
TX(2:Nx, :, :) = tx ./ (L(1,1:Nx-1, :, :) + L(1,2:Nx, :, :));
TY(:, 2:Ny, :) = ty ./ (L(2, :, 1: Ny-1, :) + L(2, :, 2:Ny, :));
TZ(:, :, 2: Nz) = tz ./ (L(3, :, :, 1: Nz-1) + L(3, :, :, 2:Nz));

```

```

TX=TX./(Mu);TY=TY./(Mu);TZ=TZ./(Mu);
% Sparce matrix of the transmissibilities
x1 = reshape(TX(1:Nx, :, :), N, 1); x2 = reshape(TX(2:Nx+1, :, :), N, 1);
y1 = reshape(TY(:, 1:Ny, :), N, 1); y2 = reshape(TY(:, 2:Ny+1, :), N, 1);
z1 = reshape(TZ(:, :, 1:Nz), N, 1); z2 = reshape(TZ(:, :, 2:Nz+1), N, 1);
DiagVecs = [-z2, -y2, -x2, x1+x2+y1+y2+z1+z2, -x1, -y1, -z1];
DiagIndx = [-Nx*Ny, -Nx, -1, 0, 1, Nx, Nx*Ny];
T = spdiags(DiagVecs, DiagIndx, N, N);
T(1, 1) = T(1, 1) + sum(Grid.K(:, 1, 1, 1));
% Solve linear system and extract interface fluxes.
p = T\q;
% Determine the pressure at grid blocks
P=reshape(p, Nx, Ny, Nz) - (max(p) + min(p)) ./ 2 + pi;
% Determine the pressure gradient at grid block interfaces
dp.x=zeros(Nx+1, Ny, Nz);
dp.y=zeros(Nx, Ny+1, Nz);
dp.x(2:Nx, :, :) = (P(2:Nx, :, :) - P(1:Nx-1, :, :)) ./ hx;
dp.y(:, 2:Ny, :) = (P(:, 2:Ny, :) - P(:, 1:Ny-1, :)) ./ hy;
% Determine the darcy velocity at grid block interfaces
U.x = zeros(Nx+1, Ny, Nz);
U.y = zeros(Nx, Ny+1, Nz);
U.x(2:Nx, :, :) = (P(1:Nx-1, :, :) - P(2:Nx, :, :)) .* TX(2:Nx, :, :). / (hy*hz);
U.y(:, 2:Ny, :) = (P(:, 1:Ny-1, :) - P(:, 2:Ny, :)) .* TY(:, 2:Ny, :). / (hx*hz);
if Nz >= 2,
    dp.z=zeros(Nx, Ny, Nz+1);
    dp.z(:, :, 2:Nz) = (P(:, :, 2:Nz) - P(:, :, 1:Nz-1)) ./ hz;
    U.z = zeros(Nx, Ny, Nz+1);
    U.z(:, :, 2:Nz) = (P(:, :, 1:Nz-1) - P(:, :, 2:Nz)) .* TZ(:, :, 2:Nz). / (hx*hy);
end

```

Appendix B: Buckley-Leverett Theory

B.1 Derivation of the fractional flow equation for a one-dimensional oil-water system

Consider a two-phase immiscible displacement of oil by water in a one-dimensional porous media system, as shown in Figure B.1. The fluid system has an inclination of θ with the horizontal plane. The fluids and rock are assumed to be incompressible.

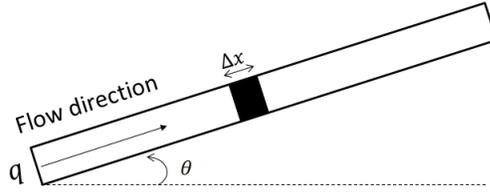


Figure B.1: One-dimensional flow system

The Darcy's law for this waterflooding problem is,

$$q_w = -A\lambda_w \left(\frac{\partial p_w}{\partial x} + \rho_w g \sin \theta \right); \quad (\text{B.1})$$

$$q_o = -\lambda_o A \left(\frac{\partial p_o}{\partial x} + \rho_o g \sin \theta \right); \quad (\text{B.2})$$

where, A is the cross sectional area of the flow system; λ_w and λ_o are the water and oil mobility ratio, respectively. The total flow rate q is given by,

$$q_t = q_o + q_w. \quad (\text{B.3})$$

The capillary pressure between oil and water is,

$$p_o - p_w = P_c; \quad (\text{B.4})$$

We can define the fractional flow function for water,

$$f = \frac{q_w}{q}. \quad (\text{B.5})$$

Substituting the above equations, the fractional flow function for water is obtained,

$$f = \frac{\lambda_w}{\lambda} - \frac{\lambda_o \lambda_w A}{q \lambda} \Delta \rho g \sin \theta + \frac{\lambda_w A}{q \lambda} \frac{dP_c(s_w)}{ds_w} \frac{\partial s_w}{\partial x}; \quad (\text{B.6})$$

where, $\lambda = \lambda_w + \lambda_o$ is the total mobility ratio; $\Delta \rho = \rho_w - \rho_o$ is the difference in water and oil density; s_w is the water saturation.

For the simplest case of horizontal flow, with negligible capillary pressure, the expression reduces to,

$$f = \frac{\lambda_w}{\lambda}. \quad (\text{B.7})$$

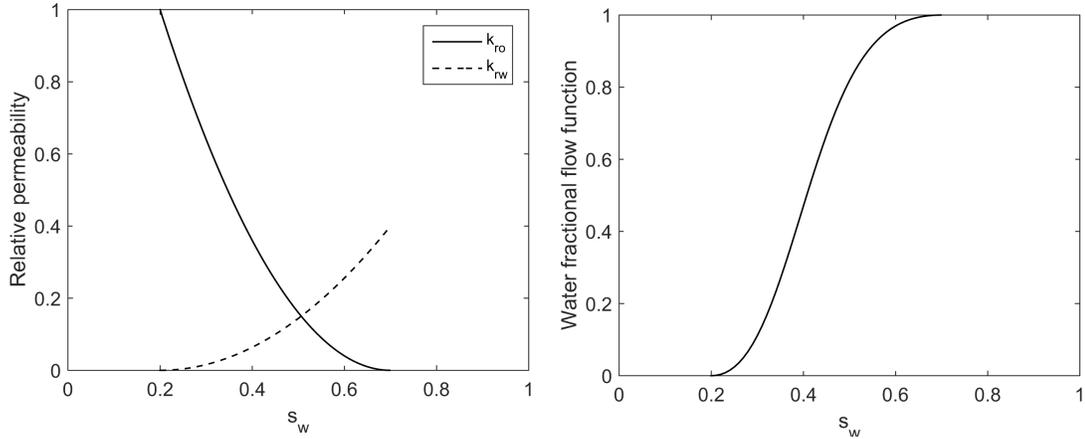


Figure B.2: The relative permeability plot and the fractional flow function

Typical plots of relative permeability and the corresponding fractional flow curve are given in Figure B.2.

B.2 Derivation of the Buckley-Leverett equation for a one-dimensional water-flooding system

In the following paragraphs, the Buckley-Leverett equation (1942),

$$\frac{\partial s_w}{\partial t} + \frac{q}{\phi A} \frac{\partial f}{\partial x} = 0; \quad (\text{B.8})$$

will be derived.

For a waterflooding process, the mass balance equation of water around a control volume of length Δx in the flow system shown in Figure B.1 for a time period of Δt can be written as,

$$[(q_w \rho_w)_x - (q_w \rho_w)_{x+\Delta x}] \Delta t = [(s_w \rho_w)^{t+\Delta t} - (s_w \rho_w)^t] A \Delta x \phi. \quad (\text{B.9})$$

When $\Delta x \rightarrow 0$ and $\Delta t \rightarrow 0$, above equation becomes the continuity equation,

$$A \phi \frac{\partial (s_w \rho_w)}{\partial t} + \frac{\partial (q_w \rho_w)}{\partial x} = 0. \quad (\text{B.10})$$

Since the fluids in the system are assumed to be incompressible, the ρ_w is constant, the above equation becomes,

$$A \phi \frac{\partial (s_w)}{\partial t} + \frac{\partial (q_w)}{\partial x} = 0. \quad (\text{B.11})$$

Substituting the fractional flow function (Eq. B.5) into the above equation yields,

$$\frac{\partial s_w}{\partial t} + \frac{q}{\phi A} \frac{\partial f}{\partial x} = 0; \quad (\text{B.12})$$

or,

$$\frac{\partial s_w}{\partial t} + \frac{q}{\phi A} \frac{\partial f}{\partial s_w} \frac{ds_w}{dx} = 0. \quad (\text{B.13})$$

B.3 Derivation of the fluid velocity

In the following paragraphs, the propagation velocity of the fluids with a certain saturation s_w expressing in,

$$\frac{dx}{dt} = \frac{q}{\phi A} \frac{df}{ds_w} \quad (\text{B.14})$$

will be derived.

Since the water saturation profile is a function of both location and time ($s_w(x, t)$), the total derivation of water saturation is,

$$ds_w = \frac{\partial s_w}{\partial x} dx + \frac{\partial s_w}{\partial t} dt. \quad (\text{B.15})$$

We can follow the propagation of fluids with a constant saturation s_w during the displacement, where the total derivation of the water saturation is zero,

$$ds_w = 0 = \frac{\partial s_w}{\partial x} dx + \frac{\partial s_w}{\partial t} dt. \quad (\text{B.16})$$

Substituting Eq. B.16 into the Buckley-Leverett equation Eq. B.13 , we get,

$$\frac{dx}{dt} = \frac{q}{\phi A} \frac{df}{ds_w} \quad (\text{B.17})$$

Integrating Eq. B.17 gives the expression for the position of the fluid front,

$$\int dx = \int \frac{q}{A\phi} \frac{df}{ds_w} dt; \quad (\text{B.18})$$

and

$$x = \frac{qt}{A\phi} \frac{df}{ds_w}; \quad (\text{B.19})$$

where x is the front location of the fluid.

B.4 The Buckley-Leverett solution

A typical plot of the fractional flow curve and its derivative is shown below:

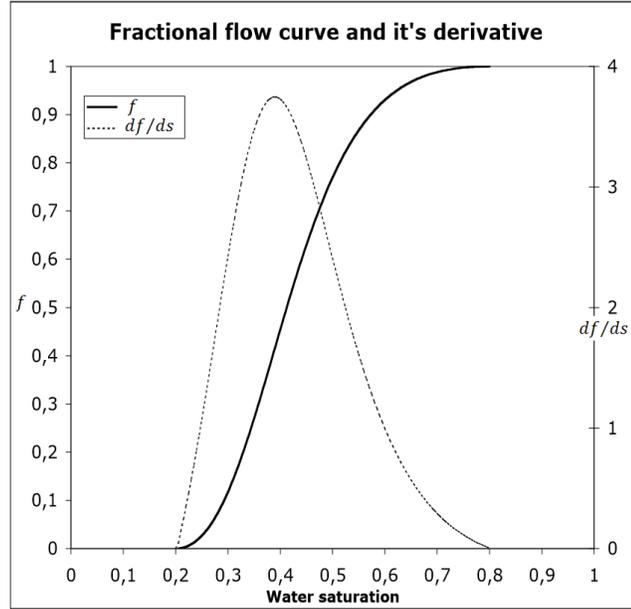


Figure B.3: An example of the fractional flow curve and its derivative

Using the expression for the front position Eq. B.19, and plotting the saturation profile we get the following results shown in Figure B.4 . As can be observed, the plot of saturation is showing an impossible physical situation, since there are two saturation values at the same location. Therefore, this is a result of the discontinuity in the saturation function, and the Buckley-Leverett solution to this problem is to modify the plot by defining a saturation discontinuity at the shock front and balancing of the areas ahead of the shock front and below the curve.

The determination of the water saturation at the shock front is shown graphically in Figure B.5. If we go from the water saturation at the inlet s_L to the water saturation at the outlet s_R , the propagation velocity for all waves composing the solution will have to increase monotonically, otherwise we will get an unphysical solution.

The final saturation profile with a shock front is shown in Figure B.6. Before the water

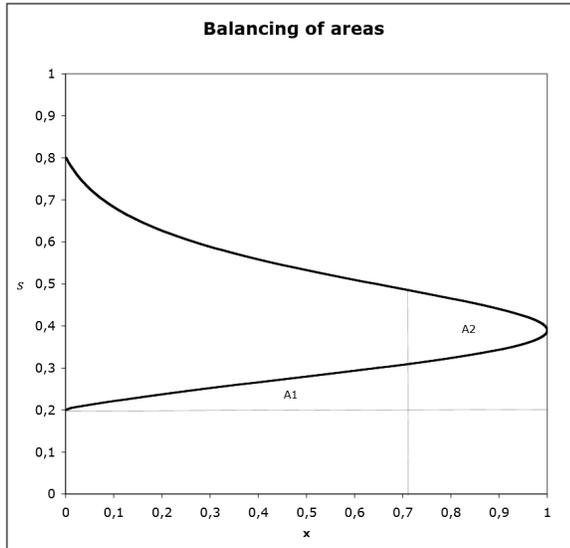


Figure B.4: The unphysical saturation profile

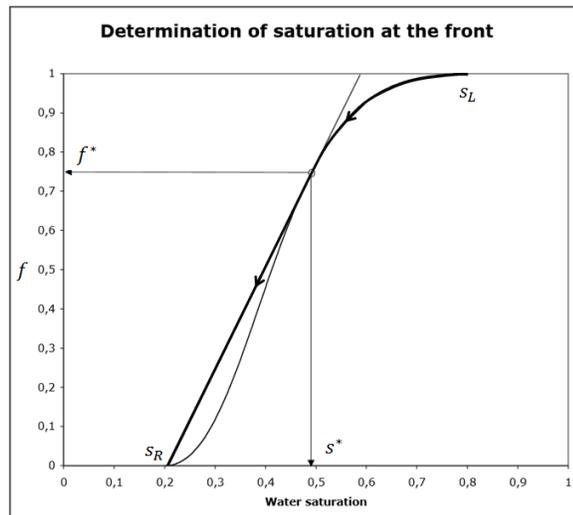


Figure B.5: The saturation at the shock front

breakthrough, this solution consists of a leading shock with shock saturation s^* , and a rarefaction wave connecting s^* to s_L .

B.5 A Riemann approach along streamline under a constant flow rate boundary

In the following paragraphs, the saturation profile of a Riemann problem under constant

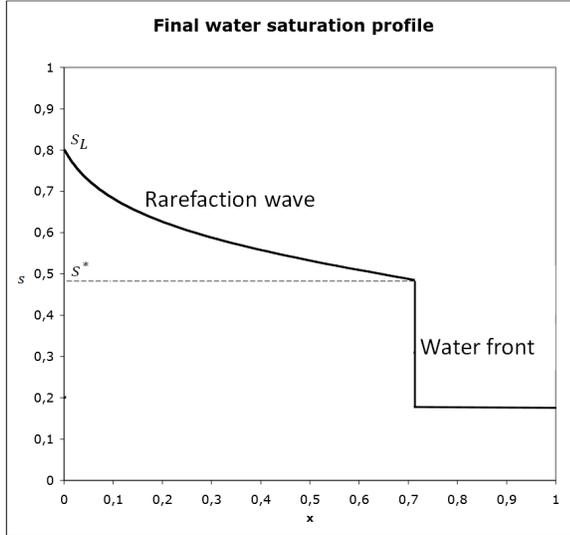


Figure B.6: The physical saturation profile

flow rate boundary along one-dimensional streamline/streamtube, *i.e.*,

$$f'(s) = \frac{\tau}{t} \quad (\text{B.20})$$

will be derived.

The rarefaction wave velocity Eq.B.17 is also valid in the streamline/streamtube. Integrating this equation gives,

$$\int_0^x \phi(x) A(x) dx = f'(s) q \int_0^t (t) dt. \quad (\text{B.21})$$

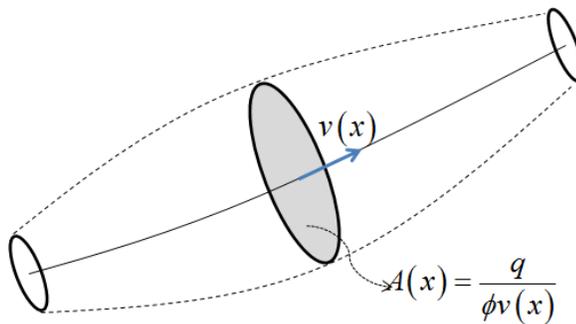


Figure B.7: The cross sectional area of a streamtube

As an example shown in Figure B.7, the Riemann problem along one dimensional stream-

line/streamtube has changing cross sectional area. The streamline method numerically determines $A(x)$ by the total flow rate q in the streamtube, and total velocity at the central streamline $v(x)$, *i.e.*

$$A(x) = \frac{q}{\phi v(x)}. \quad (\text{B.22})$$

Substituting the cross sectional area equation Eq. B.22 into the above equation yields,

$$q \int_0^x \frac{1}{v(x)} dx = f'(s) qt. \quad (\text{B.23})$$

Recall the mathematical definition of time-of-flight τ at the location x along the central streamline,

$$\tau(x) = \int_0^x \frac{1}{v(x)} dx. \quad (\text{B.24})$$

Substituting Eq. B.24 into Eq. B.23 gives,

$$q\tau(x) = f'(s) qt. \quad (\text{B.25})$$

Thus, the saturation profile along a streamline/streamtube under constant flow rate boundary is obtained,

$$f'(s) = \frac{\tau(x)}{t}. \quad (\text{B.26})$$

Appendix C: Runge-Kutta 4th Method

In this appendix, the Runge-Kutta 4th Method applied when generating streamlines in three-dimensional problems is introduced. This method is used to solve the two coupled first order differential equations.

The solutions to the two coupled 1st order differential equations:

$$\frac{dy}{dx} = F(x, y, z); \quad (\text{C.1})$$

$$\frac{dz}{dx} = G(x, y, z); \quad (\text{C.2})$$

can be obtained by the Runge-Kutta method, correct to fourth order terms in x , using,

$$K_1 = F(x_n, y_n, z_n)\Delta x; \quad (\text{C.3})$$

$$L_1 = G(x_n, y_n, z_n)\Delta x; \quad (\text{C.4})$$

$$K_2 = F\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_1}{2}, z_n + \frac{L_1}{2}\right)\Delta x; \quad (\text{C.5})$$

$$L_2 = G\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_1}{2}, z_n + \frac{L_1}{2}\right)\Delta x; \quad (\text{C.6})$$

$$K_3 = F\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_2}{2}, z_n + \frac{L_2}{2}\right)\Delta x; \quad (\text{C.7})$$

$$L_3 = G\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_2}{2}, z_n + \frac{L_2}{2}\right)\Delta x; \quad (\text{C.8})$$

$$K_4 = F\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_3}{2}, z_n + \frac{L_3}{2}\right)\Delta x; \quad (\text{C.9})$$

$$L_4 = G\left(x_n + \frac{\Delta x}{2}, y_n + \frac{K_3}{2}, z_n + \frac{L_3}{2}\right)\Delta x; \quad (\text{C.10})$$

where Δx is the incremental in x ; n is the current step, and $n + 1$ is the next step.

The incremental results are:

$$x_{n+1} = x_n + \Delta x; \quad (\text{C.11})$$

$$y_{n+1} = y_n + \frac{K_1 + 2K_2 + 2K_3 + K_4}{6}; \quad (\text{C.12})$$

$$z_{n+1} = z_n + \frac{L_1 + 2L_2 + 2L_3 + L_4}{6}. \quad (\text{C.13})$$

A Matlab function for implement the RK-4th method is given below.

Code C-1: RK4 solver

