

# DEVELOPMENT OF TECHNIQUES FOR GENERAL PURPOSE SIMULATORS

A DISSERTATION  
SUBMITTED TO THE DEPARTMENT OF PETROLEUM ENGINEERING  
AND THE COMMITTEE ON GRADUATE STUDIES  
OF STANFORD UNIVERSITY  
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
FOR THE DEGREE OF  
DOCTOR OF PHILOSOPHY

Hui Cao  
June 2002

@ Copyright by Hui Cao 2002  
All Rights Reserved

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

---

Dr. Khalid Aziz  
(Principle Advisor)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

---

Dr. Louis Durlofsky

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

---

Dr. Margot Gerritsen

Approved for the University Committee on Graduate Studies:



# Abstract

With the explosion of computational power and development of new and advanced simulation techniques, reservoir simulation has evolved into a mature technology. Today's simulator is more robust, more complex and more useful than what was available twenty years ago. Modern simulators incorporate a large variety of options and they can be used to answer important questions about reservoir management. Under these circumstances, a good design for the simulator is essential. So the first objective of this research is to develop a new General Purpose Research Simulator (GPRS), which can be used later by a large research group to preserve and enhance the work of several investigators over a period of time.

Compositional simulation is the major focus of this research. It is more complex than black-oil simulation due to the large number of components and various issues related to flash calculations. Because of this complexity, a large number of compositional models have been proposed. They use different equations, variables and implicit levels. The second objective of this research is to evaluate the performance of different compositional models. To do this in a consistent manner, a new General Formulation Approach is developed to derive any kind of model, and this approach is used in GPRS to implement various compositional models.

The IMPSAT (implicit pressure and saturations and explicit mole fractions) model has been discussed in the literature. Here we investigated this method in detail, including its characteristics, stability, construction and performance. The IMPSAT model is actually one of the models that the General Formulation Approach can generate. Besides this, a series of new IMPSAT based AIM (adaptive implicit) models are also proposed, and their performance is compared with the performance of the traditional AIM model which uses IMPES (implicit pressure and explicit saturations and mole fractions).

Various models and techniques of GPRS have been tested, including the black-oil model, compositional model, unstructured grid handling and multi-point flux approximation, and their results and performance are compared with existing simulators. The performance of different compositional models has been evaluated using a wide range of problems. We found that the natural variables (pressure, saturations and component mole fractions) are good for the fully implicit (FIM) model, while the overall

variables (pressure, overall component mole fractions) are good for the IMPES model. We have also demonstrated that the IMPSAT model and the IMPSAT based AIM models have very good performance for a wide range of problems, and they are more efficient than the traditional models. The IMPSAT model is generally over 50% faster than the IMPES model due to its improved stability. For problems that are not very hard, the IMPSAT model is cheaper than the FIM model since it reduces the number of unknowns that have to be solved implicitly. The IMPSAT based AIM models are more flexible, more stable and less expensive than the traditional AIM model. With properly tuned percentages of gridblocks for each formulation (IMPES, IMPSAT and FIM), the IMPES+IMPSAT+FIM model will always outperform the traditional IMPES+FIM model, and be the best model to use for all problems.

## Acknowledgements

I would like to thank my adviser Dr. Khalid Aziz for his support, encouragement and guidance during this work. I would also like to thank Dr. Louis Durlofsky and Dr. Margot Gerritsen for reading the dissertation and providing valuable comments. I also wish to thank the rest of the Petroleum Engineering Department for their supports and contributions to my academic achievements.

Special thanks are due to my wife, Jie Li, for her support and for giving me the joy of life, our newborn son. I would also like to thank my mother in law for helping with the care of our baby. My parents also deserve my gratitude for supporting and encouraging my education from the very beginning.

I also wish to thank ExxonMobile, ChevronTexaco and Schlumberger for summer jobs that provided valuable experience and motivation for my research. Finally, I like to thank the companies supporting the Reservoir Simulation Industrial Affiliate Program (SUPRI-B) at Stanford University for their financial contribution and interest in my research.

This research was partially supported by the United States Department of Energy through contract number DE-AC26-99BC15213. This support is greatly appreciated.





# Contents

<b>Abstract</b> .....	v
<b>Acknowledgments</b> .....	vii
<b>Table of Contents</b> .....	ix
<b>List of Tables</b> .....	xiii
<b>List of Figures</b> .....	xv
<b>1 Introduction</b> .....	1
1.1 Development of Simulation Technology .....	1
1.1.1 Explosion in Computational Power .....	2
1.1.2 Development of Simulation Techniques .....	4
1.2 Compositional Simulation .....	9
1.2.1 Formulation and Solution Method .....	10
1.2.1.1 Mass Balance Type .....	10
1.2.1.2 Volume Balance Type .....	17
1.2.2 Flash Calculation .....	20
1.2.2.1 Equilibrium Ratios Flash .....	20
1.2.2.2 Equality of Fugacities Flash .....	21
1.2.3 Treatment of Phase Disappearance and Reappearance .....	22
1.2.4 Other Numerical Techniques .....	23
1.3 Outline of Research Steps .....	25

<b>2</b>	<b>Basic Aspects of GPRS.....</b>	<b>27</b>
2.1	Gridding .....	27
2.2	Control Volume Formulation and Structure of Jacobian Matrix .....	31
2.3	Network Modeling .....	37
2.4	Multi-Point Flux Representation and Implementation.....	40
2.5	Well Treatment.....	43
2.6	Flash Calculation and Treatment of Phase Disappearance and Reappearance .....	48
2.7	Linear Solver .....	51
2.8	Timestep Control and Reservoir Initialization .....	54
<b>3</b>	<b>General Formulation Approach .....</b>	<b>57</b>
3.1	General Nonlinear Equation Set.....	59
3.2	General Formulation Approach Steps .....	62
3.2.1	Switch From the Natural Variables to Other Variables .....	63
3.2.2	Reduce Full Set to Primary Set .....	66
3.2.3	Reduce the Implicit Level of the Fully Implicit System .....	72
3.3	Stability Analysis .....	80
3.4	Influence of Equation and Variable Selections .....	84
3.4.1	Selection of Primary Equations.....	84
3.4.2	Different Variable Selections .....	86
<b>4</b>	<b>IMPSAT and IMPSAT Based AIM model .....</b>	<b>91</b>
4.1	IMPSAT Model.....	92
4.1.1	Why Does IMPSAT Model Work?.....	92
4.1.2	IMPSAT Stability Criterion? .....	95
4.1.3	Cost of IMPSAT Model.....	100
4.1.4	How to Build IMPSAT Efficiently?.....	102
4.2	IMPSAT Based AIM Model .....	103

<b>5</b>	<b>Model Validation and Performance .....</b>	<b>105</b>
5.1	Validation of GPRS.....	105
5.1.1	Black-Oil Model.....	106
5.1.2	Compositional Model.....	108
5.1.3	Unstructured Grid and Multi-Point Flux.....	111
5.1.4	General Formulation Approach.....	112
5.1.5	Non-conventional Well .....	115
5.2	Performance of Compositional Models in GPRS .....	117
5.2.1	Fully Implicit Models.....	119
5.2.2	IMPES Models .....	125
5.2.3	IMPSAT Model.....	132
5.2.4	AIM Models .....	139
<b>6</b>	<b>Conclusions and Future Work .....</b>	<b>147</b>
6.1	Conclusions .....	147
6.2	Future Work .....	150
	<b>Nomenclature.....</b>	<b>151</b>
	<b>References .....</b>	<b>155</b>
	<b>Appendix A: Overview of GPRS.....</b>	<b>165</b>
	<b>Appendix B: Black-Oil Simulation Using Compositional Formulation.....</b>	<b>179</b>
	<b>Appendix C: Flash Calculation.....</b>	<b>181</b>



## List of Tables

5.1	Iteration and timing comparisons for FIM models and Case 1 problem.....	123
5.2	Iteration and timing comparisons for FIM models and Case 2 problem.....	124
5.3	Iteration and timing comparisons for FIM models and Case 3 problem.....	124
5.4	Iteration and timing comparisons for FIM models and Case 4 problem.....	124
5.5	Iteration and timing comparisons for FIM models and Case 5 problem.....	125
5.6	Iteration and timing comparisons for IMPES models and Case 1 problem .....	130
5.7	Iteration and timing comparisons for IMPES models and Case 2 problem .....	131
5.8	Iteration and timing comparisons for IMPES models and Case 3 problem .....	131
5.9	Iteration and timing comparisons for IMPES models and Case 4 problem .....	131
5.10	Iteration and timing comparisons for the IMPSAT model and Case 1 problem .....	138
5.11	Iteration and timing comparisons for the IMPSAT model and Case 2 problem .....	138
5.12	Iteration and timing comparisons for the IMPSAT model and Case 3 problem .....	138
5.13	Iteration and timing comparisons for the IMPSAT model and Case 4 problem .....	139
5.14	Iteration and timing comparisons for AIM models and Case 1 problem .....	144
5.15	Iteration and timing comparisons for AIM models and Case 2 problem .....	144
5.16	Iteration and timing comparisons for AIM models and Case 3 problem .....	144
5.17	Iteration and timing comparisons for AIM models and Case 4 problem .....	145
5.18	Iteration and timing comparisons for AIM models and Case 5 problem .....	145



## List of Figures

1.1	State of the art CPU performance .....	2
1.2	Maximum practical model size .....	3
2.1	Effect of gridblock ordering on structure of Jacobian matrix .....	30
2.2	An unstructured grid with faults and multilateral wells.....	30
2.3	A general polyhedron.....	31
2.4	General structure of the Jacobian matrix.....	34
2.5	A simple unstructured grid with wells .....	35
2.6	Jacobian structure for the system in Figure 2.5.....	36
2.7	Gridblock connection array when the block-based approach is used .....	37
2.8	Connection array when the connection-based approach is used .....	38
2.9	Jacobian structure for two-point flux derivatives.....	40
2.10	Jacobian structure for multi-point flux derivatives .....	43
2.11	Wellbore flow.....	45
2.12	Performance comparison between GPRS solver and BlitzPak .....	52
3.1	Reduction process from full set to implicit primary set.....	58
3.2	Structure of the full set Jacobian matrix and RHS .....	60
3.3	Structure of the inverse of the transformation matrix for a Type B model.....	66
3.4	Reduce full set to primary set by Gaussian elimination.....	68
3.5	Structure of the primary set Jacobian matrix and RHS .....	71
3.6	Fully implicit Jacobian matrix of one gridblock .....	73
3.7	Jacobian matrix of one gridblock after explicit treatment of flux term .....	73
3.8	Householder reflection decoupling .....	75
3.9	Local inversion decoupling .....	76
3.10	Illustration for local inversion .....	77
4.1	$df_g / dS_g$ as a function of $S_g$ .....	96
4.2	$FS$ as a function of $S_g$ for difference $K_c$ values .....	97
4.3	Oil saturation at the well block .....	98
4.4	Maximum stable timestep size for the IMPES model.....	99
4.5	Maximum stable timestep size for the IMPSAT model.....	99
5.1	Black-oil reservoir.....	106