```

function [x,y,z]=RK4(F,G,x1,y1,z1,x2,m)
% Runge Kutta 4th Method for Solving Two Coupled First Order
% Differential Equations, dy/dx = F(x,y,z) and dz/dx = G(x,y,z).
% Using x as the parameter, and x is changing in a range of [x1,x2].
% Discrete the calculation range [x1,x2] into m intervals.
a=x1;b=x2;h = (b - a)/m;x = a:h:b;
% Define the length and the initial value for y and z.
y = zeros(1,length(x));z = zeros(1,length(x));y(1) = y1;z(1) = z1;
% Calculate the y and z at discrete intervals using the RK-4th method
for i=1:m,
    k1 = F(x(i),y(i),z(i))*h;
    l1 = G(x(i),y(i),z(i))*h;
    k2 = F(x(i)+0.5*h,y(i)+0.5*k1,z(i)+0.5*l1)*h;
    l2 = G(x(i)+0.5*h,y(i)+0.5*k1,z(i)+0.5*l1)*h;
    k3 = F(x(i)+0.5*h,y(i)+0.5*k2,z(i)+0.5*l2)*h;
    l3 = G(x(i)+0.5*h,y(i)+0.5*k2,z(i)+0.5*l2)*h;
    k4 = F(x(i)+h,y(i)+k3,z(i)+l3)*h;
    l4 = G(x(i)+h,y(i)+k3,z(i)+l3)*h;
    y(i+1) = y(i) + (1/6)*(k1+2*k2+2*k3+k4);
    z(i+1) = z(i) + (1/6)*(l1+2*l2+2*l3+l4);
end

```

Appendix D: Numerical Integration Method

The trapezoidal method approximates the integration over an interval by breaking the area down into trapezoids, and approximating the integration by piece-wise linear functions. An example is shown in Figure D.1.

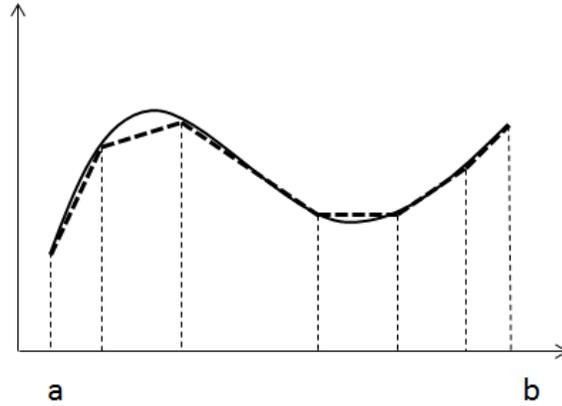


Figure D.1: Illustration of trapezoidal rule used on a sequence of samples

For an integration with $N + 1$ evenly spaced points, the approximation is,

$$\begin{aligned} \int_a^b f(x) &\approx \frac{b-a}{2N} \sum_{n=1}^N [f(x_n) + f(x_{n+1})] \\ &= \frac{b-a}{2N} [f(x_1) + 2f(x_2) + \dots + 2f(x_N) + f(x_{N+1})]; \end{aligned} \quad (\text{D.1})$$

where the spacing between each point is equal to $\frac{b-a}{N}$.

If the spacing between the points is not constant, then the formula generalizes to,

$$\int_a^b f(x) dx \approx \frac{1}{2} \sum_{n=1}^N (x_{n+1} - x_n) [f(x_n) + f(x_{n+1})]; \quad (\text{D.2})$$

where $x_{n+1} - x_n$ is the spacing between each consecutive pair of points.

Appendix E: Pollock's Method

In this section, the pressure and velocity approximation using Pollock's method in the two-dimensional is introduced. In addition, the Matlab code for the pressure, velocity approximations and streamline tracing using Pollock's method is given in this section.

To start, similar to the Bilinear and Cubic method, the grid blocks discretization and the pressure values at the primary nodes calculation are assumed to be already completed by applying the block-centered finite-difference method. An example of the pressure nodes and the regular grid blocks in the reservoir is given in Figure E.1.

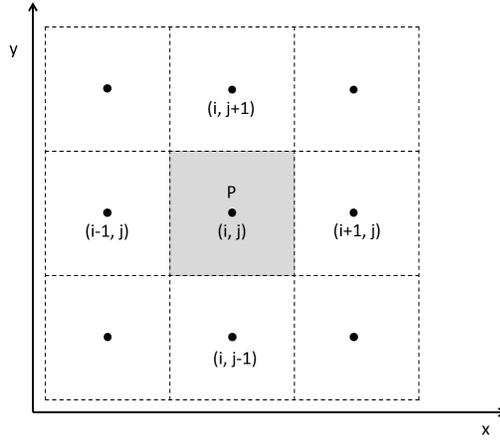


Figure E.1: An example of the pressure nodes and grid blocks in the two-dimensional reservoir

Recall the pressure and velocity approximation function applied in Pollock's method,

$$\begin{cases} P = P_0 + A_1x + A_2y + Bx^2 - Br_k y^2; \\ v_x = -\frac{K_x}{\phi\mu} (A_1 + 2Bx) = a_1 + bx; \\ v_y = -\frac{K_y}{\phi\mu} (A_2 - 2Br_k y) = a_2 - by; \end{cases} \quad (\text{E.1})$$

where, $r_k = \frac{k_x}{k_y}$; a_1, a_2, b are the velocity coefficients; and P_0, A_1, A_2, B are pressure coefficients.

The pressure and velocity field is approximated by determining the velocity coefficients first, and then determine the pressure coefficients through correlations. These velocity coefficients can be easily determined if the x - and y - directional velocities at the grid block interfaces are known.

Using the finite difference method, the Darcy velocity at the grid block interface $(i + 1/2)$ can be calculated using the pressure value at the primary nodes,

$$u_{i+1/2} = 2 \left(\frac{\Delta x_i}{\lambda_i} + \frac{\Delta x_{i+1}}{\lambda_{i+1}} \right)^{-1} (P_i - P_{i-1}); \quad (\text{E.2})$$

and therefore the velocity v ,

$$v = \frac{u}{\phi}. \quad (\text{E.3})$$

The results are shown in Figure E.2.

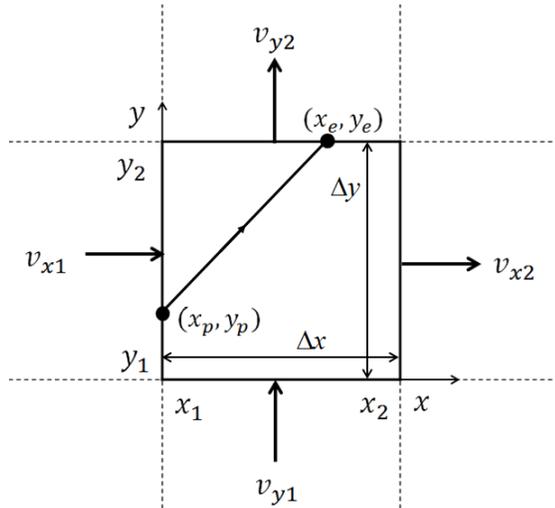


Figure E.2: The velocity at grid block interfaces

As an example shown in Figure E.2, the velocity coefficients can be determined by solving,

$$\begin{bmatrix} a_1 \\ a_2 \\ b \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_1 \\ 1 & 0 & x_2 \\ 0 & 1 & -y_1 \end{bmatrix}^{-1} \begin{bmatrix} v_{x1} \\ v_{x2} \\ v_{y1} \end{bmatrix}. \quad (\text{E.4})$$

Furthermore, the pressure coefficients can be determined using,

$$A_1 = -\frac{a_1\phi\mu}{K_x}; \quad A_2 = -\frac{a_2\phi\mu}{K_y}; \quad B = -\frac{b\phi\mu}{2K_x}; \quad (\text{E.5})$$

$$P_0 = P_m - A_1x_m - A_2x_y - Bx_m^2 + Br_kx_y^2; \quad (\text{E.6})$$

where, P_m is the pressure value at the grid block center, and x_m, y_m are its coordinates.

Code E-1: Determine the velocity coefficients in Pollock's method

```
function [a1,a2,b]=Velocity_coefficients_Pollock(Grid,V)
% Solve for the velocity coefficients (a1, a2, b) in Pollock's method
% with given velocities at grid block interfaces.
% vx = a1 + b*x; vy = a2 - b*y;
% Define the velocity at grid block interfaces.
vx1=V.x(1:end-1,1:end);vx2=V.x(2:1:end,1:1:end);
vy1=V.y(1:end,1:end-1);
% Define the size of the velocity coefficients
a1=zeros(Grid.Nx,Grid.Ny);a2=a1;b=a1;
% Calculate the velocity coefficients at each grid block by solving the
% linear equation
for i=1:Grid.Nx
    for j=1:Grid.Ny
        Xm=(i-0.5)*Grid.hx; Ym=(j-0.5)*Grid.hy;
        X1=Xm-Grid.hx/2; Vx1=vx1(i,j);
        X2=Xm+Grid.hx/2; Vx2=vx2(i,j);
        Y1=Ym-Grid.hy/2; Vy1=vy1(i,j);
        % %%% a1 a2 b
```

```

        Co=[1,    0,    X1;
            1,    0,    X2;
            0,    1,   -Y1];
        W=[Vx1;Vx2;Vy1];
        F=Co\W;
        a1(i,j)=F(1);    a2(i,j)=F(2);    b(i,j)=F(3);
    end
end

```

Code E-2: Approximate the pressure and velocity distribution using Pollock's method

```

clear all
% Approximate the velocity and pressure distribution in the reservoir
% Define the number of grid blocks and the grid block length
Grid.Nx=5; Grid.Ny=5; Grid.Nz=1; N=Grid.Nx*Grid.Ny*Grid.Nz;
Grid.hx=10; Grid.hy=10; Grid.hz=1;
% Define the permeability in grid blocks
Grid.K=ones(3,Grid.Nx,Grid.Ny,Grid.Nz);
% Define the constant flow rate Q; initial reservoir pressure Pi;
% fluid viscosity Mu; and porosity phi
Q=10;Pi=20;Mu=1;phi=0.3;
% Calculate the pressure and Darcy velocity using the finite difference
% method
[P,U]=FD_pressure(Grid,Q,Mu,pi);
% Calculate the true velocity
V.x=U.x./phi; V.y=U.y./phi;
% Calculate the velocity coefficients
[a1,a2,b]=Velocity_coefficients_Pollock(Grid,V);
% Calculate the Pressure coefficients
kx(:,:)=Grid.K(1, :, :, 1); ky(:,:)=Grid.K(2, :, :, 1); M=kx./ky;
A1=-a1.*phi.*Mu./kx; A2=-a2.*phi.*Mu./ky; B=-b.*phi.*Mu./2./kx;
xm=zeros(Grid.Nx,Grid.Ny); ym=xm;
for i=1:Grid.Nx
    for j=1:Grid.Ny
        xm(i,j)=(i-0.5)*Grid.hx; ym(i,j)=(j-0.5)*Grid.hy;
    end
end

```

```

    end
end
P0=P-A1.*xm-A2.*ym-B.*xm.^2+B.*M.*ym.^2;
% Appromiate the velocity and pressure distribution in the entire reservoir
% expect the injector and pressure grid blocks
% vx = a1 + b*x; vy = a2 - b*y;
% p = p0 + A1*x + A2*y + B*x^2 - B*M*y^2;
vx=zeros(50,50); vy=zeros(50,50); p=zeros(50,50);
for xx=1:50
    for yy=1:50
        x=xx-.5;y=yy-.5;
        NBx=fix(x./Grid.hx)+1;NBy=fix(y./Grid.hy)+1;
        a1_=a1(NBx,NBy);a2_=a2(NBx,NBy);b_=b(NBx,NBy);
        A1_=A1(NBx,NBy);A2_=A2(NBx,NBy);B_=B(NBx,NBy);
        M_=M(NBx,NBy);p0=P0(NBx,NBy);
        vx(xx,yy)=a1_+b_*x; vy(xx,yy)=a2_-b_*y;
        p(xx,yy)=p0+A1_*x+A2_*y+B_*x^2-B_*M_*y^2;
    end
end
end
p(1:10,1:10)=0;p(41:50,41:50)=0;
vx(1:10,1:10)=0;vx(41:50,41:50)=0;
vy(1:10,1:10)=0;vy(41:50,41:50)=0;

```

Code E-3: Streamline tracing using Pollock's method

```

clear all
% Tracing streamlines using Pollock's method
% Define the number of grid blocks and the grid block length
Grid.Nx=10; Grid.Ny=10; Grid.Nz=10; N=Grid.Nx*Grid.Ny*Grid.Nz;
Grid.hx=100./(Grid.Nx);Grid.hy=100./(Grid.Ny); Grid.hz=100./(Grid.Ny);
Grid.K=ones(3,Grid.Nx,Grid.Ny,Grid.Nz);
Q=10;Pi=20;Mu=1;phi=0.3;
% Calcualte the Pressure and Darcy velocity using finite difference method
[P,U]=FD_pressure(Grid,Q,Mu,pi);
% Calcualte the ture velocity
V.x=U.x./phi; V.y=U.y./phi; V.z=U.z./phi;

```

```

% Calculte the velocity coefficient/gradient
a.x(1:Grid.Nx, :, :)=(V.x(2:Grid.Nx+1, :, :)-V.x(1:Grid.Nx, :, :))./Grid.hx;
a.y(:, 1:Grid.Ny, :)=(V.y(:, 2:Grid.Ny+1, :)-V.y(:, 1:Grid.Ny, :))./Grid.hy;
a.z(:, :, 1:Grid.Nz)=(V.z(:, :, 2:Grid.Nz+1)-V.z(:, :, 1:Grid.Nz))./Grid.hz;
% Define the streamlines launching point
LX=[2,4,6,8,10,10,10,10];LY=[10,10,10,10,2,4,6,8];LZ=[5,5,5,5,5,5,5,5];
l=numel(LX);
nu=1;nn=10;
x1=LX(nu);y1=LY(nu);z1=LZ(nu);
% Define the size and the inital value of the time-of-flight
tof=zeros(1,1);
while (nu<=l)
% Locate the launching point into grid blocks
% roundn(x,nn) returns x rounded to nn digits.
NBx=fix((roundn(x1/Grid.hx,nn)))+1; NBy=fix((roundn(y1/Grid.hy,nn)))+1;
NBz=fix((roundn(z1/Grid.hz,nn)))+1;
j=1;
XX(nu,j)=x1; YY(nu,j)=y1; ZZ(nu,j)=z1;
while (NBx<=(Grid.Nx-1) || NBy<=(Grid.Ny-1) || NBz<=(Grid.Nz-1));
    x=x1-(NBx-1)*Grid.hx; y=y1-(NBy-1)*Grid.hy; z=z1-(NBz-1)*Grid.hz;
    ax=a.x(NBx,NBy,NBz); ay=a.y(NBx,NBy,NBz); az=a.z(NBx,NBy,NBz);
% Determine the normal velocity at grid block interfaces
    vx1=V.x(NBx,NBy,NBz); vy1=V.y(NBx,NBy,NBz); vz1=V.z(NBx,NBy,NBz);
    vx2=ax*Grid.hx+vx1; vy2=ay*Grid.hy+vy1; vz2=az*Grid.hz+vz1;
% Calculate the time-of-flight spent for the partical travel to each
% grid block interface
    [dtx1,dtx2]=time_of_flight_pollock(x,Grid.hx,ax,vx1,vx2);
    [dty1,dty2]=time_of_flight_pollock(y,Grid.hy,ay,vy1,vy2);
    [dtz1,dtz2]=time_of_flight_pollock(z,Grid.hz,az,vz1,vz2);
% Determine the true time-of-flihgt
    tt=[dtx1 dtx2 dty1 dty2 dtz1 dtz2];
    dte=min(tt(tt>0));
    tof(nu)=tof(nu)+dte;
% Determine the true exit point
    if dte==dtx2 || dte==dtx1

```

```

        [xe, ye, ze]=exit_pollock(dte, Grid.hx, dtx2, ay, y, vy1, az, z, vz1);
    end
    if dte==dty2 || dte==dty1
        [ye, xe, ze]=exit_pollock(dte, Grid.hy, dty2, ax, x, vx1, az, z, vz1);
    end
    if dte==dtz2 || dte==dtz1
        [ze, ye, xe]=exit_pollock(dte, Grid.hz, dtz2, ay, y, vy1, ax, x, vx1);
    end
    x2=(NBx-1)*Grid.hx+xe; y2=(NBy-1)*Grid.hy+ye; z2=(NBz-1)*Grid.hz+ze;
    XX(nu, j+1)=x2; YY(nu, j+1)=y2; ZZ(nu, j+1)=z2;
    % Use the exit point as the entry point for the next grid block
    NBx=fix((roundn(x2/Grid.hx, nn)))+1;
    NBy=fix((roundn(y2/Grid.hy, nn)))+1;
    NBz=fix((roundn(z2/Grid.hz, nn)))+1;
    x1=x2; y1=y2; z1=z2;
    j=j+1;
end
plot3(XX(nu, 1:j), YY(nu, 1:j), ZZ(nu, 1:j), 'black:', 'LineWidth', 1.5)
    hold on
    nu=nu+1;
    if nu<=1
        y1=LX(nu); x1=LY(nu); z1=LZ(nu);
    end
end
end

```

Code E-4: Determine the time-of-flight spend for the particle travel to the grid block interfaces

```

function [dt1, dt2]=time_of_flight_pollock(x, hx, ax, vx1, vx2)
% Calculate the time-of-flight spent for the particle travel to the
% interfaces 1 and 2.
% x the initial local coordinate of the particle;
% hx the length of the grid block;
% ax the velocity gradient in the grid block;
% vx1 the velocity at interface 1; vx2 the velocity at interface 2;
% dt1 the time-of-flight spent for the particle travel to the interfaces 1;

```

```

% dt2 the time-of-flight spent for the particle travel to the interfaces 2;
error=1e-10;
% The initial velocity of the particle
Vxp1=ax*x+vx1;
if ax^2-error<0
    dt2=(hx-x)/Vxp1;
    if Vxp1<0
        dt1=(-x)/Vxp1;
    else
        dt1=0;
    end
else
    dt2=(1/ax)*log(vx2/Vxp1);
    if ((Vxp1<vx1)*ax)>0
        dt1=(1/ax)*log(vx1/Vxp1);
    else
        dt1=0;
    end
end

```

Code E-5: Determine the exit point

```

function [xe,ye,ze]=exit_pollock(dte,hx,dtx2,ay,y,vy1,az,z,vz1)
% Calculate the true exit point using the Pollock's method
% dte is the true time-of-flight, and the particle exits at [0] or [hx];
% hx is the directional-length of the grid block;
% dtx2 is the time-of-flight spend for the particle exits at [hx];
% ay/az is the velocity gradient in other direction.
% vy1/vz1 is the normal velocity in interface 1.
% y/z is the initial local coordinate of the particle.
% Initial velocity at entry point.
Vyp1=ay*y+vy1; Vzpl=az*z+vz1;
error=1e-10;
if dte==dtx2
    xe=hx;
else

```