5.2	Oil production rate at the producer from GPRS and Eclipse 100 .....	107
5.3	Gas oil ratio at the producer from GPRS and Eclipse 100.....	107
5.4	Compositional reservoir .....	108
5.5	Well block and block (1,1,5) pressures from GPRS and Eclipse 300 with $\Delta t_{\max}=30$ days .....	109
5.6	Well block and block (1,1,5) oil saturations from GPRS and Eclipse 300 with $\Delta t_{\max}=30$ days .....	109
5.7	Well block and block (1,1,5) oil saturations from GPRS and Eclipse 300 with $\Delta t_{\max}=30$ days for GPRS and $\Delta t_{\max}=10$ days for Eclipse 300 .....	110
5.8	Unstructured grid.....	111
5.9	Water cut results from GPRS and FLEX for unstructured grid .....	112
5.10	Well block pressure for the FIM model with Type A and Type B variables.....	113
5.11	Well block oil saturation for the FIM model with Type A and Type B variables .....	113
5.12	Well block pressure for models with different implicit levels.....	114
5.13	Well block oil saturation for models with different implicit levels .....	114
5.14	Black-oil reservoir with a dual-lateral producer .....	115
5.15	Oil production rate at the producer from GPRS and Eclipse 100 for a dual-lateral well problem.....	116
5.16	Gas oil ratio at the producer from GPRS and Eclipse 100 for a dual-lateral well problem.....	116
5.17	Heterogeneous permeability field ( $\ln k$ ) for Case 3 .....	119
5.18	Well block oil saturation comparisons for FIM models and Case 1 problem.....	120
5.19	Well block oil saturation comparisons for FIM models and Case 2 problem.....	121
5.20	Well block oil saturation comparisons for FIM models and Case 3 problem.....	121
5.21	Well block oil saturation comparisons for FIM models and Case 4 problem.....	122
5.22	Well block oil saturation comparisons for FIM models and Case 5 problem.....	122
5.23	Stable timestep size for the IMPES model and Case 1 problem.....	126
5.24	Stable timestep size for the IMPES model and Case 2 problem.....	126
5.25	Stable timestep size for the IMPES model and Case 3 problem.....	127
5.26	Stable timestep size for the IMPES model and Case 4 problem.....	127



5.27	Well block oil saturation comparisons for IMPES models and Case 1 problem .....	128
5.28	Well block oil saturation comparisons for IMPES models and Case 2 problem .....	128
5.29	Well block oil saturation comparisons for IMPES models and Case 3 problem .....	129
5.30	Well block oil saturation comparisons for IMPES models and Case 4 problem .....	129
5.31	Stable timestep size for the IMPSAT model and Case 1 problem .....	133
5.32	Stable timestep size for the IMPSAT model and Case 2 problem .....	133
5.33	Stable timestep size for the IMPSAT model and Case 3 problem .....	134
5.34	Stable timestep size for the IMPSAT model and Case 4 problem .....	134
5.35	Well block oil saturation comparisons for the IMPSAT model and Case 1 problem .....	135
5.36	Well block oil saturation comparisons for the IMPSAT model and Case 2 problem .....	135
5.37	Well block oil saturation comparisons for the IMPSAT model and Case 3 problem .....	136
5.38	Well block oil saturation comparisons for the IMPSAT model and Case 4 problem .....	136
5.39	Well block oil saturation comparisons for AIM models and Case 1 problem ....	140
5.40	Well block oil saturation comparisons for AIM models and Case 2 problem ....	141
5.41	Well block oil saturation comparisons for AIM models and Case 3 problem ....	141
5.42	Well block oil saturation comparisons for AIM models and Case 4 problem ....	142
5.43	Well block oil saturation comparisons for AIM models and Case 5 problem ....	142
A.1	Structure of an oil field.....	166
A.2	Field level system model.....	166
A.3	Domain level system model.....	167
A.4	Reservoir level system model .....	168
A.5	Well level system model .....	169
A.6	Formulation level system model .....	170



# **Chapter 1**

## **Introduction**

Petroleum Reservoir Simulation has come a long way since its birth in the 1950s. It is worthwhile to take a brief look at the development of simulation technology before describing new developments resulting from the research reported here. Since compositional simulation is the major interest of this research, different aspects of compositional modeling are summarized, including formulation, phase behavior calculations and various numerical techniques used in the simulator. Following this, an overview of the current research project is given, including the design and implementation of a new General Purpose Research Simulator (GPRS), the development of a new flexible General Formulation Approach, the development of new compositional models and evaluation of different compositional models.