```

    xe=0;
end
if ay^2-error<0
    ye=y+vy1*dte;
else
    ye=(1/ay)*(Vyp1*exp(ay*dte)-vy1);
end
if az^2-error<0
    ze=z+vz1*dte;
else
    ze=(1/az)*(Vzp1*exp(az*dte)-vz1);
end

```

Appendix F: Macro-model Fabrication

The unconsolidated Glass-bead pack is applied as the heterogeneous porous medium for the waterflooding visualization experiment. This appendix introduces the procedures for fabricate the unconsolidated heterogeneous glass-bead pack macro-model. This model is designed by James (2013), and fabricated by the author of this thesis.

The procedures are given in below, Caution: The glass beads can cause health hazards by eye contact, skin contact, inhalation and ingestion. Lab coat, lab safety goggle, dust mask, and lab gloves are required when handling it.

1. Fabricate the two parts of the plexiglass box as shown in Figure F.1;
2. Drill two holes at the plexiglass plate at diagonal corners. These two holes are function as injector and producer for the model.
3. Glue a thin layer of glass beads at inner surface of the plexiglass box to prevent the fluid slippage at the surface.
4. Screw the two piece of plexiglass plates together to form a box;

5. Seal one hole and leave another hole open to fill the box with glass beads;
6. Drop the small sized glass beads (BT-4) from one hole to fill half the box, inject water to aid the compaction;
7. Drop the uniform larger sized glass beads (BT-3) from one hole to fill the box, inject water to aid the compaction;
8. Seal both holes and attach the model to vibrator;
9. If the glass beads can move in the model, open one hole to drop more glass beads;
10. Repeat steps 8 to 9 until the plexiglass box is filled with glass beads and water.

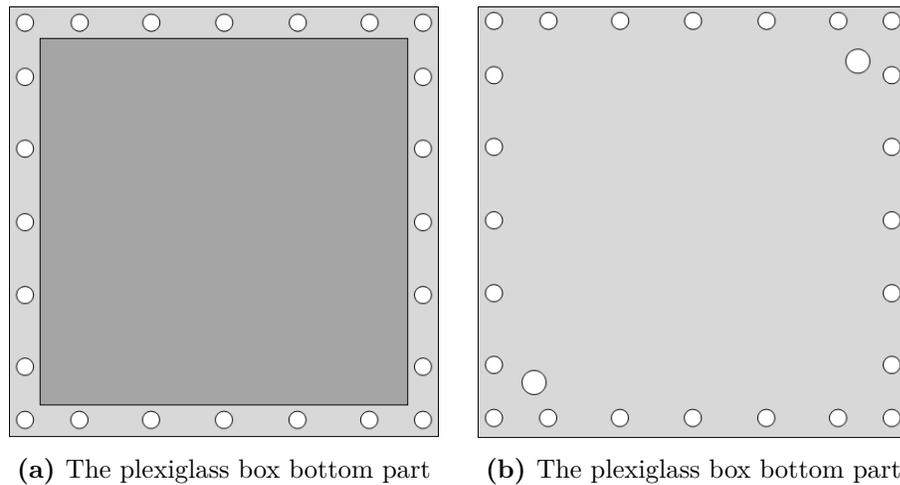


Figure F.1: The two parts for plexiglass box

Appendix G: Absolute Permeability Measurement

The permeability of the BT-3 and BT-4 glass bead packs are measured using the falling head test.

The falling head permeability test involves flow of water through the glass-bead pack in a cylinder tube which provides the water head and also allows measuring the volume of water

passing through the sample. The test starts by allowing water to flow through the glass-bead pack and ends before the water level drops to the height of glass-bead pack. The time required for the water drop from the upper to the lower level is recorded. The permeability measurement setup is shown in Fig.G.1 . The measurement results are provided in the end.

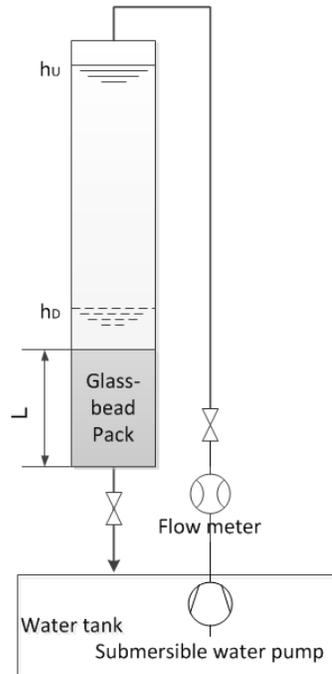


Figure G.1: Permeability measurement experiment setup

The specific permeability measurement procedures are,

1. Compact the glass beads in the lower part of the cylinder tube;
2. Fill the cylinder with water and then close the water inlet valve;
3. Measure and record the length of the glass-bead pack L ;
4. Measure and record the upper water level h_U ;
5. Set a lower water level h_D ;
6. Open the water outlet valve at the cylinder bottom and start record the time simultaneously;

7. Record the time interval Δt when the water level dropped to h_D ;
8. Calculate the permeability using (ASTM D2435),

$$K = \frac{L}{\Delta t} \frac{\mu}{\rho g} \log \frac{h_U}{h_D}. \quad (\text{G.1})$$

9. Refill the cylinder and repeat the test at least three times with different water levels;
10. Calculate the mean and standard deviation.

The water viscosity is 1.5 cP (at 5C) is The measurement results are shown in the Table G.1 and G.2.

Table G.1: Absolute permeability measurement results for BT-3 glass beads

Run order	L (cm)	h_U (cm)	h_D (cm)	Δt (s)	K (Darcy)
1	14.9	51.4	22.6	195.27	42.2
2	14.9	57.0	22.0	240.13	39.8
3	14.9	47.3	23.2	180.28	39.7
4	14.9	37.9	23.0	120.39	41.6

K Mean = 40.8 D, Std.Dev. = 0.6

Table G.2: Absolute permeability measurement results for BT-4 glass beads

Run order	L (cm)	h_U (cm)	h_D (cm)	Δt (s)	K (Darcy)
1	8.4	56.9	16.8	360.15	19.2
2	8.4	55.9	22.2	270.46	19.3
3	8.4	57.6	40.3	120.44	16.7
4	8.4	40.3	21.8	180.52	19.2
5	8.4	57.9	33.6	180.30	17.1
6	8.4	33.6	21.7	150.15	16.4

K Mean = 18.0 D, Std.Dev. = 0.5

Appendix H: Porosity Measurement

The porosity of the BT-3 and BT-4 glass bead packs are measured using a fluid saturation method. The porosity measurement setup is shown as Fig.H.1. The measurement results are provided in the end.

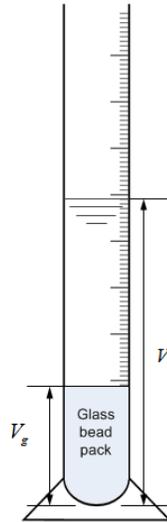


Figure H.1: Porosity measurement experiment setup

The specific porosity measurement procedures are,

1. Empty and dry the 10 *ml* graduated cylinder;
2. Drop some glass beads (BT-3 or BT-4) into the graduated cylinder;
3. Place the graduated cylinder at the mass balance and then set the balance to zero;
4. Fill water into the graduated cylinder to emerge the glass-bead pack;
5. Measure and record the mass of the water in the graduated cylinder M_w ;
6. Read and record the volume of the glass-bead pack V_g and the total volume of the mixture in the graduated cylinder V_t ;

7. Calculate porosity using,

$$\phi = \frac{V_p}{V_g} = \frac{\frac{M_w}{\rho} - (V_t - V_g)}{V_g}. \quad (\text{H.1})$$

8. Empty and dry the graduated cylinder for another porosity test;

9. Repeat the test at least three times;

10. Calculate the mean and standard deviation.

The measurement results are shown in Table H.1 and H.2.

Table H.1: Porosity measurement results for BT-3 glass beads

Run order	M_w (g)	V_t (ml)	V_g (ml)	ϕ
1	5.49	6.6	2.0	0.44
2	6.82	7.6	1.4	0.45
3	6.37	7.4	1.8	0.43
4	6.89	7.5	1.1	0.45
ϕ Mean = 0.44, Std.Dev. = 0.01				

Table H.2: Porosity measurement results for BT-4 glass beads

Run order	M_w (g)	V_t (ml)	V_g (ml)	ϕ
1	7.35	8.2	1.5	0.43
2	6.08	6.9	1.5	0.45
3	6.84	7.5	1.2	0.44
4	6.43	7.0	1.0	0.43
ϕ Mean = 0.44, Std.Dev. = 0.01				

Appendix I: Capillary Number Determination

Recall that the capillary number is defined as,

$$Ca = \frac{\mu v}{\gamma}; \quad (\text{I.1})$$

where, $\mu = 1.0 \text{ cP}$ is the water viscosity, v is the waterfront velocity, and $\gamma = 0.049 \text{ N/m}$ is the interfacial tension between silicone oil and water (Stan *et.al*, 2009).

The only unknown parameter in Eq.I.1 is the waterfront velocity, which can be determined using the experimental visualization results. The waterfront movement is along a certain streamline, and its average velocity v can be determined by,

$$v = \frac{\Delta L}{\Delta t}; \quad (\text{I.2})$$

where, Δt is the time duration; and ΔL is the travel distance along streamline. The travel distance can be calculated using $\Delta L = \sqrt{\Delta x^2 + \Delta y^2}$, where Δx and Δy are the incremental coordinate of the waterfront location along the streamline.

As an example given in Figure I.1, the incremental coordinate $(\Delta x, \Delta y)$, the travel distance ΔL , the time duration Δt , the average velocity v , and the Capillary number Ca at the waterfront along streamlines 1 and 3 are reported in Table I.1.

Table I.1: Experimental results for capillary number at waterfront (2.5 min, $\Delta P = 5.2 \text{ psig}$)

Streamline No.	$\Delta x \text{ (cm)}$	$\Delta y \text{ (cm)}$	$\Delta L \text{ (cm)}$	$\Delta t \text{ (min)}$	$v \text{ (cm/min)}$	$Ca \text{ (}\times 10^{-5}\text{)}$
1	1.5	2.0	2.5	1.0	2.5	0.85
3	2.5	2.0	3.2	1.0	3.2	1.09

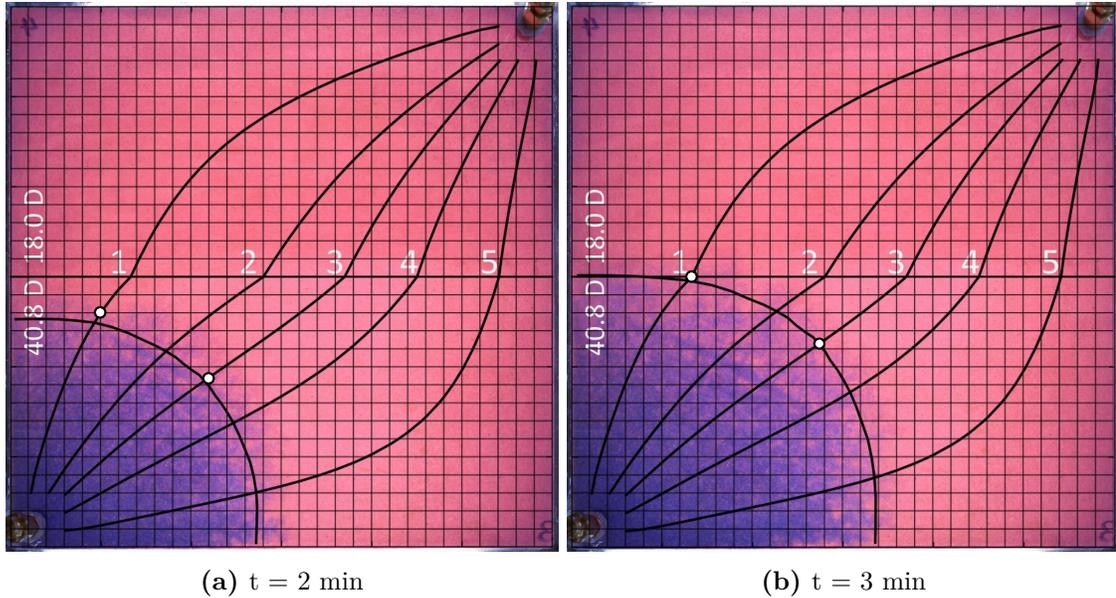


Figure I.1: Observed and simulated waterfront movement before water breakthrough, $\Delta P = 5.2 \text{ psig}$

Appendix J: Matlab Code

J.1 The Bilinear streamline tracing method

Code J.1-1: Streamline tracing using the Bilinear method

```

clear all
% Tracing streamlines using the Bilinear method
% Define the number of grid blocks and the grid block length
Grid.Nx=100; Grid.Ny=Grid.Nx; Grid.Nz=1;
Grid.hx=100./(Grid.Nx); Grid.hy=100./(Grid.Ny); Grid.hz=1;
Grid.K=ones(3,Grid.Nx,Grid.Ny,Grid.Nz);
Q=10;Pi=20;Mu=1;phi=0.3;
% Calcualte the Pressure using finite difference method
[P]=FD_pressure(Grid,Q,Mu,Pi);
% Define dual-cell system and calcualte the velocity coefficient
% at sub-cells
[a1_,a2_,b1_,b2_]=v_coe_Bilinear(Grid,P,Mu,phi);
Grid.Nx=2*(Grid.Nx); Grid.Ny=2*(Grid.Ny);
Grid.hx=Grid.hx./2; Grid.hy=Grid.hy./2; bone=Grid.Nx/10;
% Define the streamlines launching points

```

```

LX=[10,13,17,23,31,44,61,Grid.hx.*bone.*ones(1,6)];
LY=[Grid.hy.*bone.*ones(1,7),13,17,23,31,44,61];
l=numel(LX);nu=1;x1=LX(nu); y1=LY(nu);
% Define the size and the initial value of the time-of-flight
tof=zeros(1,1);error=1*10^(-6);nn=5;
while (nu<=l)
    % Locate the launching point into grid blocks
    % roundn(x,nn) returns x rounded to nn digits.
    NBx=fix((roundn(x1/Grid.hx,nn))+1);
    NBy=fix((roundn(y1/Grid.hy,nn))+1);
    XX(nu,1)=x1; YY(nu,1)=y1; j=1;
    while (NBx<=(Grid.Nx-bone) && NBy<=(Grid.Ny-bone));
        a1=a1_(NBx,NBy); a2=a2_(NBx,NBy); b1=b1_(NBx,NBy); b2=b2_(NBx,NBy);
        % Define the boundaries of local sub-cell
        x_2=NBx*Grid.hx; y_2=NBy*Grid.hy;x_1=x_2-Grid.hx; y_1=y_2-Grid.hy;
        % If the streamline is a straight line
        if abs(b1)<=error
            % M is the constant value of streamfunction
            M=a1*y1-a2*x1;
            Xab=[x_1,x_2]; xab=Xab(Xab~=x1); yab=(M+a2.*xab)./a1; yab=roundn(yab,nn);
            if sum(yab(yab<=y_2)>=y_1)~=0
                y2_=yab(yab<=y_2); y2=y2_(y2_>=y_1); x2=xab(y2==yab);
            else
                Yab=[y_1,y_2]; yab=Yab(Yab~=y1); xab=(a1.*yab-M)./a2;
                xab=roundn(xab,nn); x2_=xab(xab<=x_2); x2=x2_(x2_>=x_1); y2=yab(x2==(xab));
            end
        else
            % If the streamline is a hyperbola
            % M is the constant value of streamfunction
            M=a1*y1-a2*x1+b1/2*y1^2-b2/2*x1^2;
            % C=(M+a1^2/b1-a2^2/b2)*2/b1;
            % Determine the intersection points with the sub-cell boundaries
            nx=(y1*b1+a1)./(a1^2+2*a2*b1*x1+b1*b2*x1^2+2*b1*M)^0.5;
            nx=roundn(nx,nn);
            X_x=[x_1,x_2]; x_x=X_x(X_x~=x1);

```

```

y_x=(-a1+nx.*(a1.^2+2.*a2.*b1.*x_x+b1.*b2.*x_x.^2+2.*b1.*M).^0.5)./b1;
y_x=roundn(y_x,nn);
if sum(y_x(y_x<=y_2)>=y_1)~=0
    y2=y_x(y_x>=y_1);y2=y2_(y2<=y_2);x2=x_x(y2==y_x);
else
    ny=(x1*b2+a2)./(a2^2+2*a1*b2*y1+b1*b2*y1^2-2*b2*M)^0.5;
    ny=roundn(ny,nn);Y_y=[y_1,y_2];y_y=Y_y(Y_y==y1);
    x_y=(-a2+ny.*(a2^2+2.*a1.*b2.*y_y+b1.*b2.*y_y.^2-2*b2*M).^0.5)./b2;
    x_y=roundn(x_y,nn);x2=x_y(x_y>=x_1);x2=x2_(x2<=x_2);y2=y_y(x2==x_y);
end
end
% Determine the incremental time-of-flight
tt=TOF_bilinear(a1,a2,b1,b2,x1,x2,y1,y2,M);
XX(nu,j+1)=x2;YY(nu,j+1)=y2;tof(nu)=tof(nu)+tt;
% Use the exit point as the entry point for the next grid block
x1=x2; y1=y2; j=j+1;
if y2==y_2
    NBy=NBy+1;
elseif y2==y_1
    NBy=NBy-1;
end
if x2==x_2
    NBx=NBx+1;
elseif x2==x_1
    NBx=NBx-1;
end
end
plot(XX(nu,1:j),YY(nu,1:j),'black','LineWidth',1,'MarkerSize',6)
hold on
nu=nu+1;
if nu<=l
    y1=LY(nu); x1=LX(nu);
end
end

```

Code J.1-2: Determine the pressure and velocity coefficient in the Bilinear method

```

function [a1,a2,b1,b2]=v_coe_Bilinear(Grid,P,Mu,phi)
% Determine the pressure and velocity coefficients in the Bilinear method
%  $P = A1*x+A2*y+B*x*y+p0$ ;  $vx = a1*y+b1$ ;  $vy = a2*x+b2$ ;
% Define the boundary grid block properties
K_g=zeros(3,Grid.Nx+2,Grid.Ny+2,1);
K=Grid.K;
K_g(:,2:end-1,2:end-1,1)=K;K_g(:,1, :, 1)=K_g(:,2, :, 1);
K_g(:,end, :, 1)=K_g(:,end-1, :, 1);K_g(:, :, 1, 1)=K_g(:, :, 2, 1);
K_g(:, :, end, 1)=K_g(:, :, end-1, 1);
P_g=zeros(Grid.Nx+2,Grid.Ny+2);
P_g(2:end-1,2:end-1)=P;P_g(1, :)=P_g(2, :);P_g(end, :)=P_g(end-1, :);
P_g(:, 1)=P_g(:, 2);P_g(:, end)=P_g(:, end-1);
kx(1:Grid.Nx+2,1:Grid.Ny+2)=K_g(1,1:Grid.Nx+2,1:Grid.Ny+2,1);
ky(1:Grid.Nx+2,1:Grid.Ny+2)=K_g(2,1:Grid.Nx+2,1:Grid.Ny+2,1);
% Define the dual-cell system
Grid.Nxs=2*(Grid.Nx);Grid.Nys=2*(Grid.Ny);
kxs_=zeros(2*(Grid.Nx+2),2*(Grid.Ny+2));kys_=zeros(2*(Grid.Nx+2),2*(Grid.Ny+2));
% Assign the permeability in sub-cells
kxs_(1:2:end,1:2:end)=kx(1:end,1:end);kxs_(1:2:end,2:2:end)=kx(1:end,1:end);
kxs_(2:2:end,2:2:end)=kx(1:end,1:end);kxs_(2:2:end,1:2:end)=kx(1:end,1:end);
kys_(1:2:end,1:2:end)=ky(1:end,1:end);kys_(1:2:end,2:2:end)=ky(1:end,1:end);
kys_(2:2:end,2:2:end)=ky(1:end,1:end);kys_(2:2:end,1:2:end)=ky(1:end,1:end);
ks.x=kxs_(2:end-1,2:end-1);ks.y=kys_(2:end-1,2:end-1);
% Define the size of the pressure coefficients
A1=zeros(Grid.Nxs,Grid.Nys);A2=A1;B=A1;
% Determine the pressure coefficients in dual-cells
for i=1:Grid.Nx+1
    for j=1:Grid.Ny+1
        % Assign the permeabilites into subcells
        k1x=ks.x(i*2-1,j*2-1);    k1y=ks.y(i*2-1,j*2-1);
        k2x=ks.x(i*2,j*2-1);      k2y=ks.y(i*2,j*2-1);
        k3x=ks.x(i*2,j*2);        k3y=ks.y(i*2,j*2);
        k4x=ks.x(i*2-1,j*2);      k4y=ks.y(i*2-1,j*2);
    