### **1.1 Development of Simulation Technology**

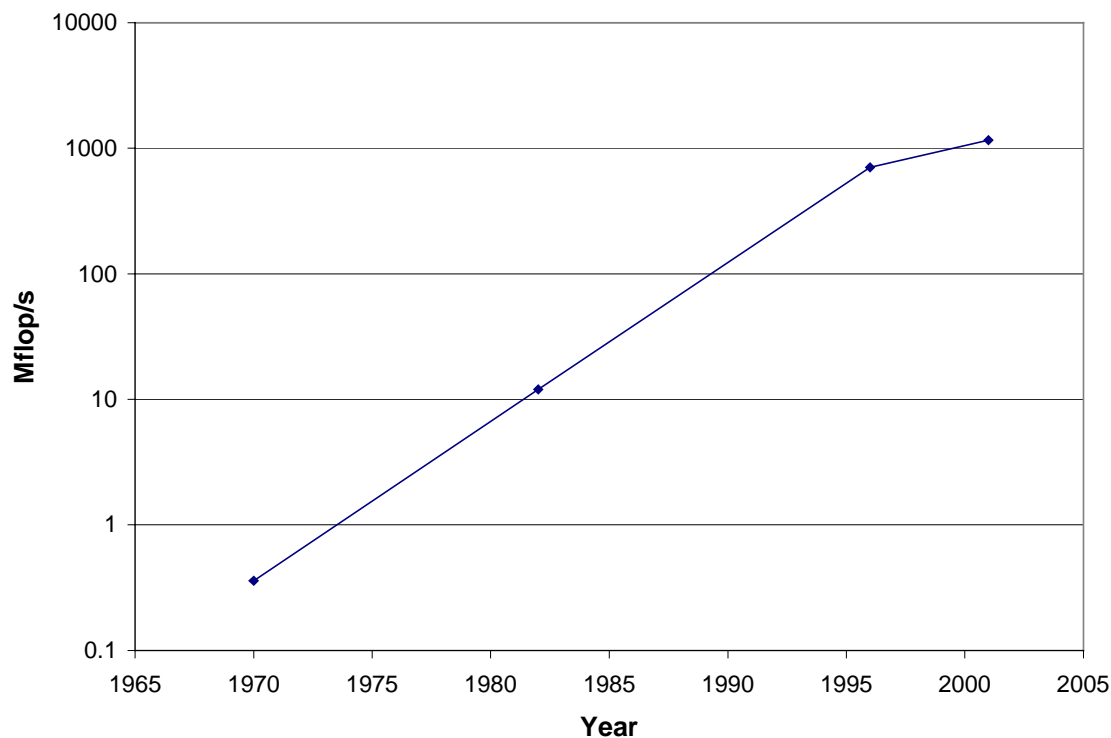
Reservoir simulation is the art and science of using numerical techniques to solve the equations for heat and mass flow in porous media, considering the appropriate initial and boundary conditions (Aziz and Settari, 1979). Since the availability of digital computers to the petroleum industry in the 1950s, more and more of the reservoirs have been studied with the aid of simulators (computer programs).

Reservoir simulation has evolved substantially since its birth. Today, reservoir simulation is a mature technology, and it is widely used in reservoir management. Nearly all major reservoir development decisions are based at least partially on simulation results (Watts, 1997).

The revolution in reservoir simulation over the last 50 years has been fueled by two major factors, the explosion in computational power and the development of new and advanced simulation techniques.

### 1.1.1 Explosion in Computational Power

The earliest computers were little more than adding machines by today's standards. Without substantial progress in computing, progress in reservoir simulation would have been impossible. Developments in computing power are concentrated in four areas, CPU speed, memory, parallel computing and programming languages. Each of them has contributed to the development of simulation in its own way.



**Figure 1.1** State of the art CPU performance (Dongarra, 2002)

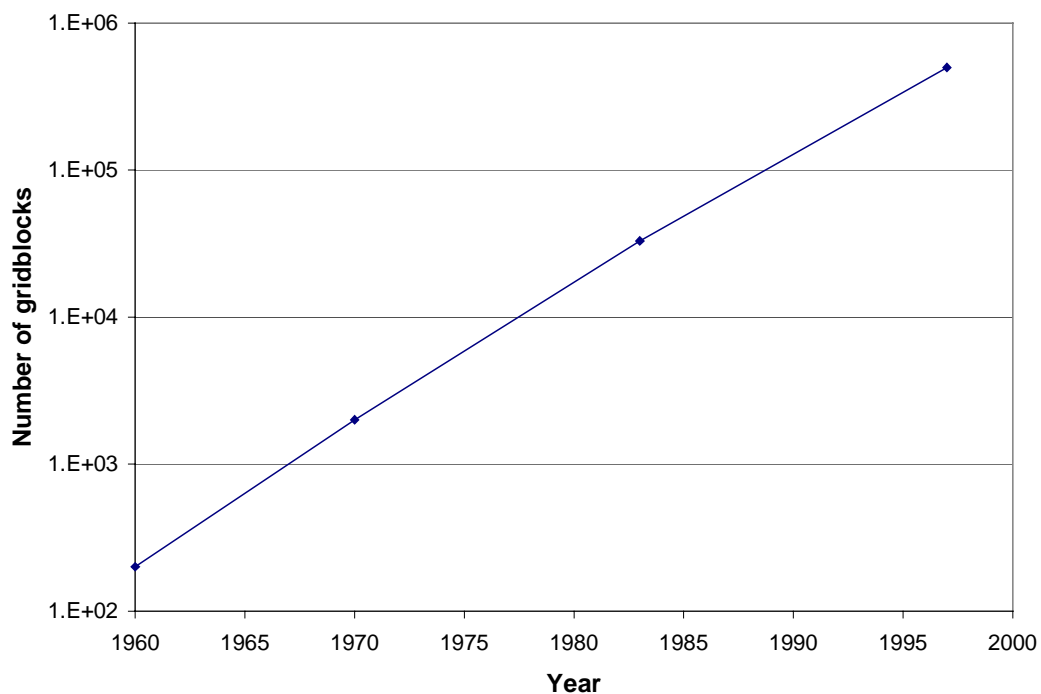
**CPU speed** Figure 1.1 gives the computing speed in millions of floating point operations per second (Mflop/s) for the four fastest CPUs of their time. CPU speed increased by nearly 2000 times from 1970 to 1996, just as predicted by Moore's Law (the number of transistors per square inch on integrated circuits doubles every 18 months). In the last several years, the rate of increase of CPU speed appears to be slowing down.