```

```

% Assign the primary pressures into subcells
p1=P_g(i,j); p2=P_g(i+1,j); p3=P_g(i+1,j+1); p4=P_g(i,j+1);
% Define the coordinates of subcell vertexes
xm=(i-1)*Grid.hx;ym=(j-1)*Grid.hy;
xs=xm;ys=ym-Grid.hy/2;
xw=xm-Grid.hx/2;yw=ym;
xn=xm;yn=ym+Grid.hy/2;
xe=xm+Grid.hx/2;ye=ym;
x1=xm-Grid.hx/2; y1=ym-Grid.hy/2;
x2=xm+Grid.hx/2; y2=ym-Grid.hy/2;
x3=xm+Grid.hx/2; y3=ym+Grid.hy/2;
x4=xm-Grid.hx/2; y4=ym+Grid.hy/2;
% Determine the pressures at S, E, N, W
S=(p1*k1x+p2*k2x)/(k1x+k2x);N=(p3*k3x+p4*k4x)/(k3x+k4x);
E=(p2*k2y+p3*k3y)/(k2y+k3y);W=(p1*k1y+p4*k4y)/(k1y+k4y);
% Determine the pressure at dual-cell center
a=-k1y.*(-p1+W-S)-k1x.*(-p1-W+S)-k2y.*(-p2+E-S)+k2x.*(p2+E-S);
a=a+k3y.*(p3-E+N)+k3x.*(p3+E-N)+k4y.*(p4-W+N)-k4x.*(-p4-W+N);
b=-(k1x+k2x+k3x+k4x+k1y+k2y+k3y+k4y);
M=-a./b;
% Determine the pressure coefficients , P = A1*x+A2*y+B*x*y+p0;
I1=[p1,S,W,M]'; I2=[S,p2,M,E]'; I3=[M,E,N,p3]'; I4=[W,M,p4,N]';
I1_=[x1,y1,x1*y1,1;xs,ys,xs*ys,1;xw,yw,xw*yw,1;xm,ym,xm*ym,1;];
I2_=[xs,ys,xs*ys,1;x2,y2,x2*y2,1;xm,ym,xm*ym,1;xe,ye,xe*ye,1;];
I3_=[xm,ym,xm*ym,1;xe,ye,xe*ye,1;xn,yn,xn*yn,1;x3,y3,x3*y3,1;];
I4_=[xw,yw,xw*yw,1;xm,ym,xm*ym,1;x4,y4,x4*y4,1;xn,yn,xn*yn,1;];
F1=I1_ \ I1;F2=I2_ \ I2;F3=I3_ \ I3;F4=I4_ \ I4;
A1(i*2-1,j*2-1)=F1(1);A2(i*2-1,j*2-1)=F1(2); B(i*2-1,j*2-1)=F1(3);
A1(i*2,j*2-1)=F2(1); A2(i*2,j*2-1)=F2(2); B(i*2,j*2-1)=F2(3);
A1(i*2,j*2)=F3(1); A2(i*2,j*2)=F3(2); B(i*2,j*2)=F3(3);
A1(i*2-1,j*2)=F4(1); A2(i*2-1,j*2)=F4(2); B(i*2-1,j*2)=F4(3);
end
end
% Determine the velocity coefficients , vx = a1*y+b1; vy = a2*x+b2;
a1=-A1(2:end-1,2:end-1).*ks.x(2:end-1,2:end-1)./phi./Mu;

```

```

a2=-A2(2:end-1,2:end-1).*ks.y(2:end-1,2:end-1)./phi./Mu;
b1=-B(2:end-1,2:end-1).*ks.x(2:end-1,2:end-1)./phi./Mu;
b2=-B(2:end-1,2:end-1).*ks.y(2:end-1,2:end-1)./phi./Mu;

```

Code J.1-3: Determine incremental time-of-flight using the Bilinear method

```

function tt=TOF_bilinear(a1,a2,b1,b2,x1,x2,y1,y2,M)
% Determine the incremental time-of-flight in the local subcell
% a1, a2, b1, b2 are velocity coefficients; vx = a1*y+b1; vy = a2*x+b2;
% (x1, y1) and (x2, y2) are the entry and exits points of the streamline,
% M is the constant value of streamfunction. M = a1*y - a2*x + y^2*b1/2 - x^2*b2/2;
% If the streamfunction is a hyperbola equation,
if abs(b1)>=1*10^(-8)
    if abs(M)<=1*10^(-8) && abs(a1-a2)<=1*10^(-8) && abs(b1-b2)<=1*10^(-8)
        tt=1/b1*log((a1+b1*x2)/(a1+b1*x1));
    else
        t2=(a2*b1+b1*b2*x2)/(b1*b2)^0.5+(b1*b2*x2^2+2*a2*b1*x2+a1^2+2*b1*M)^0.5;
        t1=(a2*b1+b1*b2*x1)/(b1*b2)^0.5+(b1*b2*x1^2+2*a2*b1*x1+a1^2+2*b1*M)^0.5;
        tt=1/(b1*b2)^0.5*log(t2/t1);
    end
% If the streamfunction is a straight line,
elseif abs(a1)>=1*10^(-8)
    tt=(x2-x1)/a1;
else
    tt=(y2-y1)/a2;
end

```

J.2 The Trilinear streamline tracing method

Code J.2-1: Streamline tracing using the Trilinear method

```

clear all
% Tracing streamlines using the Trilinear method
% Define the number of grid blocks and the grid block length
Grid.Nx=20; Grid.hx=20/(Grid.Nx); Grid.Ny=Grid.Nx; Grid.hy=20/(Grid.Ny);
Grid.Nz=20; Grid.hz=20/(Grid.Nz);
Grid.K=ones(3,Grid.Nx,Grid.Ny,Grid.Nz);
nn=10;Q=10;Pi=5;Mu=1;phi=30;

```

```

% Define dual-cell system and calculate the velocity coefficient at sub-cells
[a1x, b1x, b2x, c1x, a2y, b1y, b3y, c2y, a3z, b2z, b3z, c3z]=trilinear_factor(Grid,Q,Mu,Pi);
Grid.Nxs=2*(Grid.Nx); Grid.Nys=2*(Grid.Ny); Grid.Nzs=2*(Grid.Nz);
hx=Grid.hx./2; hy=Grid.hy./2; hz=Grid.hz./2;
% Define the streamlines launching points
La=[0.5,0.5,2;1,0.5,2;1.5,0.5,2;0.5,1,2;1,1,2;1.5,1,2;0.5,1.5,2;
    1,1.5,2;1.5,1.5,2;2,0.5,0.5;2,1,0.5;2,1.5,0.5;2,0.50,1;
    2,1,1;2,1.5,1;2,0.5,1.5;2,1,1.5;2,1.5,1.5;0.5,2,0.5;0.5,2,1;
    0.50,2,1.5;1,2,0.5;1,2,1;1,2,1.5;1.5,2,0.5;1.5,2,1;1.5,2,1.5];
l=numel(La)/3; bone=Grid.Nxs/10;
% Define the size and the initial value of the time-of-flight
tof=zeros(1,1);
nu=1;m=4;er=1e-5;% m is the number of intervals used in RK-4th method
x1=La(nu,1); y1=La(nu,2); z1=La(nu,3);
while (nu<=l)
% Locate the launching point into grid blocks
% roundn(x,nn) returns x rounded to nn digits.
NBx=fix((roundn(x1/hx,nn))+1); NBy=fix((roundn(y1/hy,nn))+1);
NBz=fix((roundn(z1/hz,nn))+1);
j=1;XX(j)=x1;YY(j)=y1;ZZ(j)=z1;
while ((NBx<=(Grid.Nxs-bone) || NBy<=(Grid.Nys-bone) || NBz<=(Grid.Nzs-bone)))
% Define the boundaries of local sub-cell
x_2=NBx.*hx; y_2=NBy.*hy; z_2=NBz.*hz; x_1=x_2-hx; y_1=y_2-hy; z_1=z_2-hz;
A1x=a1x(NBx,NBy,NBz); B1x=b1x(NBx,NBy,NBz); B2x=b2x(NBx,NBy,NBz);
C1x=c1x(NBx,NBy,NBz); C2y=c2y(NBx,NBy,NBz); C3z=c3z(NBx,NBy,NBz);
A2y=a2y(NBx,NBy,NBz); B1y=b1y(NBx,NBy,NBz); B3y=b3y(NBx,NBy,NBz);
A3z=a3z(NBx,NBy,NBz); B2z=b2z(NBx,NBy,NBz); B3z=b3z(NBx,NBy,NBz);
% Calculate the entry directional-velocities
vx1=A1x+B1x*y1+B2x*z1+C1x*y1*z1; vy1=A2y+B1y*x1+B3y*z1+C2y*x1*z1;
vz1=A3z+B2z*x1+B3z*y1+C3z*x1*y1;
Si=0;
V1=abs([vx1,vy1,vz1]);
DoV=sort(V1,'descend');
pro=1;
% Determine the intersection points with the sub-cell boundaries using

```

```

% RK-4th method
while Si==0 && pro<=3
if abs(vx1)==DoV(pro) && Si==0
F = @(x,y,z) (A2y+B1y*x+B3y*z+C2y*x*z)./(A1x+B1x*y+B2x*z+C1x*y*z);
G = @(x,y,z) (A3z+B2z*x+B3z*y+C3z*x*y)./(A1x+B1x*y+B2x*z+C1x*y*z);
if abs(x1-x_2)>=er
[x_x,y_x,z_x]=RK4(F,G,x1,y1,z1,x_2,m);
y_x=round(y_x,5);z_x=round(z_x,5);
if y_x(end)<=y_2+er && y_x(end)>=y_1-er && z_x(end)<=z_2+er && z_x(end)>=z_1-er
Si=1;
end
end

if Si==0 && abs(x1-x_1)>=er
[x_x,y_x,z_x]=RK4(F,G,x1,y1,z1,x_1,m);y_x=round(y_x,5);z_x=round(z_x,5);
if y_x(end)<=y_2+er && y_x(end)>=y_1-er && z_x(end)<=z_2+er && z_x(end)>=z_1-er
Si=-1;
end
end
end

if vabs(vy1)==DoV(pro) && Si==0
F = @(y,x,z) (A1x+B1x*y+B2x*z+C1x*y*z)./(A2y+B1y*x+B3y*z+C2y*x*z);
G = @(y,x,z) (A3z+B2z*x+B3z*y+C3z*x*y)./(A2y+B1y*x+B3y*z+C2y*x*z);
[y_y,x_y,z_y]=RK4(F,G,y1,x1,z1,y_2,m);
if abs(y1-y_2)>=er
[y_y,x_y,z_y]=RK4(F,G,y1,x1,z1,y_2,m);x_y=round(x_y,5);z_y=round(z_y,5);
if x_y(end)<=x_2+er && x_y(end)>=x_1-er && z_y(end)<=z_2+er && z_y(end)>=z_1-er
Si=2;
end
end

if Si==0 && abs(y1-y_1)>=er
[y_y,x_y,z_y]=RK4(F,G,y1,x1,z1,y_1,m);x_y=round(x_y,5);z_y=round(z_y,5);
if x_y(end)<=x_2+er && x_y(end)>=x_1-er && z_y(end)<=z_2+er && z_y(end)>=z_1-er
Si=-2;
end

```

```

    end
end
end
if Si==0 && abs(vz1)==DoV(pro)
F = @(z,x,y) (A1x+B1x*y+B2x*z+C1x*y*z)./(A3z+B2z*x+B3z*y+C3z*x*y);
G = @(z,x,y) (A2y+B1y*x+B3y*z+C2y*x*z)./(A3z+B2z*x+B3z*y+C3z*x*y);
[z_z,x_z,y_z]=RK4(F,G,z1,x1,y1,z_2,m);
if vz1>0
    [z_z,x_z,y_z]=RK4(F,G,z1,x1,y1,z_2,m);y_z=round(y_z,5);x_z=round(x_z,5);
    if x_z(end)<=x_2+er && x_z(end)>=x_1-er && y_z(end)<=y_2+er && y_z(end)>=y_1-er
        Si=3;
    end
end
if vz1<0
    [z_z,x_z,y_z]=RK4(F,G,z1,x1,y1,z_1,m);y_z=round(y_z,5);x_z=round(x_z,5);
    if x_z(end)<=x_2+er && x_z(end)>=x_1-er && y_z(end)<=y_2+er && y_z(end)>=y_1-er
        Si=-3;
    end
end
end
end
pro=pro+1;
end
    if pro==5
        break
    end
    if abs(Si)==1
        x2=x_x(end);y2=y_x(end);z2=z_x(end);
    elseif abs(Si)==2
        x2=x_y(end);y2=y_y(end);z2=z_y(end);
    elseif abs(Si)==3
        x2=x_z(end);y2=y_z(end);z2=z_z(end);
    end
XX(j+1)=x2;YY(j+1)=y2;ZZ(j+1)=z2;
% Calculate the exit directional-velocities
vx2=A1x+B1x*y2+B2x*z2+C1x*y2*z2;

```

```

vy2=A2y+B1y*x2+B3y*z2+C2y*x2*z2;
vz2=A3z+B2z*x2+B3z*y2+C3z*x2*y2;
% Determine the incremental time-of-flight
if vx1*vx2>0
    Vx=[vx1;vx2];X=[x1,x2];tt=trapz(X,1./Vx);
elseif vy1*vy2>0
    Vy=[vy1;vy2];Y=[y1,y2];tt=trapz(Y,1./Vy);
else
    Vz=[vz1;vz2];Z=[z1,z2];tt=trapz(Z,1./Vz);
end
tof(nu)=tof(nu)+tt;
j=j+1;
% Use the exit point as the entry point for the next grid block
x1=x2;y1=y2;z1=z2;
NBx=fix((roundn(x1/hx,nn))+1);
NBy=fix((roundn(y1/hy,nn))+1);
NBz=fix((roundn(z1/hz,nn))+1);
end
plot3(XX(1:j),YY(1:j),ZZ(1:j),'black','LineWidth',2,'MarkerSize',6)
hold on
axis([0 Grid.Nxs*hx 0 Grid.Nys*hy 0 Grid.Nzs*hz])
nu=nu+1;
if nu<=1
x1=La(nu,1); y1=La(nu,2); z1=La(nu,3);
end
end
end

```

Code J.2-2: Determine the pressure and velocity coefficient in the Trilinear method

```

function [a1x,b1x,b2x,c1x,a2y,b1y,b3y,c2y,a3z,b2z,b3z,c3z]=...
    trilinear_factor(Grid,Q,Mu,Pi)
% Determine the pressure and velocity coefficients in the Trilinear method
% vx = (a1x+b1x*y+b2x*z+c1x*y*z); vy = (a2y+b1y*x+b3y*z+c2y*x*z);
% vz = (a3z+b2z*x+b3z*y+c3z*x*y);
% Calculate the primary pressures using a finite difference approach

```

```

[P]=FD_pressure(Grid,Q,Mu,Pi);
% Define the boundary grid block properties
P_g=zeros(Grid.Nx+2,Grid.Ny+2,Grid.Nz+2);
P_g(2:end-1,2:end-1,2:end-1)=P;P_g(1,,:)=P_g(2,,:);
P_g(end,,:)=P_g(end-1,,:);P_g(:,1,)=P_g(:,2,);P_g(:,end,)=P_g(:,end-1,);
P_g(:,:,1)=P_g(:,:,2);P_g(:,:,end)=P_g(:,:,end-1);
K_g=zeros(3,Grid.Nx+2,Grid.Ny+2,Grid.Nz+2);
K_g(:,2:end-1,2:end-1,2:end-1)=Grid.K;
K_g(:,1,,:)=K_g(:,2,,:);K_g(:,end,,:)=K_g(:,end-1,,:);
K_g(:,:,1,)=K_g(:,:,2,);K_g(:,:,end,)=K_g(:,:,end-1,);
K_g(:,:,:,1)=K_g(:,:,:,2);K_g(:,:,:,end)=K_g(:,:,:,end-1);
kx(1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2)=...
    K_g(1,1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2);
ky(1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2)=...
    K_g(2,1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2);
kz(1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2)=...
    K_g(3,1:Grid.Nx+2,1:Grid.Ny+2,1:Grid.Nz+2);
% Define the dual-cell system
kxs_=zeros(2*(Grid.Nx+2),2*(Grid.Ny+2),2*(Grid.Nz+2));
kys_=kxs_;kzs_=kxs_;
% Assign the permeability in sub-cells
kxs_(1:2:end,1:2:end,1:2:end)=kx(1:end,1:end,1:end);
kxs_(1:2:end,2:2:end,1:2:end)=kx(1:end,1:end,1:end);
kxs_(2:2:end,2:2:end,1:2:end)=kx(1:end,1:end,1:end);
kxs_(2:2:end,1:2:end,1:2:end)=kx(1:end,1:end,1:end);
kxs_(1:2:end,1:2:end,2:2:end)=kx(1:end,1:end,1:end);
kxs_(1:2:end,2:2:end,2:2:end)=kx(1:end,1:end,1:end);
kxs_(2:2:end,2:2:end,2:2:end)=kx(1:end,1:end,1:end);
kxs_(2:2:end,1:2:end,2:2:end)=kx(1:end,1:end,1:end);
kys_(1:2:end,1:2:end,1:2:end)=ky(1:end,1:end,1:end);
kys_(1:2:end,2:2:end,1:2:end)=ky(1:end,1:end,1:end);
kys_(2:2:end,2:2:end,1:2:end)=ky(1:end,1:end,1:end);
kys_(2:2:end,1:2:end,1:2:end)=ky(1:end,1:end,1:end);
kys_(1:2:end,1:2:end,2:2:end)=ky(1:end,1:end,1:end);
kys_(1:2:end,2:2:end,2:2:end)=ky(1:end,1:end,1:end);

```

```

kys_(2:2:end,2:2:end,2:2:end)=ky(1:end,1:end,1:end);
kys_(2:2:end,1:2:end,2:2:end)=ky(1:end,1:end,1:end);
kzs_(1:2:end,1:2:end,1:2:end)=kz(1:end,1:end,1:end);
kzs_(1:2:end,2:2:end,1:2:end)=kz(1:end,1:end,1:end);
kzs_(2:2:end,2:2:end,1:2:end)=kz(1:end,1:end,1:end);
kzs_(2:2:end,1:2:end,1:2:end)=kz(1:end,1:end,1:end);
kzs_(1:2:end,1:2:end,2:2:end)=kz(1:end,1:end,1:end);
kzs_(1:2:end,2:2:end,2:2:end)=kz(1:end,1:end,1:end);
kzs_(2:2:end,2:2:end,2:2:end)=kz(1:end,1:end,1:end);
kzs_(2:2:end,1:2:end,2:2:end)=kz(1:end,1:end,1:end);
ks.x=kxs_(2:end-1,2:end-1,2:end-1);
ks.y=kys_(2:end-1,2:end-1,2:end-1);ks.z=kzs_(2:end-1,2:end-1,2:end-1);
Grid.Nxs=2*Grid.Nx;Grid.Nys=2*Grid.Ny;Grid.Nzs=2*Grid.Nz;
% Determine the pressure coefficients in dual-cells
for i=1:Grid.Nx+1;
    for j=1:Grid.Ny+1;
        for k=1:Grid.Nz+1;
            % Assign the permeabilites into subcells
            k1x=ks.x(i*2-1,j*2-1,k*2-1);k1y=ks.y(i*2-1,j*2-1,k*2-1);k1z=ks.z(i*2-1,j*2-1,k*2-1);
            k2x=ks.x(i*2,j*2-1,k*2-1);k2y=ks.y(i*2,j*2-1,k*2-1);k2z=ks.z(i*2,j*2-1,k*2-1);
            k3x=ks.x(i*2,j*2,k*2-1);k3y=ks.y(i*2,j*2,k*2-1);k3z=ks.z(i*2,j*2,k*2-1);
            k4x=ks.x(i*2-1,j*2,k*2-1);k4y=ks.y(i*2-1,j*2,k*2-1);k4z=ks.z(i*2-1,j*2,k*2-1);
            k5x=ks.x(i*2-1,j*2-1,k*2);k5y=ks.y(i*2-1,j*2-1,k*2);k5z=ks.z(i*2-1,j*2-1,k*2);
            k6x=ks.x(i*2,j*2-1,k*2);k6y=ks.y(i*2,j*2-1,k*2);k6z=ks.z(i*2,j*2-1,k*2);
            k7x=ks.x(i*2,j*2,k*2);k7y=ks.y(i*2,j*2,k*2);k7z=ks.z(i*2,j*2,k*2);
            k8x=ks.x(i*2-1,j*2,k*2);k8y=ks.y(i*2-1,j*2,k*2);k8z=ks.z(i*2-1,j*2,k*2);
            % Assign the primary pressures into subcells
            p1=P_g(i,j,k);p2=P_g(i+1,j,k);p3=P_g(i+1,j+1,k);p4=P_g(i,j+1,k);
            p5=P_g(i,j,k+1);p6=P_g(i+1,j,k+1);p7=P_g(i+1,j+1,k+1);p8=P_g(i,j+1,k+1);
            % Determine the pressures at a, b, ..., l
            a=(k1x*p1+k2x*p2)/(k1x+k2x);b=(k2y*p2+k3y*p3)/(k2y+k3y);
            c=(k3x*p3+k4x*p4)/(k3x+k4x);d=(k1y*p1+k4y*p4)/(k1y+k4y);
            e=(k1z*p1+k5z*p5)/(k1z+k5z);f=(k2z*p2+k6z*p6)/(k2z+k6z);
            g=(k3z*p3+k7z*p7)/(k3z+k7z);h=(k4z*p4+k8z*p8)/(k4z+k8z);
            I=(k5x*p5+k6x*p6)/(k5x+k6x);J=(k6y*p6+k7y*p7)/(k6y+k7y);
        end
    end
end

```