**Memory** A steady increase of available memory enables us to run larger and more complex programs. In the 1970s, the best computers available had only several Megabyte

of memory. Today, a good PC could have 2 Gigabyte (1Gigabyte = 1000Megabyte) of memory, and a parallel supercomputer could have over 50 Gigabyte of memory (Dogru et al., 2001).

**Parallel computing** Figure 1.1 shows single processor performance. Today, high-performance computing is achieved by using multiple processors in parallel. The resulting performance (parallel speedup) depends widely on problem type, size, and number of processors. Generally 32 processors lead to a speedup of 15-25 times for large (hundreds of thousands of gridblocks) reservoir models (Watts, 1997), and higher speedup factors are expected for even larger models.

**Programming languages** From the early version of Fortran to today's Fortran90 and C++, programming languages have developed through several stages. Today's object-oriented languages (such as C++) facilitate the design and development of large and complex computer programs. Modular approach and all kinds of source code control software (such as SourceSafe from Microsoft) make group programming easy and feasible, the cost of development is reduced, and programs developed using modern tools are easier to maintain.



**Figure 1.2** Maximum practical model size

A steady decrease in the price of computers has made simulation more and more affordable. A direct consequence of increasing computing speed and memory is the growth of model size. In 1960, the largest model size was around 200 gridblocks. By 1970, it had grown to about 2000 gridblocks. In 1997, it was roughly 500,000 gridblocks for simulations on a single processor (Watts, 1997). Today with parallel computing, the model sizes are becoming even larger. In a recent meeting Dogru (2001) reported simulations with up to 16 million gridblocks for three-phase black-oil problems, and routine simulation with one million gridblocks. Figure 1.2 summarizes these results, it shows a constant rate of growth in model size from 1960 to 1997, which is roughly proportional to the growth in computing speed. The fast growth in model size in the last several years is primarily due to the use of parallel computers.

The growth in computational power enables engineers to simulate large and complex compositional problems, and to develop and apply new simulation techniques. As a result, it is becoming possible to more accurately account for reservoir heterogeneities and process characteristics. At the same time, engineers and scientists are looking for ways to reduce the cost of reservoir simulation by reducing the number of unknowns per gridblock and by developing more robust linear solvers and preconditioners.

### **1.1.2 Development of Simulation Techniques**

In recent years, new techniques have appeared in all areas of simulation, including gridding, fluid modeling, numerical approximations, linear solvers, reservoir and geological modeling, etc. All of these technical advances are making simulation more accurate and realistic, and the simulator more robust, fast, stable, and user-friendly.

**Gridding** In the early stages of this technology, all reservoir simulations were performed on rectangular Cartesian grid, radial grid was developed later to simulate near well flow. Local grid refinement was developed to achieve better accuracy in high flow regions (Ciment and Sweet, 1973; Nacul, 1991; Pedrosa and Aziz, 1985). Development of corner-point geometry grid (Ponting, 1989; Ding and Lemonnier, 1995; Peaceman, 1996) made it possible to use non-rectangular gridblocks, providing the capability to model faults and other complex geological features. Up to this point, all grids were

structured, where the neighbors of a gridblock could be easily identified from their  $i, j, k$  indices. In the last decade, unstructured grids (Heinemann and Brand, 1989; Palagi and Aziz, 1994; Gunasekera et al., 1997; Kocerber, 1997; Aavatsmark et al., 1997; Verma and Aziz, 1996-1997) were introduced, which can easily force gridblocks to conform to major geological features. In unstructured grids, the connections between gridblocks are flexible, and a connection list is used to keep track of the connected gridblocks. Most unstructured grids are non-orthogonal grids, which often require the use of multi-point flux approximations. Today, users can define major reservoir units to which the simulation grid must conform, within each unit, the grid can be generated automatically with some guiding input from the user (Geoquest, 2000). Automatic gridding software packages are becoming available from commercial vendors, such as, FloGrid from Geoquest, GOCAD from the GOCAD group and GridPro from Program Development Corporation (PDC).

In the last several years, flow based unstructured grids (Agut et al., 1998; Edwards et al., 1998; Castellini et al., 2000) have been proposed, which are normally formed by generating streamlines and iso-potential lines from a fine grid single-phase simulation. Considerable smoothing is required to make the grid suitable for simulation. These grids roughly follow major geological features, such as faults, and they are concentrated in high flow rate areas, such as wells or high permeability regions. Edwards et al. (1998) showed that, if the grids follow the streamlines exactly, the multi-point flux calculations can be reduced to two-point flux calculations.