```

K=(k7x*p7+k8x*p8)/(k7x+k8x); l=(k5y*p5+k8y*p8)/(k5y+k8y);
% Determine the pressures at x1, x2, ..., z1, z2
x1=(d*(k1z+k4z)/Grid.hz+h*(k4y+k8y)/Grid.hy+I*(k5z+k8z)/Grid.hz...
+e*(k1y+k5y)/Grid.hy)/((k1y+k4y+k8y+k5y)/Grid.hy+(k1z+k4z+k8z+k5z)/Grid.hz);
x2=(b*(k2z+k3z)/Grid.hz+g*(k3y+k7y)/Grid.hy+J*(k6z+k7z)/Grid.hz...
+f*(k2y+k6y)/Grid.hy)/((k2y+k3y+k7y+k6y)/Grid.hy+(k2z+k3z+k7z+k6z)/Grid.hz);
y1=(a*(k1z+k2z)/Grid.hz+f*(k2x+k6x)/Grid.hx+I*(k5z+k6z)/Grid.hz...
+e*(k1x+k5x)/Grid.hx)/((k1z+k2z+k5z+k6z)/Grid.hz+(k1x+k2x+k5x+k6x)/Grid.hx);
y2=(c*(k3z+k4z)/Grid.hz+g*(k3x+k7x)/Grid.hx+K*(k7z+k8z)/Grid.hz...
+h*(k5x+k8x)/Grid.hx)/((k4z+k3z+k7z+k8z)/Grid.hz+(k4x+k3x+k7x+k8x)/Grid.hx);
z1=(a*(k1y+k2y)/Grid.hy+b*(k2x+k3x)/Grid.hx+c*(k3y+k4y)/Grid.hy...
+d*(k2x+k4x)/Grid.hy)/((k1y+k2y+k3y+k4y)/Grid.hy+(k1x+k2x+k3x+k4x)/Grid.hx);
z2=(I*(k5y+k6y)/Grid.hy+J*(k6x+k7x)/Grid.hx+K*(k7y+k8y)/Grid.hy...
+l*(k5x+k8x)/Grid.hy)/((k5y+k6y+k7y+k8y)/Grid.hy+(k5x+k6x+k7x+k8x)/Grid.hx);
% Determine the pressures at dual-cell center, M
fx1=k1x*Grid.hy*Grid.hz/4/Grid.hx*(a-p1+z1-d+y1-e-x1);
fy1=k1y*Grid.hx*Grid.hz/4/Grid.hy*(d-p1+z1-a-y1+x1-e);
fz1=k1z*Grid.hx*Grid.hy/4/Grid.hz*(e-p1+y1-a-z1+x1-d);
fx2=k2x*Grid.hy*Grid.hz/4/Grid.hx*(p2-a+b-z1+f-y1+x2);
fy2=k2y*Grid.hx*Grid.hz/4/Grid.hy*(z1-a+b-p2+x2-f-y1);
fz2=k2z*Grid.hx*Grid.hy/4/Grid.hz*(y1-a+f-p2+x2-b-z1);
fx3=k3x*Grid.hy*Grid.hz/4/Grid.hx*(b-z1+p3-c+x2+g-y2);
fy3=k3y*Grid.hx*Grid.hz/4/Grid.hy*(c-z1+p3-b+g-x2+y2);
fz3=k3z*Grid.hx*Grid.hy/4/Grid.hz*(x2-b+g-p3+y2-c-z1);
fx4=k4x*Grid.hy*Grid.hz/4/Grid.hx*(z1-d+c-p4+y2-h-x1);
fy4=k4y*Grid.hx*Grid.hz/4/Grid.hy*(p4-d+c-z1+y2+h-x1);
fz4=k4z*Grid.hx*Grid.hy/4/Grid.hz*(x1-d+h-p4+y2-c-z1);
fx5=k5x*Grid.hy*Grid.hz/4/Grid.hx*(y1-e-x1+z2-l+I-p5);
fy5=k5y*Grid.hx*Grid.hz/4/Grid.hy*(x1-e-y1+z2-I+l-p5);
fz5=k5z*Grid.hx*Grid.hy/4/Grid.hz*(p5-e+I-y1+z2+l-x1);
fx6=k6x*Grid.hy*Grid.hz/4/Grid.hx*(f-y1+x2+J-z2+p6-I);
fy6=k6y*Grid.hx*Grid.hz/4/Grid.hy*(x2-f-y1+J-p6+z2-I);
fz6=k6z*Grid.hx*Grid.hy/4/Grid.hz*(I-y1+p6-f+J-x2+z2);
fx7=k7x*Grid.hy*Grid.hz/4/Grid.hx*(x2+g-y2+J-z2+p7-K);
fy7=k7y*Grid.hx*Grid.hz/4/Grid.hy*(y2+g-x2+p7-J+K-z2);

```

```

fz7=k7z*Grid . hx*Grid . hy/4/ Grid . hz*(z2+J-x2+p7-g+K-y2);
fx8=k8x*Grid . hy*Grid . hz/4/ Grid . hx*(y2-h+z2-l+K-p8-x1);
fy8=k8y*Grid . hx*Grid . hz/4/ Grid . hy*(h-x1+y2+K-z2+p8-l);
fz8=k8z*Grid . hx*Grid . hy/4/ Grid . hz*(1-x1+z2+K-y2+p8-h);
A=-fx1-fy1-fz1+fx2-fy2-fz2+fx3+fy3-fz3-fx4+fy4-fz4-fx5-fy5...
    +fz5+fx6-fy6+fz6+fx7+fy7+fz7-fx8+fy8+fz8;
B=Grid . hy*Grid . hz/4/ Grid . hx+k1y*Grid . hx*Grid . hz/4/ Grid . hy...
    +k1z*Grid . hx*Grid . hy/4/ Grid . hz+k2x*Grid . hy*Grid . hz/4/ Grid . hx...
    +k2y*Grid . hx*Grid . hz/4/ Grid . hy+k2z*Grid . hx*Grid . hy/4/ Grid . hz...
    +k3x*Grid . hy*Grid . hz/4/ Grid . hx+k3y*Grid . hx*Grid . hz/4/ Grid . hy...
    +k3z*Grid . hx*Grid . hy/4/ Grid . hz+k4x*Grid . hy*Grid . hz/4/ Grid . hx...
    +k4y*Grid . hx*Grid . hz/4/ Grid . hy+k4z*Grid . hx*Grid . hy/4/ Grid . hz...
    +k5x*Grid . hy*Grid . hz/4/ Grid . hx+k5y*Grid . hx*Grid . hz/4/ Grid . hy...
    +k5z*Grid . hx*Grid . hy/4/ Grid . hz +k6x*Grid . hy*Grid . hz/4/ Grid . hx...
    +k6y*Grid . hx*Grid . hz/4/ Grid . hy+k6z*Grid . hx*Grid . hy/4/ Grid . hz...
    +k7x*Grid . hy*Grid . hz/4/ Grid . hx+k7y*Grid . hx*Grid . hz/4/ Grid . hy...
    +k7z*Grid . hx*Grid . hy/4/ Grid . hz+k8x*Grid . hy*Grid . hz/4/ Grid . hx...
    +k8y*Grid . hx*Grid . hz/4/ Grid . hy+k8z*Grid . hx*Grid . hy/4/ Grid . hz;
M=A./B;    m(i,j,k)=M;
% Define the coordinates of subcell vertexes
Xm=(i-1)*Grid . hx;Ym=(j-1)*Grid . hy;Zm=(k-1)*Grid . hz;
X1=Xm-Grid . hx/2;Y1=Ym-Grid . hy/2;Z1=Zm-Grid . hz/2;
X2=Xm+Grid . hx/2;Y2=Ym-Grid . hy/2;Z2=Zm-Grid . hz/2;
X3=Xm+Grid . hx/2;Y3=Ym+Grid . hy/2;Z3=Zm-Grid . hz/2;
X4=Xm-Grid . hx/2;Y4=Ym+Grid . hy/2;Z4=Zm-Grid . hz/2;
X5=Xm-Grid . hx/2;Y5=Ym-Grid . hy/2;Z5=Zm+Grid . hz/2;
X6=Xm+Grid . hx/2;Y6=Ym-Grid . hy/2;Z6=Zm+Grid . hz/2;
X7=Xm+Grid . hx/2;Y7=Ym+Grid . hy/2;Z7=Zm+Grid . hz/2;
X8=Xm-Grid . hx/2;Y8=Ym+Grid . hy/2;Z8=Zm+Grid . hz/2;
Xa=Xm;Ya=Ym-Grid . hy/2;Za=Zm-Grid . hz/2;
Xb=Xm+Grid . hx/2;Yb=Ym;Zb=Zm-Grid . hz/2;
Xc=Xm;Yc=Ym+Grid . hy/2;Zc=Zm-Grid . hz/2;
Xd=Xm-Grid . hx/2;Yd=Ym;Zd=Zm-Grid . hz/2;
Xe=Xm-Grid . hx/2;Ye=Ym-Grid . hy/2;Ze=Zm;
Xf=Xm+Grid . hx/2;Yf=Ym-Grid . hy/2;Zf=Zm;

```

```

Xg=Xm+Grid . hx / 2; Yg=Ym+Grid . hy / 2; Zg=Zm;
Xh=Xm-Grid . hx / 2; Yh=Ym+Grid . hy / 2; Zh=Zm;
Xi=Xm; Yi=Ym-Grid . hy / 2; Zi=Zm+Grid . hz / 2;
Xj=Xm+Grid . hx / 2; Yj=Ym; Zj=Zm+Grid . hz / 2;
Xk=Xm; Yk=Ym+Grid . hy / 2; Zk=Zm+Grid . hz / 2;
Xl=Xm-Grid . hx / 2; Yl=Ym; Zl=Zm+Grid . hz / 2;
Xz1=Xm; Yz1=Ym; Zz1=Zm-Grid . hz / 2;
Xz2=Xm; Yz2=Ym; Zz2=Zm+Grid . hz / 2;
Xx1=Xm-Grid . hx / 2; Yx1=Ym; Zx1=Zm;
Xx2=Xm+Grid . hx / 2; Yx2=Ym; Zx2=Zm;
Xy1=Xm; Yy1=Ym-Grid . hy / 2; Zy1=Zm;
Xy2=Xm; Yy2=Ym+Grid . hy / 2; Zy2=Zm;
% Determine the pressure coefficients ,
%P = p0+A1*x+A2*y+A3*z+B1*x*y+B2*x*z+B3*y*z+C*x*y*z ;
I1=[p1 , a , z1 , d , e , y1 , M , x1 ] ' ; I2=[a , p2 , b , z1 , y1 , f , x2 , M] ' ;
I3=[z1 , b , p3 , c , M , x2 , g , y2 ] ' ; I4=[d , z1 , c , p4 , x1 , M , y2 , h ] ' ;
I5=[e , y1 , M , x1 , p5 , I , z2 , l ] ' ; I6=[y1 , f , x2 , M , I , p6 , J , z2 ] ' ;
I7=[M , x2 , g , y2 , z2 , J , p7 , K] ' ; I8=[x1 , M , y2 , h , l , z2 , K , p8 ] ' ;
I1_=[1 , X1 , Y1 , Z1 , X1*Y1 , X1*Z1 , Y1*Z1 , X1*Y1*Z1 ;
1 , Xa , Ya , Za , Xa*Ya , Xa*Za , Ya*Za , Xa*Ya*Za ;
1 , Xz1 , Yz1 , Zz1 , Xz1*Yz1 , Xz1*Zz1 , Yz1*Zz1 , Xz1*Yz1*Zz1 ;
1 , Xd , Yd , Zd , Xd*Yd , Xd*Zd , Yd*Zd , Xd*Yd*Zd ;
1 , Xe , Ye , Ze , Xe*Ye , Xe*Ze , Ye*Ze , Xe*Ye*Ze ;
1 , Xy1 , Yy1 , Zy1 , Xy1*Yy1 , Xy1*Zy1 , Yy1*Zy1 , Xy1*Yy1*Zy1 ;
1 , Xm , Ym , Zm , Xm*Ym , Xm*Zm , Ym*Zm , Xm*Ym*Zm ;
1 , Xx1 , Yx1 , Zx1 , Xx1*Yx1 , Xx1*Zx1 , Yx1*Zx1 , Xx1*Yx1*Zx1 ; ] ;
I2_=[1 , Xa , Ya , Za , Xa*Ya , Xa*Za , Ya*Za , Xa*Ya*Za ;
1 , X2 , Y2 , Z2 , X2*Y2 , X2*Z2 , Y2*Z2 , X2*Y2*Z2 ;
1 , Xb , Yb , Zb , Xb*Yb , Xb*Zb , Yb*Zb , Xb*Yb*Zb ;
1 , Xz1 , Yz1 , Zz1 , Xz1*Yz1 , Xz1*Zz1 , Yz1*Zz1 , Xz1*Yz1*Zz1 ;
1 , Xy1 , Yy1 , Zy1 , Xy1*Yy1 , Xy1*Zy1 , Yy1*Zy1 , Xy1*Yy1*Zy1 ;
1 , Xf , Yf , Zf , Xf*Yf , Xf*Zf , Yf*Zf , Xf*Yf*Zf ;
1 , Xx2 , Yx2 , Zx2 , Xx2*Yx2 , Xx2*Zx2 , Yx2*Zx2 , Xx2*Yx2*Zx2 ;
1 , Xm , Ym , Zm , Xm*Ym , Xm*Zm , Ym*Zm , Xm*Ym*Zm ; ] ;
I3_=[1 , Xz1 , Yz1 , Zz1 , Xz1*Yz1 , Xz1*Zz1 , Yz1*Zz1 , Xz1*Yz1*Zz1 ;

```

$1, Xb, Yb, Zb, Xb*Yb, Xb*Zb, Yb*Zb, Xb*Yb*Zb;$
 $1, X3, Y3, Z3, X3*Y3, X3*Z3, Y3*Z3, X3*Y3*Z3;$
 $1, Xc, Yc, Zc, Xc*Yc, Xc*Zc, Yc*Zc, Xc*Yc*Zc;$
 $1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;$
 $1, Xx2, Yx2, Zx2, Xx2*Yx2, Xx2*Zx2, Yx2*Zx2, Xx2*Yx2*Zx2;$
 $1, Xg, Yg, Zg, Xg*Yg, Xg*Zg, Yg*Zg, Xg*Yg*Zg;$
 $1, Xy2, Yy2, Zy2, Xy2*Yy2, Xy2*Zy2, Yy2*Zy2, Xy2*Yy2*Zy2;];$

$I4_=[1, Xd, Yd, Zd, Xd*Yd, Xd*Zd, Yd*Zd, Xd*Yd*Zd;$
 $1, Xz1, Yz1, Zz1, Xz1*Yz1, Xz1*Zz1, Yz1*Zz1, Xz1*Yz1*Zz1;$
 $1, Xc, Yc, Zc, Xc*Yc, Xc*Zc, Yc*Zc, Xc*Yc*Zc;$
 $1, X4, Y4, Z4, X4*Y4, X4*Z4, Y4*Z4, X4*Y4*Z4;$
 $1, Xx1, Yx1, Zx1, Xx1*Yx1, Xx1*Zx1, Yx1*Zx1, Xx1*Yx1*Zx1;$
 $1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;$
 $1, Xy2, Yy2, Zy2, Xy2*Yy2, Xy2*Zy2, Yy2*Zy2, Xy2*Yy2*Zy2;$
 $1, Xh, Yh, Zh, Xh*Yh, Xh*Zh, Yh*Zh, Xh*Yh*Zh;];$

$I5_=[1, Xe, Ye, Ze, Xe*Ye, Xe*Ze, Ye*Ze, Xe*Ye*Ze;$
 $1, Xy1, Yy1, Zy1, Xy1*Yy1, Xy1*Zy1, Yy1*Zy1, Xy1*Yy1*Zy1;$
 $1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;$
 $1, Xx1, Yx1, Zx1, Xx1*Yx1, Xx1*Zx1, Yx1*Zx1, Xx1*Yx1*Zx1;$
 $1, X5, Y5, Z5, X5*Y5, X5*Z5, Y5*Z5, X5*Y5*Z5;$
 $1, Xi, Yi, Zi, Xi*Yi, Xi*Zi, Yi*Zi, Xi*Yi*Zi;$
 $1, Xz2, Yz2, Zz2, Xz2*Yz2, Xz2*Zz2, Yz2*Zz2, Xz2*Yz2*Zz2;$
 $1, Xl, Yl, Zl, Xl*Yl, Xl*Zl, Yl*Zl, Xl*Yl*Zl;];$

$I6_=[1, Xy1, Yy1, Zy1, Xy1*Yy1, Xy1*Zy1, Yy1*Zy1, Xy1*Yy1*Zy1;$
 $1, Xf, Yf, Zf, Xf*Yf, Xf*Zf, Yf*Zf, Xf*Yf*Zf;$
 $1, Xx2, Yx2, Zx2, Xx2*Yx2, Xx2*Zx2, Yx2*Zx2, Xx2*Yx2*Zx2;$
 $1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;$
 $1, Xi, Yi, Zi, Xi*Yi, Xi*Zi, Yi*Zi, Xi*Yi*Zi;$
 $1, X6, Y6, Z6, X6*Y6, X6*Z6, Y6*Z6, X6*Y6*Z6;$
 $1, Xj, Yj, Zj, Xj*Yj, Xj*Zj, Yj*Zj, Xj*Yj*Zj;$
 $1, Xz2, Yz2, Zz2, Xz2*Yz2, Xz2*Zz2, Yz2*Zz2, Xz2*Yz2*Zz2;];$

$I7_=[1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;$
 $1, Xx2, Yx2, Zx2, Xx2*Yx2, Xx2*Zx2, Yx2*Zx2, Xx2*Yx2*Zx2;$
 $1, Xg, Yg, Zg, Xg*Yg, Xg*Zg, Yg*Zg, Xg*Yg*Zg;$
 $1, Xy2, Yy2, Zy2, Xy2*Yy2, Xy2*Zy2, Yy2*Zy2, Xy2*Yy2*Zy2;$

```

1, Xz2, Yz2, Zz2, Xz2*Yz2, Xz2*Zz2, Yz2*Zz2, Xz2*Yz2*Zz2;
1, Xj, Yj, Zj, Xj*Yj, Xj*Zj, Yj*Zj, Xj*Yj*Zj;
1, X7, Y7, Z7, X7*Y7, X7*Z7, Y7*Z7, X7*Y7*Z7;
1, Xk, Yk, Zk, Xk*Yk, Xk*Zk, Yk*Zk, Xk*Yk*Zk; ];
I8_=[1, Xx1, Yx1, Zx1, Xx1*Yx1, Xx1*Zx1, Yx1*Zx1, Xx1*Yx1*Zx1;
1, Xm, Ym, Zm, Xm*Ym, Xm*Zm, Ym*Zm, Xm*Ym*Zm;
1, Xy2, Yy2, Zy2, Xy2*Yy2, Xy2*Zy2, Yy2*Zy2, Xy2*Yy2*Zy2;
1, Xh, Yh, Zh, Xh*Yh, Xh*Zh, Yh*Zh, Xh*Yh*Zh;
1, Xl, Yl, Zl, Xl*Yl, Xl*Zl, Yl*Zl, Xl*Yl*Zl;
1, Xz2, Yz2, Zz2, Xz2*Yz2, Xz2*Zz2, Yz2*Zz2, Xz2*Yz2*Zz2;
1, Xk, Yk, Zk, Xk*Yk, Xk*Zk, Yk*Zk, Xk*Yk*Zk;
1, X8, Y8, Z8, X8*Y8, X8*Z8, Y8*Z8, X8*Y8*Z8; ];

```

% %