Beside developments in gridding techniques, Lim et al. (1994) proposed a new approach to the representation of grid information. In the conventional approach or the block-based approach, gridblocks and wells are tracked when computing flux terms. While in the new connection based approach, the network of connections are considered. A connection always links two nodes, each node can be a gridblock node, a well node or a surface facility node. A cell (gridblock) list is used for the calculation of accumulation terms, and a connection list is used for the calculation of flux terms. In Chapter 2, we will discuss the construction of the Jacobian matrix and the residual using this approach. This connection-based approach is extremely convenient for unstructured grids, since the

connection list itself has no structure at all. Besides this, it is also suitable for domain decomposition, surface facility modeling (Lim et al., 1994) and multi-point flux calculations.

**Fluid modeling** Initially, all simulations were based on the black-oil fluid model, where the hydrocarbon system is represented by two pseudo-components, oil and gas, according to their status at standard conditions. In the early 1980s, compositional simulation, where the hydrocarbon system is represented by an arbitrary number of components and pseudo-components (e.g. Aziz, 1996), became more mature and ready to use. The development of compositional simulation makes it possible to simulate volatile oil reservoirs, CO<sub>2</sub> flooding and other EOR processes. However, compositional simulation is much more expensive than black-oil simulation, due to the larger number of unknowns per gridblock and complex flash behavior.

**Numerical approximations** Conventional simulators use finite-difference methods with two-point flux calculations. More recently, multi-point flux calculations (Verma and Aziz, 1996; Gunasekera et al., 1998) are becoming more and more common due to the use of full tensor permeability (Lee et al., 1994; Lee et al., 1997) and non-orthogonal grids. At the same time, control volume methods (Aavatsmark et al., 1997; Verma and Aziz, 1997) have become the method of choice for today's simulator, because of their easy handling of unstructured grids. Limited work has also been done on higher-order schemes (Sammon, 1991; Chen et al., 1991) and finite element methods (Young, 1978; Fung et al., 1991; Sukirman and Lewis, 1994) to achieve higher accuracy.

**Linear Solver** Solving the linear system is the single most costly part for a simulation. So it is extremely important to have a good linear solver. Initially, direct solvers were used, but as the problems have become larger and larger (more and more gridblocks), iterative solvers have become more and more common. The performance of iterative solvers depends on the quality of preconditioners. In the petroleum industry, most of the effort in this area has been on building better preconditioners.

Any iterative solver can be used as a preconditioner for other iterative solvers, and different preconditioners can be combined together to form multistage preconditioners. Traditional preconditioners include, Incomplete LU decomposition (ILU) (e.g. Behie and



Forsyth, 1983), Gauss-Siedel (GS) (e.g. Aziz and Settari, 1979), Algebraic Multi-Grid (AMG) (Stueben, 1983), etc. Their performance depends strongly on the nature of the linear system. For near-elliptic system, AMG works well, for near-hyperbolic system, both ILU and GS work well. In reservoir simulation, we have a near-elliptic pressure equation and near-hyperbolic saturation equations. For this kind of mixed system, none of the single stage preconditioners (ILU, GS, AMG, etc) work well, we need a smarter preconditioner. Wallis et al. (1985) introduced the Constrained Pressure Residual (CPR) preconditioner, which is specially designed for reservoir simulation equations. It uses a two-stage approach to solve the pressure part and the saturation part of the reservoir equations separately, which make it the most promising preconditioner for fully implicit simulations. Unfortunately, a good preconditioner is still missing for the Adaptive Implicit (AIM) Method (Forsyth and Sammon, 1986).

Most of the available commercial linear solvers have been designed for structured grids, which result in Jacobian matrices with a banded structure. For unstructured grids, the Jacobian matrix is a general sparse matrix, and for such systems the existing solvers lose their efficiency. For problems with structured grids, the best iterative solvers can solve a problem with  $n$  unknowns in  $O(n^{1.1-1.2})$  time (see timing results in Section 2.7). However a lot of work needs to be done for unstructured grid solvers and AIM system solvers. For unstructured grids, the ordering of gridblocks and connections are quite important for the efficiency of the linear solver, and these issues will be discussed in Section 2.7.

**Reservoir and Geological Modeling** Reservoir simulations have also expanded in terms of options and features. Modern reservoir simulators can simultaneously handle multiple reservoirs, surface facilities (Schiozer and Aziz, 1994; Byer, 2000) and rock mechanics. Furthermore, there is more and more cooperation among reservoir engineer, geologist and geophysicist to achieve more realistic modeling of reservoir geology for simulation projects (e.g. Journal, 1990; Ballin et al., 1993).

With all of these developments in simulation technology and computers, the simulator has also reached a new stage. Today's simulator is more robust, more complex, and easier to use than what was available 20 years ago. Because of all the features that are

required in a modern simulator, it takes more time to develop a simulator and more effort to maintain it. In the future, simulators will be required to solve even more complex problems than they can handle today. Under these circumstances, a good design for the simulator is vital. Fortunately, today we have all kinds of design tools and suitable computer languages to develop technology to meet the future needs of the industry.

Research students working in this area need a lot of time to program the basic simulator before they can even start to explore research topics of interest to them. Also, after a student leaves, most of his/her development becomes unusable within a short time, due to lack of good design and documentation. Hence developing a good environment for simulation research has become essential for a research group like the one at Stanford University.

The first objective of this research is to develop a General Purpose Research Simulator (GPRS) with a highly modular structure. This will enable each future developer to work on extending modules of interest to him/her, without having an intimate knowledge of every aspect of other modules. A new researcher will be required to understand only the most basic structure of the simulator. Hopefully, this approach will save time in developing new techniques, and help preserve the work of new students. After the basic design and components have been put in place, the entire research group will be able to contribute to the project. An overview of the GPRS program, including its design, is included in Appendix A.