```

F1=I1_ \ I1 ; F2=I2_ \ I2 ; F3=I3_ \ I3 ; F4=I4_ \ I4 ;
F5=I5_ \ I5 ; F6=I6_ \ I6 ; F7=I7_ \ I7 ; F8=I8_ \ I8 ;
A1_( i *2 -1, j *2 -1, k*2-1)=F1 ( 2);    A2_( i *2 -1, j *2 -1, k*2-1)=F1 ( 3);
A3_( i *2 -1, j *2 -1, k*2-1)=F1 ( 4);    B1_( i *2 -1, j *2 -1, k*2-1)=F1 ( 5);
B2_( i *2 -1, j *2 -1, k*2-1)=F1 ( 6);    B3_( i *2 -1, j *2 -1, k*2-1)=F1 ( 7);
C_( i *2 -1, j *2 -1, k*2-1)=F1 ( 8);    C_( i *2, j *2 -1, k*2-1)=F2 ( 8);
A1_( i *2, j *2 -1, k*2-1)=F2 ( 2);    A2_( i *2, j *2 -1, k*2-1)=F2 ( 3);
A3_( i *2, j *2 -1, k*2-1)=F2 ( 4);    B1_( i *2, j *2 -1, k*2-1)=F2 ( 5);
B2_( i *2, j *2 -1, k*2-1)=F2 ( 6);    B3_( i *2, j *2 -1, k*2-1)=F2 ( 7);
A1_( i *2, j *2, k*2-1)=F3 ( 2);    A2_( i *2, j *2, k*2-1)=F3 ( 3);
A3_( i *2, j *2, k*2-1)=F3 ( 4);    B1_( i *2, j *2, k*2-1)=F3 ( 5);
B2_( i *2, j *2, k*2-1)=F3 ( 6);    B3_( i *2, j *2, k*2-1)=F3 ( 7);
C_( i *2, j *2, k*2-1)=F3 ( 8);    C_( i *2-1, j *2, k*2-1)=F4 ( 8);
A1_( i *2-1, j *2, k*2-1)=F4 ( 2);    A2_( i *2-1, j *2, k*2-1)=F4 ( 3);
A3_( i *2-1, j *2, k*2-1)=F4 ( 4);    B1_( i *2-1, j *2, k*2-1)=F4 ( 5);
B2_( i *2-1, j *2, k*2-1)=F4 ( 6);    B3_( i *2-1, j *2, k*2-1)=F4 ( 7);
A1_( i *2-1, j *2-1, k*2)=F5 ( 2);    A2_( i *2-1, j *2-1, k*2)=F5 ( 3);
A3_( i *2-1, j *2-1, k*2)=F5 ( 4);    B1_( i *2-1, j *2-1, k*2)=F5 ( 5);
B2_( i *2-1, j *2-1, k*2)=F5 ( 6);    B3_( i *2-1, j *2-1, k*2)=F5 ( 7);
C_( i *2-1, j *2-1, k*2)=F5 ( 8);    C_( i *2, j *2-1, k*2)=F6 ( 8);
A1_( i *2, j *2-1, k*2)=F6 ( 2);    A2_( i *2, j *2-1, k*2)=F6 ( 3);
A3_( i *2, j *2-1, k*2)=F6 ( 4);    B1_( i *2, j *2-1, k*2)=F6 ( 5);

```

```

B2_( i *2 , j *2 -1 , k *2 )=F6 ( 6 ) ;      B3_( i *2 , j *2 -1 , k *2 )=F6 ( 7 ) ;
A1_( i *2 , j *2 , k *2 )=F7 ( 2 ) ;      A2_( i *2 , j *2 , k *2 )=F7 ( 3 ) ;
A3_( i *2 , j *2 , k *2 )=F7 ( 4 ) ;      B1_( i *2 , j *2 , k *2 )=F7 ( 5 ) ;
B2_( i *2 , j *2 , k *2 )=F7 ( 6 ) ;      B3_( i *2 , j *2 , k *2 )=F7 ( 7 ) ;
C_( i *2 , j *2 , k *2 )=F7 ( 8 ) ;      C_( i *2 -1 , j *2 , k *2 )=F8 ( 8 ) ;
A1_( i *2 -1 , j *2 , k *2 )=F8 ( 2 ) ;    A2_( i *2 -1 , j *2 , k *2 )=F8 ( 3 ) ;
A3_( i *2 -1 , j *2 , k *2 )=F8 ( 4 ) ;    B1_( i *2 -1 , j *2 , k *2 )=F8 ( 5 ) ;
B2_( i *2 -1 , j *2 , k *2 )=F8 ( 6 ) ;    B3_( i *2 -1 , j *2 , k *2 )=F8 ( 7 ) ;
end
end
end
% Determine the velocity coefficients , vx = ( a1x+b1x*y+b2x*z+c1x*y*z ) ;
% vy = ( a2y+b1y*x+b3y*z+c2y*x*z ) ; vz = ( a3z+b2z*x+b3z*y+c3z*x*y ) ;
a1x=-A1_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . x ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
a2y=-A2_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . y ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
a3z=-A3_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . z ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b1x=-B1_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . x ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b2x=-B2_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . x ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b1y=-B1_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . y ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b3y=-B3_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . y ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b2z=-B2_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . z ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
b3z=-B3_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . z ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
c1x=-C_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . x ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
c2y=-C_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . y ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;
c3z=-C_( 2 : end -1 , 2 : end -1 , 2 : end -1 ) . * ks . z ( 2 : end -1 , 2 : end -1 , 2 : end -1 ) ;

```

J.3 The 2D Cubic streamline tracing method

Code J.3-1: Streamline tracing using the 2D Cubic method

```

clear all
% Tracing streamlines using the Cubic 2D method
% Define the number of grid blocks and the grid block length
Grid.Nx=10; Grid.Ny=Grid.Nx; Grid.Nz=1; N=Grid.Nx*Grid.Ny;
Grid.hx=100/Grid.Nx; Grid.hy=100/Grid.Ny; Grid.hz=1;
Grid.K=ones(3, Grid.Nx, Grid.Ny, Grid.Nz);
% Muo, oil viscosity; ao, the maximum oil relative permeability

```

```

Q=10;Pi=20;phi=0.3;Mu=5;ao=1;
% Calculate pressure and pressure derivatives using a finite difference
% approach
[P,U,dp]=FD_pressure(Grid,Q,Mu,phi);
% Determine the velocity coefficients in the Cubic 2D method
[a1,a2,b,c1,c2,d1,d2]=Cubic_2d_factor(Grid,dp,phi,Mu);nn=12;
% roundn(x,nn) returns x rounded to nn digits.
a1=roundn(a1,nn);a2=roundn(a2,nn);b=roundn(b,nn);
c1=roundn(c1,nn);c2=roundn(c2,nn);d1=roundn(d1,nn);d2=roundn(d2,nn);
% Define the streamlines launching points
LX=[0.7,1.4,2.2,3.2,4.3,5.9,7.7,10,10*ones(1,7)];
LY=[10*ones(1,8),0.7,1.4,2.2,3.2,4.3,5.9,7.7];
l=numel(LX);nu=1;bone=Grid.Nx/10;x1=LX(nu);y1=LY(nu);tof=zeros(1,1);
% Determine the reference flow rate along each streamline
[q]=q_assign2d(a1,a2,b,c1,c2,d1,d2,Grid,l,Q,LX,LY,phi);
while (nu<=l)
% Locate the launching point into grid blocks
NBx=fix((roundn(x1/Grid.hx,nn)))+1;NBy=fix((roundn(y1/Grid.hy,nn)))+1;
XX(nu,1)=x1;YY(nu,1)=y1;j=1;dT(nu,1)=0;
while (NBx<=(Grid.Nx-bone) || NBy<=(Grid.Ny-bone))
A1=a1(NBx,NBy);A2=a2(NBx,NBy);B=b(NBx,NBy);
C1=c1(NBx,NBy);C2=c2(NBx,NBy);D1=d1(NBx,NBy);D2=d2(NBx,NBy);
% Define the boundaries of local grid block
x_2=NBx*Grid.hx;y_2=NBy*Grid.hy;x_1=x_2-Grid.hx;y_1=y_2-Grid.hy;
% Calculate the constant value of streamline function M;
M=A2*x1-A1*y1-B*x1*y1-C1/2*y1^2+C2/2*x1^2+D1*x1^2*y1+D2*x1*y1^2-...
D2/3*y1^3-D1/3*x1^3;
% Determine the intersection points with the grid block boundaries in terms of y
% by solving the equation T3*y^3+S2*y^2+F1*y+C0=0;
T3=-D2/3;S2=D2.*x_2-C1/2;F1=D1*x_2^2-B*x_2-A1;C0=A2*x_2+C2*x_2^2/2-...
D1*x_2^3/3-M;
y_ab=roots([T3 S2 F1 C0]);y_x=y_ab(imag(y_ab)==0);y_x=roundn(y_x,nn);
if sum(isreal(y_x))~=0 && sum(y_x(y_x<=y_2)>=y_1)~=0
y2=y_x(y_x<=y_2);y2=y2_(y2>=y_1);x2=x_2;
else

```

```

    % Determine the intersection points with the grid block boundaries in terms of x
    % by solving the equation  $T3*x^3+S2*x^2+F1*x+C0=0$ ;
    T3=-D1/3; S2=D1.*y_2+C2/2; F1=D2*y_2^2-B*y_2+A2; C0=-A1*y_2-C1*y_2^2/2-...
    D2*y_2^3/3-M;
    x_ab=roots([T3 S2 F1 C0]); x_y=x_ab(imag(x_ab) == 0); x_y=roundn(x_y,nn);
    x2=x_y(x_y<=x_2); x2=x2_(x2_>=x_1); y2=y_2;
    end
x2=round(x2,nn); y2=round(y2,nn); X=[x1,x2]; Y=[y1,y2]; XX(nu,j+1)=x2; YY(nu,j+1)=y2;
% Determine the entry and exit directional velocities
Vx=A1+B.*X+C1.*Y+D2.*Y.^2-2.*D2.*X.*Y-D1.*X.^2;
Vy=A2-B.*Y+C2.*X-D1.*X.^2+2.*D1.*X.*Y+D2.*Y.^2;
% Calculate the total velocity ut and directional permeability k
kx=Grid.K(1,NBx,NBy,1); ky=Grid.K(2,NBx,NBy,1);
if j==1
    ut(nu,1)=(Vx(1)^2+Vy(1)^2)^0.5;
    k(nu,1)=kx.*(Vx(1)/ut(nu,1)).^2+ky.*(Vy(1)/ut(nu,1)).^2;
end
ut(nu,j+1)=(Vx(2)^2+Vy(2)^2)^0.5;
k(nu,j+1)=kx.*(Vx(2)/ut(nu,j+1)).^2+ky.*(Vy(2)/ut(nu,j+1)).^2;
% Calculate the incremental time-of-flight
if Vx(1)*Vx(2)>0
    dTOF=trapz(X,1./Vx);
else
    dTOF=trapz(Y,1./Vy);
end
dT(nu,j+1)=dT(nu,j)+dTOF;
% Use the exit point as the entry point for the next grid block
x1=x2; y1=y2; j=j+1;
NBx=fix((roundn(x1/Grid.hx,nn))+1); NBy=fix((roundn(y1/Grid.hy,nn))+1);
end
plot(XX(nu,1:j),YY(nu,1:j),'g','LineWidth',1,'MarkerSize',6)
hold on
nu=nu+1;
if nu<=l
    y1=LY(nu); x1=LX(nu);

```

```

end
end
% Determine the differential pressure of each streamtube
[DP]=p_assign(Muo,ao,k,l,ut,dT,phi);

```

Code J.3-2: Determine the velocity coefficient in the 2D Cubic method

```

function [a1,a2,b,c1,c2,d_1,d_2,V2]=Cubic_2d_factor(Grid,dp,phi,Muo)
% Determine the velocity coefficients in the Cubic 2D method
% vx = a1+b*x+c1*y+d2.*y^2-2*d2*x*y-d1*x^2;
% vy = a2-b*y+c2*x-d1.*x^2+2*d1*x*y+d2*y^2;
Nx=Grid.Nx;Ny=Grid.Ny;hx=Grid.hx;hy=Grid.hy;
% Determine the velocity at grid block vertexes
[V2]=Cubic_2d_velocity(Grid,dp,Muo,phi);
Vx1=V2.x(1:1:end-1,1:2:end-1);Vx2=V2.x(2:1:end,1:2:end-1);
Vx4=V2.x(1:1:end-1,2:2:end);Vx3=V2.x(2:1:end,2:2:end);
Vy1=V2.y(1:2:end-1,1:1:end-1);Vy2=V2.y(2:2:end,1:1:end-1);
Vy4=V2.y(1:2:end-1,2:1:end);Vy3=V2.y(2:2:end,2:1:end);
Factor=zeros(Nx,Ny,7);
for i=1:Nx
    for j=1:Ny
        % Define the coordinates and velocities of grid block vertexes
        X0=(i-0.5)*hx;Y0=(j-0.5)*hy;
        Xux1=X0-hx/2; Yux1=Y0-hy/2; ux1=Vx1(i,j);
        Xuy1=X0-hx/2; Yuy1=Y0-hy/2; uy1=Vy1(i,j);
        Xux2=X0+hx/2; Yux2=Y0-hy/2; ux2=Vx2(i,j);
        Xuy2=X0+hx/2; Yuy2=Y0-hy/2; uy2=Vy2(i,j);
        Xux3=X0+hx/2; Yux3=Y0+hy/2; ux3=Vx3(i,j);
        Xuy3=X0+hx/2; Yuy3=Y0+hy/2; uy3=Vy3(i,j);
        Xux4=X0-hx/2; Yux4=Y0+hy/2; ux4=Vx4(i,j);
        Xuy4=X0-hx/2; Yuy4=Y0+hy/2; uy4=Vy4(i,j);
        % Determine the velocity coefficients,
        % vx = a1+b*x+c1*y+d2.*y^2-2*d2*x*y-d1*x^2;
        % vy = a2-b*y+c2*x-d1.*x^2+2*d1*x*y+d2*y^2;
        Co=[1, 0, Xux1, Yux1, 0, -Xux1^2,-2*Xux1*Yux1+Yux1^2;
            0, 1, -Yuy1, Xuy1, 2*Xuy1*Yuy1-Xuy1^2, Yuy1^2;

```

```

1,    0,    Xux2, Yux2,    0,    -Xux2^2,-2*Xux2*Yux2+Yux2^2;
0,    1,    -Yuy2,    0,    Xuy2,    2*Xuy2*Yuy2-Xuy2^2,    Yuy2^2;
1,    0,    Xux3, Yux3,    0,    -Xux3^2,-2*Xux3*Yux3+Yux3^2;
0,    1,    -Yuy3,    0,    Xuy3,    2*Xuy3*Yuy3-Xuy3^2,    Yuy3^2;
1,    0,    Xux4, Yux4,    0,    -Xux4^2,-2*Xux4*Yux4+Yux4^2;
0,    1,    -Yuy4,    0,    Xuy4,    2*Xuy4*Yuy4-Xuy4^2,    Yuy4^2];
W=[ux1;uy1;ux2;uy2;ux3;uy3;ux4;uy4]; Factor(i,j,:)=Co\W;

end

end

a1=Factor(:, :, 1); a2=Factor(:, :, 2); b=Factor(:, :, 3); c1=Factor(:, :, 4); c2=Factor(:, :, 5);
d_1=Factor(:, :, 6); d_2=Factor(:, :, 7);

```

Code J.3-3: Interpret the velocity at grid block vertexes

```

function [V2]=Cubic_2d_velocity(Grid,dp,Muo,phi)
% Interpret the velocity at 2D grid block vertexes
Nx=Grid.Nx;Ny=Grid.Ny;hx=Grid.hx;hy=Grid.hy;
% Calculate the upscaled permeability using harmonic averaging
L = Grid.K.^(-1);kx = zeros(Nx+1,Ny);ky = zeros(Nx,Ny+1);
kx(2:Nx,:) = 2./(L(1,1:Nx-1, :,1)+L(1,2:Nx, :,1));
ky(:,2:Ny) = 2./(L(2, :, 1: Ny-1,1)+L(2, :, 2:Ny,1));
% Determine the x-directional pressure derivative slop (DSx) along y-axis
DSx=abs(dp.x(2:Nx,1:end-1)-dp.x(2:Nx,2:end));
% Find where the DSx is minimized
[sx, index]=sort(DSx,2);x=index(:,1);dp2.x=zeros(Nx+1,Ny+1);
% Determine the distribution of x-directional pressure derivation along
% y-axis using piece-wise linear function.
for i=1:Nx-1
    J=x(i,1); I=i+1;dp2.x(I,J+1)=(dp.x(I,J)+dp.x(I,J+1))/2;
    for j=J:-1:1
        dp2.x(I,j)=2*dp2.x(I,j+1)-dp.x(I,j);
    end
    for j=J+1:1:Ny
        dp2.x(I,j)=2*dp2.x(I,j-1)-dp.x(I,j);
    end
end
end
end

```

```

% Determine the y-directional pressure derivative slope (DSy) along x-axis
DSy=abs(dp.y(1:end-1,2:Ny)-dp.y(2:end,2:Ny));
% Find where the DSy is minimized
[sy, indey]=sort(DSy); y=indey(1,:); dp2.y=zeros(Nx+1,Ny+1);
% Determine the distribution of y-directional pressure derivation along
% x-axis using piece-wise linear function.
for i=1:Ny-1
    Jy=y(i); I=i+1; dp2.y(Jy+1,I)=(dp.y(Jy,I)+dp.y(Jy+1,I))/2;
    for j=Jy:-1:1
        dpy=dp.y(j,I); dp2.y(j,I)=2*dpy-dp2.y(j+1,I);
    end
    for j=Jy+1:1:Nx
        dpy=dp.y(j,I); dp2.y(j+1,I)=2*dpy-dp2.y(j,I);
    end
end
% Calculate the velocities at grid block vertexes using Darcy's law
V2.x=zeros(Nx+1,2*Ny); V2.y=zeros(2*Nx,Ny+1);
V2.x(:,1:2:end)=-dp2.x(:,1:Ny).*kx(:,:)/phi./Mu0;
V2.x(:,2:2:end)=-dp2.x(:,2:Ny+1).*kx(:,:)/phi./Mu0;
V2.y(1:2:end,:)= -dp2.y(1:Nx,:).*ky(:,:)/phi./Mu0;
V2.y(2:2:end,:)= -dp2.y(2:Nx+1,:).*ky(:,:)/phi./Mu0;

```

J.4 The 3D Cubic streamline tracing method

Code J.4-1: Streamline tracing using the 3D Cubic method

```

clear all
% Tracing streamlines using the Cubic 3D method
% Define the number of grid blocks and the grid block length
Grid.Nx=20; Grid.hx=100/(Grid.Nx); Grid.Ny=Grid.Nx; Grid.hy=100/(Grid.Ny);
Grid.Nz=Grid.Nx; Grid.hz=100/(Grid.Nz); hx=Grid.hx; hy=Grid.hy; hz=Grid.hz;
Grid.K=ones(3,Grid.Nx,Grid.Ny,Grid.Nz); bone=Grid.Nx/10; bonez=Grid.Nz/10;
% Muo, oil viscosity; ao, the maximum oil relative permeability
Q=100; phi=0.3; Muo=1; ao=1; m=3; nn=15; pi=5;
% Determine the velocity coefficient in the Cubic 3D method
[alx, clx, c2x, b1, a2y, c1y, c3y, b2, a3z, c2z, c3z, b3, d1, d2, d3, d4, d5, d6]=...
    cubic_3d_factor(Grid,Q,Muo,pi,phi);

```

```

l=27;% Define the streamlines launching points and flow rate along each streamline
[q,La]=q_assign3d(a1x,c1x,c2x,b1,a2y,c1y,c3y,b2,a3z,c2z,c3z,b3,d1,d2,d3,d4,d5,d6,...
    Grid,l,bone,bonez,Q,phi);
l=numel(La(:,1)); nu=1;x1=La(nu,1); y1=La(nu,2); z1=La(nu,3); tof=zeros(1,1);
while (nu<=l)
    % Locate the launching point into grid blocks
    NBx=fix((roundn(x1/hx,nn)))+1;NBy=fix((roundn(y1/hy,nn)))+1;
    NBz=fix((roundn(z1/hz,nn)))+1;
    j=1;XX(j)=x1;YY(j)=y1;ZZ(j)=z1;dT(nu,j)=0;
    while (NBx<=(Grid.Nx-bone) || NBy<=(Grid.Ny-bone) || NBz<=(Grid.Nz-bone))
        % Define the boundaries of local grid block
        x_2=NBx.*hx;y_2=NBy.*hy;z_2=NBz.*hz;x_1=x_2-hx;y_1=y_2-hy;z_1=z_2-hz;
        A1x=a1x(NBx,NBy,NBz);C1x=c1x(NBx,NBy,NBz);C2x=c2x(NBx,NBy,NBz);
        B1=b1(NBx,NBy,NBz);A2y=a2y(NBx,NBy,NBz);C1y=c1y(NBx,NBy,NBz);
        C3y=c3y(NBx,NBy,NBz);B2=b2(NBx,NBy,NBz);A3z=a3z(NBx,NBy,NBz);
        C2z=c2z(NBx,NBy,NBz);C3z=c3z(NBx,NBy,NBz);B3=b3(NBx,NBy,NBz);
        D1=d1(NBx,NBy,NBz);D2=d2(NBx,NBy,NBz);D3=d3(NBx,NBy,NBz);
        D4=d4(NBx,NBy,NBz);D5=d5(NBx,NBy,NBz);D6=d6(NBx,NBy,NBz);
        % Calculate the entry directional velocities
        vx1=A1x+C1x*y1+C2x*z1+B1*x1+D1*x1.^2+D2.*y1.^2+2.*D3.*x1.*y1+...
            2.*D5.*x1.*z1-(D1+D2).*z1.^2;
        vy1=A2y+C1y*x1+C3y*z1+B2*y1+2.*D2*x1*y1+D3.*x1.^2+D4.*y1.^2+...
            2.*D6.*y1.*z1-(D3+D4).*z1.^2;
        vz1=A3z+C2z*x1+C3z*y1+B3*z1+D5.*x1.^2+D6.*y1.^2-2.*(D1+D2).*x1.*z1-...
            2.*(D3+D4).*y1.*z1-(D5+D6).*z1.^2;
        Si=0;
        % Determine the intersection points with the sub-cell boundaries using
        % RK-4th method
        if vx1>1e-5;
            F = @(x,y,z) (A2y+C1y*x+C3y*z+B2*y+2.*D2*x*y+D3.*x.^2+D4.*y.^2+...
                2.*D6.*y.*z-(D3+D4).*z.^2)./(A1x+C1x*y+C2x*z+B1*x+D1*x.^2+...
                D2.*y.^2+2.*D3.*x.*y+2.*D5.*x.*z-(D1+D2).*z.^2);
            G = @(x,y,z) (A3z+C2z*x+C3z*y+B3*z+D5.*x.^2+D6.*y.^2-2.*(D1+D2).*x.*z-...
                2.*(D3+D4).*y.*z-(D5+D6).*z.^2)./(A1x+C1x*y+C2x*z+B1*x+D1*x.^2+...
                D2.*y.^2+2.*D3.*x.*y+2.*D5.*x.*z-(D1+D2).*z.^2);

```

```

[x_x,y_x,z_x]=RK4(F,G,x1,y1,z1,x_2,m);y_x=round(y_x,5);z_x=round(z_x,5);
if y_x(end)<=y_2 && z_x(end)<=z_2 && z_x(end)>=z1 && y_x(end)>=y1
    Si=1;
end
end
if vy1>1e-5 && Si==0
    F = @(y,x,z) (A1x+C1x*y+C2x*z+B1*x+D1*x.^2+D2.*y.^2+2.*D3.*x.*y+...
        2.*D5.*x.*z-(D1+D2).*z.^2)./(A2y+C1y*x+C3y*z+B2*y+2.*D2*x*y+...
        D3.*x.^2+D4.*y.^2+2.*D6.*y.*z-(D3+D4).*z.^2);
    G = @(y,x,z) (A3z+C2z*x+C3z*y+B3*z+D5.*x.^2+D6.*y.^2-2.*(D1+D2).*x.*z-...
        2.*(D3+D4).*y.*z-(D5+D6).*z.^2)./(A2y+C1y*x+C3y*z+B2*y+2.*D2*x*y+...
        D3.*x.^2+D4.*y.^2+2.*D6.*y.*z-(D3+D4).*z.^2);
    [y_y,x_y,z_y]=RK4(F,G,y1,x1,z1,y_2,m);x_y=round(x_y,5);z_y=round(z_y,5);
    if x_y(end)<=x_2 && z_y(end)<=z_2 && z_y(end)>=z1 && x_y(end)>=x1
        Si=2;
    end
end
if Si==0
    F = @(z,x,y) (A1x+C1x*y+C2x*z+B1*x+D1*x.^2+D2.*y.^2+2.*D3.*x.*y+...
        2.*D5.*x.*z-(D1+D2).*z.^2)./(A3z+C2z*x+C3z*y+B3*z+D5.*x.^2+D6.*y.^2-...
        2.*(D1+D2).*x.*z-2.*(D3+D4).*y.*z-(D5+D6).*z.^2);
    G = @(z,x,y) (A2y+C1y*x+C3y*z+B2*y+2.*D2*x*y+D3.*x.^2+D4.*y.^2+...
        2.*D6.*y.*z-(D3+D4).*z.^2)./(A3z+C2z*x+C3z*y+B3*z+D5.*x.^2+D6.*y.^2-...
        2.*(D1+D2).*x.*z-2.*(D3+D4).*y.*z-(D5+D6).*z.^2);
    [z_z,y_z,x_z]=RK4(F,G,z1,y1,x1,z_2,m);y_z=round(y_z,5);x_z=round(x_z,5);Si=3;
end
end
if Si==1
    x2=x_x(end);y2=y_x(end);z2=z_x(end);
elseif Si==2
    x2=x_y(end);y2=y_y(end);z2=z_y(end);
elseif Si==3
    x2=x_z(end);y2=y_z(end);z2=z_z(end);
end
end
XX(j+1)=x2;YY(j+1)=y2;ZZ(j+1)=z2;
% Calculate the exit directional-velocities

```

```

vx2=A1x+C1x*y2+C2x*z2+B1*x2+D1*x2.^2+D2.*y2.^2+2.*D3.*x2.*y2+...
    2.*D5.*x2.*z2-(D1+D2).*z2.^2;
vy2=A2y+C1y*x2+C3y*z2+B2*y2+2.*D2*x2*y2+D3.*x2.^2+D4.*y2.^2+...
    2.*D6.*y2.*z2-(D3+D4).*z2.^2;
vz2=A3z+C2z*x2+C3z*y2+B3*z2+D5.*x2.^2+D6.*y2.^2-2.*(D1+D2).*x2.*z2-...
    2.*(D3+D4).*y2.*z2-(D5+D6).*z2.^2;
kx=Grid.K(1,NBx,NBy,NBz);ky=Grid.K(2,NBx,NBy,NBz);kz=Grid.K(3,NBx,NBy,NBz);
% Calculate the total velocity ut and directional permeability k
if j==1
    ut(nu,1)=(vx1^2+vy1^2+vz1^2)^0.5;
    k(nu,1)=kx.*(vx1/ut(nu,1)).^2+ky.*(vy1./ut(nu,1)).^2+...
        kz.*(vz1/ut(nu,1)).^2;
end
ut(nu,j+1)=(vx2^2+vy2^2+vz2^2)^0.5;
k(nu,j+1)=kx.*(vx2/ut(nu,j+1)).^2+ky.*(vy2./ut(nu,j+1)).^2+...
    kz.*(vz2/ut(nu,j+1)).^2;
% Determine the incremental time-of-flight
if vx1*vz2>0
    Vx=[vx1,vz2]; X=[x1,x2];dTOF=trapz(X,1./Vx);
else
    if vy1*vy2>0
        Vy=[vy1,vy2]; Y=[y1,y2];dTOF=trapz(Y,1./Vy);
    else
        Vz=[vz1,vz2]; Z=[z1,z2];dTOF=trapz(Z,1./Vz);
    end
end
end
dT(nu,j+1)=dT(nu,j)+dTOF;
% Use the exit point as the entry point for the next grid block
j=j+1;x1=x2;y1=y2;z1=z2;
NBx=fix((roundn(x1/hx,nn))+1);NBy=fix((roundn(y1/hy,nn))+1);
NBz=fix((roundn(z1/hz,nn))+1);
end
plot3(XX(1:j),YY(1:j),ZZ(1:j),'k','LineWidth',1)
hold on
nu=nu+1;

```

```

    if nu<=1
        x1=La(nu,1); y1=La(nu,2); z1=La(nu,3);
    end
end
% Determine the differential pressure of each streamtube
[DP]=p_assign(Muo,ao,k,l,ut,dT,phi);

```

Code J.4-2: Determine the velocity coefficient in the 3D Cubic method

```

function [a1x,c1x,c2x,b1,a2y,c1y,c3y,b2,a3z,c2z,c3z,b3,d1,d2,d3,d4,d5,d6]=...
    cubic_3d_factor(Grid,Q,Mu,pi,phi)
% Determine the velocity coefficients in the Cubic 3D method
% vx1=a1x+c1x*y+c2x*z+b1*x+d1*x^2+d2*y^2+2*d3*x*y+2*d5*x*z-(d1+d2)*z^2;
% vy1=a2y+c1y*x+c3y*z+b2*y+2*d2*x*y+d3*x^2+d4*y^2+2*d6*y*z-(d3+d4)*z^2;
% vz1=a3z+c2z*x+c3z*y+b3*z+d5*x^2+d6*y^2-2*(d1+d2)*x*z-2*(d3+d4)*y*z-(d5+d6)*z^2;
hx=Grid.hx;hy=Grid.hy;hz=Grid.hz;Nx=Grid.Nx; Ny=Grid.Ny; Nz=Grid.Nz;
% Determine the pressure derivatives using a finite difference approach
[P,U,dp]=FD_pressure(Grid,Q,Mu,pi);
% Calculate the upscaled permeability using harmonic averaging
L = Grid.K.^(-1);kx = zeros(Nx+1,Ny,Nz);ky = zeros(Nx,Ny+1,Nz);kz = zeros(Nx,Ny,Nz+1);
kx(2:Nx, :, :) = 2./(L(1,1:Nx-1, :, :)+L(1,2:Nx, :, :));
ky(:, 2:Ny, :) = 2./(L(2, :, 1:Ny-1, :)+L(2, :, 2:Ny, :));
kz(:, :, 2:Nz) = 2./(L(3, :, :, 1:Nz-1)+L(3, :, :, 2:Nz));
% Determine the distribution of x-directional pressure derivative along y-axis
DS=abs(dp.x(:,1:end-1,:)-dp.x(:,2:end,:));[sxy,indexy]=sort(DS,2);xy=indexy(:,1,:);
for i=1:Nx+1
    for k=1:Nz
        J=xy(i,1,k);I=i;dpxy(I,J+1,k)=(dp.x(I,J,k)+dp.x(I,J+1,k))/2;
        for j=J:-1:1
            dx=dp.x(I,j,k);dpxy(I,j,k)=2*dx-dpxy(I,j+1,k);
        end
        for j=J+1:1:Ny
            dx=dp.x(I,j,k);dpxy(I,j+1,k)=2*dx-dpxy(I,j,k);
        end
    end
end
end

```

```

% Determine the distribution of x-directional pressure derivative along z-axis
DS=abs(dp.x(:, :, 1:end-1)-dp.x(:, :, 2:end)); [sxz, indexz]=sort(DS,3); xz=indexz(:, :, 1);
for i=1:Nx+1
    for j=1:Ny
        K=xz(i, j); I=i; dpxz(I, j, K+1)=(dp.x(I, j, K)+dp.x(I, j, K+1))/2;

        for k=K:-1:1
            dx=dp.x(I, j, k); dpxz(I, j, k)=2*dx-dpxz(I, j, k+1);
        end
        for k=K+1:1:Nz
            dx=dp.x(I, j, k); dpxz(I, j, k+1)=2*dx-dpxz(I, j, k);
        end
    end
end
end
% Determine the distribution of y-directional pressure derivative along z-axis
DS=abs(dp.y(:, :, 1:end-1)-dp.y(:, :, 2:end)); [sz, indeyz]=sort(DS,3); yz=indeyz(:, :, 1);
for j=1:Ny+1
    for i=1:Nx
        K=yz(i, j); J=j; dpyz(i, J, K+1)=(dp.y(i, J, K)+dp.y(i, J, K+1))/2;
        for k=K:-1:1
            dy=dp.y(i, J, k); dpyz(i, J, k)=2*dy-dpyz(i, J, k+1);
        end
        for k=K+1:1:Nz
            dy=dp.y(i, J, k); dpyz(i, J, k+1)=2*dy-dpyz(i, J, k);
        end
    end
end
end
% Determine the distribution of y-directional pressure derivative along x-axis
DS=abs(dp.y(1:end-1, :, :) - dp.y(2:end, :, :)); [syx, indeyx]=sort(DS,1); yx=indeyx(1, :, :);
for j=1:Ny+1
    for k=1:Nz
        I=yx(1, j, k); J=j; dpyx(I+1, J, k)=(dp.y(I, J, k)+dp.y(I+1, J, k))/2;
        for i=I:-1:1
            dy=dp.y(i, J, k); dpyx(i, J, k)=2*dy-dpyx(i+1, J, k);
        end
    end
end

```

```

    for i=I+1:1:Nx
        dy=dp.y(i,J,k); dpyx(i+1,J,k)=2*dy-dpyx(i,J,k);
    end
end
end
end
% Determine the distribution of z-directional pressure derivative along x-axis
DS=abs(dp.z(1:end-1,:,:) - dp.z(2:end,:,:)); [sx, indezx]=sort(DS,1); zx=indezx(1,:,:);
for k=1:Nz+1
    for j=1:Ny
        I=zx(1,j,k); K=k; dpzx(I+1,j,K)=(dp.z(I,j,K)+dp.z(I+1,j,K))/2;
        for i=I:-1:1
            dz=dp.z(i,j,K); dpzx(i,j,K)=2*dz-dpxz(i+1,j,K);
        end
        for i=I+1:1:Nx
            dz=dp.z(i,j,K); dpzx(i+1,j,K)=2*dz-dpxz(i,j,K);
        end
    end
end
end
end
% Determine the distribution of z-directional pressure derivative along y-axis
DS=abs(dp.z(:,1:end-1,:) - dp.z(:,2:end,:)); [sy, indexy]=sort(DS,2); zy=indexy(:,1,:);
for k=1:Nz+1
    for i=1:Nx
        J=zy(i,1,k); K=k; dpzy(i,J+1,K)=(dp.z(i,J,K)+dp.z(i,J+1,K))/2;
        for j=J:-1:1
            dz=dp.z(i,j,K); dpzy(i,j,K)=2*dz-dpzy(i,j+1,K);
        end
        for j=J+1:1:Ny
            dz=dp.z(i,j,K); dpzy(i,j+1,K)=2*dz-dpzy(i,j,K);
        end
    end
end
end
end
% Calculate the grid block interface velocities using Darcy's law
Vx.y=zeros(Nx+1,2*Ny,Nz); Vx.z=zeros(Nx+1,Ny,2*Nz);
Vx.y(:,1:2:end,:)= -dpxy(:,1:Ny,:).*kx(:, :, :)/phi./Mu;
Vx.y(:,2:2:end,:)= -dpxy(:,2:Ny+1,:).*kx(:, :, :)/phi./Mu;

```

```

Vx.z (: , : , 1:2:end)=-dpzx (: , : , 1:Nz).*kx (: , : , :)/phi./Mu;
Vx.z (: , : , 2:2:end)=-dpzx (: , : , 2:Nz+1).*kx (: , : , :)/phi./Mu;
Vy.x=zeros (2*Nx,Ny+1,Nz);Vy.z=zeros (Nx,Ny+1,2*Nz);
Vy.x (1:2:end, :, :)= -dpyx (1:Nx, :, :).*ky (: , : , :)/phi./Mu;
Vy.x (2:2:end, :, :)= -dpyx (2:Nx+1, :, :).*ky (: , : , :)/phi./Mu;
Vy.z (: , : , 1:2:end)=-dpyz (: , : , 1:Nz).*ky (: , : , :)/phi./Mu;
Vy.z (: , : , 2:2:end)=-dpyz (: , : , 2:Nz+1).*ky (: , : , :)/phi./Mu;
Vz.x=zeros (2*Nx,Ny,Nz+1);Vz.y=zeros (Nx,2*Ny,Nz+1);
Vz.x (1:2:end, :, :)= -dpzx (1:Nx, :, :).*kz (: , : , :)/phi./Mu;
Vz.x (2:2:end, :, :)= -dpzx (2:Nx+1, :, :).*kz (: , : , :)/phi./Mu;
Vz.y (: , 1:2:end, :)= -dpzy (: , 1:Ny, :).*kz (: , : , :)/phi./Mu;
Vz.y (: , 2:2:end, :)= -dpzy (: , 2:Ny+1, :).*kz (: , : , :)/phi./Mu;
Vx1=Vx.y (1:end-1, 1:2:end-1, :);Vx2=Vx.y (1:end-1, 2:2:end, :);
Vx3=Vx.z (1:end-1, :, 1:2:end-1);Vx4=Vx.z (1:end-1, :, 2:2:end);
Vx5=Vx.y (2:end, 1:2:end-1, :);Vx6=Vx.y (2:end, 2:2:end, :);
Vx7=Vx.z (2:end, :, 1:2:end-1);Vx8=Vx.z (2:end, :, 2:2:end);
Vy1=Vy.x (1:2:end-1, 1:end-1, :);Vy2=Vy.x (2:2:end, 1:end-1, :);
Vy3=Vy.z (: , 1:end-1, 1:2:end-1);Vy4=Vy.z (: , 1:end-1, 2:2:end);
Vy5=Vy.x (1:2:end-1, 2:end, :);Vy6=Vy.x (2:2:end, 2:end, :);
Vy7=Vy.z (: , 2:end, 1:2:end-1);Vy8=Vy.z (: , 2:end, 2:2:end);
Vz1=Vz.x (1:2:end-1, :, 1:end-1);Vz2=Vz.x (2:2:end, :, 1:end-1);
Vz3=Vz.y (: , 1:2:end-1, 1:end-1);Vz4=Vz.y (: , 2:2:end, 1:end-1);
Vz5=Vz.x (1:2:end-1, :, 2:end);Vz6=Vz.x (2:2:end, :, 2:end);
Vz7=Vz.y (: , 1:2:end-1, 2:end);Vz8=Vz.y (: , 2:2:end, 2:end);
Factor=zeros (Nx,Ny,Nz,17);
for i=1:Nx
    for j=1:Ny
        for k=1:Nz
            % Define the grid block interface coordinates and velocities
            X0=(i-0.5)*Grid.hx;Y0=(j-0.5)*Grid.hy;Z0=(k-0.5)*Grid.hz;
            Xx1=X0-hx/2; Yx1=Y0-hy/2; Zx1=Z0; ux1=Vx1(i,j,k);
            Xx2=X0-hx/2; Yx2=Y0+hy/2; Zx2=Z0; ux2=Vx2(i,j,k);
            Xx3=X0-hx/2; Yx3=Y0; Zx3=Z0-hz/2; ux3=Vx3(i,j,k);
            Xx4=X0-hx/2; Yx4=Y0; Zx4=Z0+hz/2; ux4=Vx4(i,j,k);
            Xx5=X0+hx/2; Yx5=Y0-hy/2; Zx5=Z0; ux5=Vx5(i,j,k);

```