Inside this new General Purpose Research Simulator, we will incorporate many of the available new simulation techniques that were developed in the last decade, a number of which are unavailable in commercial simulators. Some of the features of particular interest to our research group are,

- unstructured grid (structured grid is handled as a special case of unstructured grid),
- network modeling (connection based approach),
- both two-point flux and multi-point flux calculation for network modeling,
- advanced linear solver technology (need further development),
- fully coupled surface facilities model (future development), and
- fully coupled geo-mechanical model (future development).

Currently, the first four features have been included in GPRS, and the fourth feature needs further development. We plan to add in the last two features in the near future.

GPRS is using a flexible compositional formulation, which is the major focus of this research, so it is worthwhile to take a further look at various aspects of compositional simulation.

## 1.2 Compositional Simulation

Petroleum reservoir fluids contain thousands of chemical components that affect their physical properties and phase behavior during production (e.g. Aziz, 1996). It is not practical to describe petroleum fluids in terms of individual components. Instead, pseudo-components (groups of molecules) with average physical properties are used to describe the reservoir fluid.

As mentioned earlier, we have two models for the fluid, black-oil and compositional. This selection is usually based on the volatility of the oil (e.g. Aziz, 1996). An oil, with gas solubility ratio ( $R_s$ ) less than 750scf/stb, oil formation volume factor ( $B_o$ ) less than 1.4bbl/stb and API gravity less than 30, is typically referred to as black-oil. In this case, phase behavior is simply represented by  $B_o$  and  $R_s$ , which are only functions of pressure, and flash calculations are not needed.

The black-oil model is just a special case of the compositional model, where the phase equilibrium relations can be reduced to linear relations between component mole fractions and pressure. Because of this, we can perform black-oil simulation using compositional formulation (refer to Appendix B for details), this is the case in GPRS.

Whenever it is inappropriate to use a black-oil model, the oil and gas must be described by more than two pseudo-components. In this case, phase behavior is represented by an Equation of State (EOS) and phase equilibrium relations, and this requires flash calculations. Sometimes, even when the reservoir fluid satisfies the black-oil condition, we may still need to use a compositional model for simulating such processes as  $\text{CO}_2$  injection,  $\text{N}_2$  injection, etc.

As we go from black-oil to compositional models, the number of equations and components increases, and the choice of equations and variables gets more complex. The

performance of the simulator (convergence of Newton iterations and convergence of linear solver iterations) can be strongly influenced by these factors. The choice of the implicit level (number of implicit variables for each gridblock) is another important factor here. Depending on the variable type, we may also need to switch variables when a hydrocarbon phase disappears or reappears.

The essential aspects of compositional simulation are discussed in detail in the next several subsections, including formulation, phase behavior and various numerical techniques. Here, we only consider isothermal compositional models.

### 1.2.1 Formulation and Solution Method

We can separate different compositional models into two basic types, mass balance type and volume balance type, according to the basic equations they solve. All of the following equations are for three-phase systems, and water and hydrocarbon phases are totally separated. In the following discussion,  $n_c$  is the number of components,  $n_h = n_c - 1$  is the number of hydrocarbon components, and  $n_p$  is the number of phases.

#### 1.2.1.1 Mass Balance Type

Just as the name implies, equations for this type of model are the mass balance equation for each hydrocarbon component (or pseudo-component) and for water (e.g. Coats, 1980; Young and Stephen, 1983):

$$F_{c,i} = \frac{\partial}{\partial t} [V\phi(S_o\rho_o x_c + S_g\rho_g y_c)]_i - \sum_l [T(\lambda_o\rho_o x_c \Delta\Phi_o + \lambda_g\rho_g y_c \Delta\Phi_g)]_{l,i} + \sum_w (\rho_o x_c q_o^w + \rho_g y_c q_g^w)_i = 0 \quad (1.1)$$

$$c = 1, \dots, n_h$$

$$F_{w,i} = \frac{\partial}{\partial t} (V\phi S_w \rho_w)_i - \sum_l (T\lambda_w \rho_w \Delta\Phi_w)_{l,i} + \sum_w (\rho_w q_w^w)_i = 0 \quad (1.2)$$

where,  $x_c$  and  $y_c$  are the hydrocarbon component mole fractions in the oil and gas phases respectively, and  $q_o^w$ ,  $q_g^w$  and  $q_w^w$  are the phase volumetric flow rates of well  $W$ .

Transmissibility is calculated by

$$T = \alpha \frac{kA}{\Delta L} \quad (1.3)$$

Phase mobility is determined by

$$\lambda_p = \frac{k_{rp}}{\mu_p} \quad (1.4)$$

Phase potential difference between gridblock i and j is defined as

$$\Delta\Phi_{p,i,j} = \Phi_{p,j} - \Phi_{p,i} = (p_{p,j} - p_{p,i}) - \beta \frac{g}{g_c} \rho_{p,i,j} (D_j - D_i) \quad (1.5)$$

where  $\alpha$  and  $\beta$  are unit conversion constants (For field units,  $\alpha=6.32654\text{E-}3$  and  $\beta=1.0/144$ ).