```

Xx6=X0+hx/2; Yx6=Y0+hy/2; Zx6=Z0; ux6=Vx6(i,j,k);
Xx7=X0+hx/2; Yx7=Y0; Zx7=Z0-hz/2; ux7=Vx7(i,j,k);
Xx8=X0+hx/2; Yx8=Y0; Zx8=Z0+hz/2; ux8=Vx8(i,j,k);
Xy1=X0-hx/2; Yy1=Y0-hy/2; Zy1=Z0; uy1=Vy1(i,j,k);
Xy2=X0+hx/2; Yy2=Y0-hy/2; Zy2=Z0; uy2=Vy2(i,j,k);
Xy3=X0; Yy3=Y0-hy/2; Zy3=Z0-hz/2; uy3=Vy3(i,j,k);
Xy4=X0; Yy4=Y0-hy/2; Zy4=Z0+hz/2; uy4=Vy4(i,j,k);
Xy5=X0-hx/2; Yy5=Y0+hy/2; Zy5=Z0; uy5=Vy5(i,j,k);
Xy6=X0+hx/2; Yy6=Y0+hy/2; Zy6=Z0; uy6=Vy6(i,j,k);
Xy7=X0; Yy7=Y0+hy/2; Zy7=Z0-hz/2; uy7=Vy7(i,j,k);
Xy8=X0; Yy8=Y0+hy/2; Zy8=Z0+hz/2; uy8=Vy8(i,j,k);
Xz1=X0-hx/2; Yz1=Y0; Zz1=Z0-hz/2; uz1=Vz1(i,j,k);
Xz2=X0+hx/2; Yz2=Y0; Zz2=Z0-hz/2; uz2=Vz2(i,j,k);
Xz3=X0; Yz3=Y0-hy/2; Zz3=Z0-hz/2; uz3=Vz3(i,j,k);
Xz4=X0; Yz4=Y0+hy/2; Zz4=Z0-hz/2; uz4=Vz4(i,j,k);
Xz5=X0-hx/2; Yz5=Y0; Zz5=Z0+hz/2; uz5=Vz5(i,j,k);
Xz6=X0+hx/2; Yz6=Y0; Zz6=Z0+hz/2; uz6=Vz6(i,j,k);
Xz7=X0; Yz7=Y0-hy/2; Zz7=Z0+hz/2; uz7=Vz7(i,j,k);
Xz8=X0; Yz8=Y0+hy/2; Zz8=Z0+hz/2; uz8=Vz8(i,j,k);

% Determine the velocity coefficients by solving linear equations
% vx1=a1x+c1x*y+c2x*z+b1*x+d1*x^2+d2*y^2+2*d3*x*y+2*d5*x*z-(d1+d2)*z^2;
% vy1=a2y+c1y*x+c3y*z+b2*y+2*d2*x*y+d3*x^2+d4*y^2+2*d6*y*z-(d3+d4)*z^2;
% vz1=a3z+c2z*x+c3z*y+b3*z+d5*x^2+d6*y^2-2*(d1+d2)*x*z-2*(d3+d4)*y*z-(d5+d6)*z^2;
Co=[1, Yx1, Zx1, Xx1, 0, 0, 0, 0, 0, 0, 0, (Xx1^2-Zx1^2),
(Yx1^2-Zx1^2), 2.*Xx1.*Yx1, 0, 2.*Xx1.*Zx1, 0;
1, Yx2, Zx2, Xx2, 0, 0, 0, 0, 0, 0, 0, (Xx2^2-Zx2^2),
(Yx2^2-Zx2^2), 2.*Xx2.*Yx2, 0, 2.*Xx2.*Zx2, 0;
1, Yx3, Zx3, Xx3, 0, 0, 0, 0, 0, 0, 0, (Xx3^2-Zx3^2),
(Yx3^2-Zx3^2), 2.*Xx3.*Yx3, 0, 2.*Xx3.*Zx3, 0;
1, Yx4, Zx4, Xx4, 0, 0, 0, 0, 0, 0, 0, (Xx4^2-Zx4^2),
(Yx4^2-Zx4^2), 2.*Xx4.*Yx4, 0, 2.*Xx4.*Zx4, 0;
1, Yx5, Zx5, Xx5, 0, 0, 0, 0, 0, 0, 0, (Xx5^2-Zx5^2),
(Yx5^2-Zx5^2), 2.*Xx5.*Yx5, 0, 2.*Xx5.*Zx5, 0;
1, Yx6, Zx6, Xx6, 0, 0, 0, 0, 0, 0, 0, (Xx6^2-Zx6^2),
(Yx6^2-Zx6^2), 2.*Xx6.*Yx6, 0, 2.*Xx6.*Zx6, 0;

```

$$\begin{aligned}
& 1, \quad Yx7, \quad Zx7, \quad Xx7, \quad 0, \quad (Xx7^2-Zx7^2), \\
& (Yx7^2-Zx7^2), \quad 2.*Xx7.*Yx7, \quad 0, \quad 2.*Xx7.*Zx7, \quad 0; \\
& 1, \quad Yx8, \quad Zx8, \quad Xx8, \quad 0, \quad (Xx8^2-Zx8^2), \\
& (Yx8^2-Zx8^2), \quad 2.*Xx8.*Yx8, \quad 0, \quad 2.*Xx8.*Zx8, \quad 0; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy1, \quad Zy1, \quad Yy1, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy1.*Yy1, \\
& (Xy1^2-Zy1^2), \quad (Yy1^2-Zy1^2), \quad 0, \quad 2.*Yy1.*Zy1; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy2, \quad Zy2, \quad Yy2, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy2.*Yy2, \\
& (Xy2^2-Zy2^2), \quad (Yy2^2-Zy2^2), \quad 0, \quad 2.*Yy2.*Zy2; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy3, \quad Zy3, \quad Yy3, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy3.*Yy3, \\
& (Xy3^2-Zy3^2), \quad (Yy3^2-Zy3^2), \quad 0, \quad 2.*Yy3.*Zy3; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy4, \quad Zy4, \quad Yy4, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy4.*Yy4, \\
& (Xy4^2-Zy4^2), \quad (Yy4^2-Zy4^2), \quad 0, \quad 2.*Yy4.*Zy4; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy5, \quad Zy5, \quad Yy5, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy5.*Yy5, \\
& (Xy5^2-Zy5^2), \quad (Yy5^2-Zy5^2), \quad 0, \quad 2.*Yy5.*Zy5; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy6, \quad Zy6, \quad Yy6, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy6.*Yy6, \\
& (Xy6^2-Zy6^2), \quad (Yy6^2-Zy6^2), \quad 0, \quad 2.*Yy6.*Zy6; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy7, \quad Zy7, \quad Yy7, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy7.*Yy7, \\
& (Xy7^2-Zy7^2), \quad (Yy7^2-Zy7^2), \quad 0, \quad 2.*Yy7.*Zy7; \\
& 0, \quad 0, \quad 0, \quad 0, \quad 1, \quad Xy8, \quad Zy8, \quad Yy8, \quad 0, \quad 0, \quad 0, \quad 0, \quad 2.*Xy8.*Yy8, \\
& (Xy8^2-Zy8^2), \quad (Yy8^2-Zy8^2), \quad 0, \quad 2.*Yy8.*Zy8; \\
& 0, \quad 0, \quad 0, \quad -Zz1, \quad 0, \quad 0, \quad 0, \quad -Zz1, \quad 1, \quad Xz1, \quad Yz1, \quad -2.*Xz1.*Zz1, \\
& -2.*Xz1.*Zz1, \quad -2.*Yz1.*Zz1, \quad -2.*Yz1.*Zz1, \quad (Xz1^2-Zz1^2), \quad (Yz1^2-Zz1^2); \\
& 0, \quad 0, \quad 0, \quad -Zz2, \quad 0, \quad 0, \quad 0, \quad -Zz2, \quad 1, \quad Xz2, \quad Yz2, \quad -2.*Xz2.*Zz2, \\
& -2.*Xz2.*Zz2, \quad -2.*Yz2.*Zz2, \quad -2.*Yz2.*Zz2, \quad (Xz2^2-Zz2^2), \quad (Yz2^2-Zz2^2); \\
& 0, \quad 0, \quad 0, \quad -Zz3, \quad 0, \quad 0, \quad 0, \quad -Zz3, \quad 1, \quad Xz3, \quad Yz3, \quad -2.*Xz3.*Zz3, \\
& -2.*Xz3.*Zz3, \quad -2.*Yz3.*Zz3, \quad -2.*Yz3.*Zz3, \quad (Xz3^2-Zz3^2), \quad (Yz3^2-Zz3^2); \\
& 0, \quad 0, \quad 0, \quad -Zz4, \quad 0, \quad 0, \quad 0, \quad -Zz4, \quad 1, \quad Xz4, \quad Yz4, \quad -2.*Xz4.*Zz4, \\
& -2.*Xz4.*Zz4, \quad -2.*Yz4.*Zz4, \quad -2.*Yz4.*Zz4, \quad (Xz4^2-Zz4^2), \quad (Yz4^2-Zz4^2); \\
& 0, \quad 0, \quad 0, \quad -Zz5, \quad 0, \quad 0, \quad 0, \quad -Zz5, \quad 1, \quad Xz5, \quad Yz5, \quad -2.*Xz5.*Zz5, \\
& -2.*Xz5.*Zz5, \quad -2.*Yz5.*Zz5, \quad -2.*Yz5.*Zz5, \quad (Xz5^2-Zz5^2), \quad (Yz5^2-Zz5^2); \\
& 0, \quad 0, \quad 0, \quad -Zz6, \quad 0, \quad 0, \quad 0, \quad -Zz6, \quad 1, \quad Xz6, \quad Yz6, \quad -2.*Xz6.*Zz6, \\
& -2.*Xz6.*Zz6, \quad -2.*Yz6.*Zz6, \quad -2.*Yz6.*Zz6, \quad (Xz6^2-Zz6^2), \quad (Yz6^2-Zz6^2); \\
& 0, \quad 0, \quad 0, \quad -Zz7, \quad 0, \quad 0, \quad 0, \quad -Zz7, \quad 1, \quad Xz7, \quad Yz7, \quad -2.*Xz7.*Zz7, \\
& -2.*Xz7.*Zz7, \quad -2.*Yz7.*Zz7, \quad -2.*Yz7.*Zz7, \quad (Xz7^2-Zz7^2), \quad (Yz7^2-Zz7^2);
\end{aligned}$$

```

0, 0, 0, -Zz8, 0, 0, 0, -Zz8, 1, Xz8, Yz8, -2.*Xz8.*Zz8,
-2.*Xz8.*Zz8, -2.*Yz8.*Zz8, -2.*Yz8.*Zz8, (Xz8^2-Zz8^2), (Yz8^2-Zz8^2)];
W=[ux1; ux2; ux3; ux4; ux5; ux6; ux7; ux8; uy1; uy2; uy3; uy4; uy5; uy6; uy7; uy8; uz1; uz2; uz3; ...
uz4; uz5; uz6; uz7; uz8];
Factor(i, j, k, :) = Co\W;
end
end
end
a1x=Factor(:, :, :, 1); c1x=Factor(:, :, :, 2); c2x=Factor(:, :, :, 3); b1=Factor(:, :, :, 4);
a2y=Factor(:, :, :, 5); c1y=Factor(:, :, :, 6); c3y=Factor(:, :, :, 7); b2=Factor(:, :, :, 8);
a3z=Factor(:, :, :, 9); c2z=Factor(:, :, :, 10); c3z=Factor(:, :, :, 11); b3=-b1-b2;
d1=Factor(:, :, :, 12); d2=Factor(:, :, :, 13); d3=Factor(:, :, :, 14);
d4=Factor(:, :, :, 15); d5=Factor(:, :, :, 16); d6=Factor(:, :, :, 17); nn=15;
a1x=roundn(a1x, nn); c1x=roundn(c1x, nn); c2x=roundn(c2x, nn); b1=roundn(b1, nn);
a2y=roundn(a2y, nn); c1y=roundn(c1y, nn); c3y=roundn(c3y, nn); b2=roundn(b2, nn);
a3z=roundn(a3z, nn); c2z=roundn(c2z, nn); c3z=roundn(c3z, nn); b3=roundn(b3, nn);
d1=roundn(d1, nn); d2=roundn(d2, nn); d3=roundn(d3, nn);
d4=roundn(d4, nn); d5=roundn(d5, nn); d6=roundn(d6, nn);

```

J.5 The semi-analytical Riemann solver

Code J.5-1: The semi-analytical Riemann solver

```

% A Semi-analytical Riemann solver
% Define the relative permeability relations of oil and water
Muw=1; Muo=5; % Muw, water viscosity;
Swc=0.2; Sor=0.3; % Swc, connate water saturation; Sor, residual oil saturation
no=2; nw=2; % no, nw, exponent in corey model for oil and water, respectively
ao=1; aw=0.1; % ao, aw, maximum relative permeability for oil and water, respectively
% kro, krw, relative permeability of oil and water, respectively
syms S
kro= ao*((1-S-Sor)/(1-Sor-Swc))^no; krw= aw*((S-Swc)/(1-Sor-Swc))^nw;
lamda= (kro./Muo+krw./Muw); % lamda, total mobility
f= (krw./Muw)/(krw./Muw+kro./Muo); % fractional flow of water
df= simplify(diff(f,S)); % df/ds
ff= diff((S-Swc)/f,S); % d((s-Swc)/f)/ds
FF=double(subs(ff, Swc+0.01:0.001:1-Sor));

```

```

S_u = interp1(FF, Swc+0.01:0.001:1-Sor, 0); % Su, the shock front water saturation
% sf=diff((S-Swc)/f, S); SF=double(subs(sf, Swc+0.01:0.001:1-Sor));
% Su = interp1(SF, Swc+0.01:0.001:1-Sor, 0);
rs=S_u:0.001:1-Sor; % rs, rarefaction wave saturation
DF=double(subs(df, rs)); dfs=DF(1); % DF, rarefaction wave df/ds; dfs, shock front df/ds;
nu=1; Time=800; % Time, simulation time
while nu<=l
    % solve the Riemann problem along individual streamlines
    num=sum(ut(nu,:) > 0);
    dn=1:num; % dn, discretization stage before the water breakthrough
    % u, total velocity; tof, time-of-flight; kd, directional permeability;
    % qr, reference flow rate
    u=ut(nu, dn); tof=dT(nu, dn); kd=k(nu, dn); qr=q(nu);
    i=1; j=2;
    % Qn, cumulative water injection volume; qn, flow rate; tn, time;
    % sn, saturation at the outlet of streamtube
    Qn(1)=0; qn(1)=qr; tn(1)=0; sn(1)=Swc; t=0;
    while j<=numel(dn) % Before the water breakthrough
        % Determine the cumulative water injection volume at discretization
        % stage j
        Qn(j)=tof(j).*qr./dfs;
        df_S=qr.*tof(2:j-1)/Qn(j); % df_S, f'(s) at rarefaction wave
        % ss, saturation profile of the rarefaction wave
        ss=[1-Sor, interp1(DF, rs, df_S), S_u];
        % lamda_b, total mobility of the rarefaction wave
        lamda_b=double(subs(lamda, ss)).*kd(1:j);
        % R_b, Flow resistance of the rarefaction wave
        R_b=trapz(tof(1:j), u(1:j).^2.*phi./lamda_b);
        % lamda_a, total mobility of the unsupt area
        lamda_a=double(subs(lamda, Swc)).*kd(j:end);
        if j==numel(dn);
            R_a=0; % at water breakthrough, flow resistance of the unsupt area = 0
        else
            % flow resistance of the unsupt area
            R_a=trapz(tof(j:end), u(j:end).^2.*phi./lamda_a);

```

```

end
qn(j)=(DP(nu).*qr)./(R_b+R_a);% Calcualte the flow rate qn
dt=(Qn(j)-Qn(j-1))./(qn(j)+qn(j-1)).*2;% Calcualte the incremental time
tn(j)=tn(j-1)+dt;t=tn(end);% Calcualte the true time
% sn, the water saturation at the outlet of the streamtube
if tof(j)==max(tof)
    sn(j)=S_;
else
    sn(j)=Swc;
end
if tn(end)>=Time
    break
end
j=j+1;i=i+1;
clear ('ss','lamda_a','lamda_b');
end
Dts=Time/50;% Dts, an approximated incremental time
while t<Time % After the water breakthrough
Qn(j)=Qn(j-1)+qn(j-1)*Dts;
df_S=(qr.*tof)./Qn(j);% df_S, f'(s) at rarefaction wave
% ss, saturation profile of the rarefaction wave
ss=[1-Sor,interp1(DF,rs,df_S(2:end))];
lamda_b=double(subs(lamda,ss)).*kd;% total mobility of the rarefaction wave
R_b=trapz(tof,u.^2.*phi./lamda_b);% Flow resistance of the rarefaction wave
qn(j)=(DP(nu).*qr)./R_b;% Calcualte the flow rate qn
dt=(Qn(j)-Qn(j-1))./(qn(j)+qn(j-1)).*2;% Calcualte the incremental time
tn(j)=tn(j-1)+dt;t=tn(end);% Calcualte the true time
sn(j)=ss(end);% sn, the water saturation at the outlet of the streamtube
j=j+1;
end
plot(qn,tn)
hold on
t_n(nu,1:numel(tn))=tn;s_n(nu,1:numel(tn))=sn;q_n(nu,1:numel(tn))=qn;
clear ('u','k','tn','sn','qn','R_a','R_b');
nu=nu+1;

```

```

end
% Integrate the results from all streamlines, plot the phase flow rate profile
[TT,qw,qo]=phase_flow_rate(q_n,t_n,s_n,l,f,Time);
plot(TT,qw,'r',TT,qo,'r—')

```

Code J.5-2: Determine the phase flow rate profile

```

function [TT,qw,qo]=phase_flow_rate(q_n,t_n,s_n,l,f,Time)
% Integrate the results from all streamlines, plot the phase flow rate profile
nx=Time/80;maxT=Time;Tt=1:nx:maxT;% Discretization stage of time
nu=1;
while nu<=l
% Interpret the results from individual streamlines
q=q_n(nu,:);qq=q(q>0);m=numel(qq);tt=t_n(nu,1:m);s=s_n(nu,1:m);
qt(nu,:)=interp1(tt,qq,Tt);% Interpret the flow rate along individual streamlines
% Interpret the saturation at the outlet of streamtube along individual streamlines
S(nu,:)=interp1(tt,s,Tt);
nu=nu+1;clear ('q');
end
% Determine the water fractional flow at the outlet of streamtube
fw=double(subs(f,S));
% Interpret the oil and water phase flow rate as a function of time
qw=sum(fw.*qt);qo=sum((1-fw).*qt);qw=qw';qo=qo';TT=Tt;TT=TT';

```

Code J.5-3: Determine the initial differential pressure along each streamline

```

function [DP]=p_assign(Muo,ao,k,l,ut,dT,phi)
% Determine the constant differential pressure (DP) of each streamline during
% single phase flow period
nu=1;
while nu<=l
num=sum(ut(nu,:)>0);u=ut(nu,1:num);k_=k(nu,1:num);
tof=dT(nu,1:num);lamda=ao.*k_/Muo;
DP(nu)= trapz(tof,u.^2.*phi./lamda);nu=nu+1;
end

```

Code J.5-4: Determine the reference flow rate along each streamline

```

function [q]=q_assign2d(a1 , a2 , b , c1 , c2 , d1 , d2 , Grid , l , Q, LX, LY, phi)
nu=1;
while (nu<=l)
    if nu~=1 && nu~=l
        dx1=abs((LX(nu)-LX(nu-1)))/2; dx2=abs(LX(nu+1)-LX(nu))/2;
        dy1=abs((LY(nu)-LY(nu-1)))/2; dy2=abs(LY(nu+1)-LY(nu))/2;
    elseif nu==1
        dx2=abs(LX(nu+1)-LX(nu))/2; dx1=LX(nu)*(dx2>0);
        dy2=abs(LY(nu+1)-LY(nu))/2; dy1=LY(nu)*(dy2>0);
    else
        dx1=abs((LX(nu)-LX(nu-1)))/2; dx2=LX(nu)*(dx1>0);
        dy1=abs((LY(nu)-LY(nu-1)))/2; dy2=LY(nu)*(dy1>0);
    end
    x=LX(nu); y=LY(nu); NBx=fix(x/Grid.hx)+1; NBy=fix(y/Grid.hy)+1;
    A1=a1(NBx, NBy); A2=a2(NBx, NBy); B=b(NBx, NBy);
    C1=c1(NBx, NBy); C2=c2(NBx, NBy); D1=d1(NBx, NBy); D2=d2(NBx, NBy);
    vx=A1+B.*x+C1.*y+D2.*y.^2-2.*D2.*x.*y-D1.*x.^2;
    vy=A2-B.*y+C2.*x-D1.*x.^2+2.*D1.*x.*y+D2.*y.^2;
    q(nu)=(vx*(dy1+dy2)+vy*(dx1+dx2))*phi; nu=nu+1;
end
q=q.*Q./sum(q);% normalize q

```