Sometimes it is convenient to replace one of the hydrocarbon component mass balance equations by a total hydrocarbon mass balance equation obtained by summing Eqn. 1.1 for all hydrocarbon components.

$$\begin{aligned} F_{h,i} = & \frac{\partial}{\partial t} [V\phi(S_o\rho_o + S_g\rho_g)]_i - \sum_l [T(\lambda_o\rho_o\Delta\Phi_o + \lambda_g\rho_g\Delta\Phi_g)]_{l,i} \\ & + \sum_w (\rho_o q_o^w + \rho_g q_g^w)_i = 0 \end{aligned} \quad (1.6)$$

In addition to mass balance equations, we need a phase equilibrium relation for each hydrocarbon component. Since water and hydrocarbon components are totally separated, we only need two-phase equilibrium relations between the oil and gas phases:

$$F_e = f_{c,o} - f_{c,g} = 0, \quad c = 1, \dots, n_h \quad (1.7)$$

where  $f_{c,o}$  and  $f_{c,g}$  are the fugacities of hydrocarbon component  $c$  in the oil and gas phases, respectively. Appendix C provides details of flash and fugacity calculations.

Finally some linear constraints have to be satisfied:

- Capillary pressure constraints:

$$F_{pcow} = p_{c,ow} - (p_o - p_w) = 0 \quad (1.8a)$$

$$F_{pcgo} = p_{c,go} - (p_g - p_o) = 0 \quad (1.8b)$$

- Saturation or volume constraint:

$$F_S = \sum_{p=1}^{n_p} S_p - 1 = 0 \quad (1.8c)$$

$$\text{or } F_V = V_\phi - V_T = 0 \quad (1.8d)$$

where  $V_\phi$  is the pore volume, and  $V_T$  is the total fluid volume.

- Component mole fraction constraints:

$$F_{po} = \sum_{c=1}^{n_h} x_c - 1 = 0 \quad (1.8e)$$

$$F_{pg} = \sum_{c=1}^{n_h} y_c - 1 = 0 \quad (1.8f)$$

For this system, the equations are:

<u>Type</u>	<u>Number</u>
$F_c$	$n_h - 1$
$F_h$	1
$F_w$	1
$F_e$	$n_h$
$F_{pcow}, F_{pcgo}$	2
$F_S$	1
$F_{po}, F_{pg}$	2
<hr/>	
Total	$2n_h + 6$

Same number of variables need to be selected, one possible choice is:

<u>Type</u>	<u>Number</u>
$P_p$	3
$S_p$	3
$x_c$	$n_h$
$y_c$	$n_h$
<hr/>	
Total	$2n_h + 6$

There are a total of  $2n_h + 6$  equations and  $2n_h + 6$  variables. It is neither practical nor necessary to solve all of these equations together. According to Gibbs phase rule, the thermodynamic degrees of freedom for this system are  $f_{thermo} = n_c + 2 - n_p$ . They fix the intensive state of the system. Besides this, the volume fraction of each phase ( $n_p - 1$  saturations) must also be determined to fix the extensive state. Assuming isothermal conditions, we have one less degree of freedom. The final degrees of freedom (Aziz, 1997) for an isothermal compositional system are

$$f = (n_c + 2 - n_p) + (n_p - 1) - 1 = n_c$$

So we only need to solve  $n_c$  of the  $2n_h + 6$  total equations to fix both the intensive and extensive state of the system. These  $n_c$  equations are our primary equations. They are also the equations solved simultaneously for fully implicit (FIM) models. There should be the same number of primary variables (variables for the FIM model). After solving for the primary variables, other variables (secondary variables) can be solved for from the remaining equations (secondary equations).

We can treat all of the primary variables implicitly, like in the FIM model. In order to further speedup the simulation process, we can also treat only some of the primary variables implicitly, first solve for them, then solve for the remaining primary variables by explicit or sequential implicit methods (Aziz and Settari, 1979; Watts, 1986).

Note, only the first  $2n_h + 1$  equations ( $F_c, F_h, F_w, F_e$ ) are non-linear, and the total number of independent mass balance equations ( $F_c, F_h, F_w$ ) is equal to the number of

primary equations and variables. We can use the linear constraint equations to remove two pressures, one saturation and two component mole fractions. Finally, only  $2n_h + 1$  number of non-linear equations and variables is left, which is the full set of equations and variables. Out of this full set,  $n_c$  primary equations and variables are selected. Since mass balance equations represent flow in the reservoir, and their number always equals the number of primary equations, they are the natural choice for primary equations. But the selection of primary variables is widely varied among models. After the primary set is selected, the equations and variables left over in the full set form the secondary equations and variables.

There are a lot of models that belong to this class, and they are different in the way that primary equations and variables are chosen. Some of the basic models are reviewed next.

**The Fussell and Fussell Model (1979)** For this model the primary equations are the  $n_h$  phase equilibrium relations and the saturation constraint equation. The primary variables are  $p$ ,  $l$  (hydrocarbon liquid mole fraction), and  $y_c$  or  $x_c$ ,  $c=2, \dots, n_h$ , depending on whether the system in a gridblock is predominantly vapor or liquid. The  $n_h$  phase equilibrium relations are used to remove all of the primary variables except pressure from the saturation constraint equation. The saturation constraint equation is the final IMPES (Only pressure implicit formulation) pressure equation. This is the only model that uses phase equilibrium relations as primary equations, and it is rarely used.

**The Coats Model (1980)** For this model the primary equations are the  $n_c$  mass balance equations for each hydrocarbon component and for water. The primary variables are,

- $p, S_o, S_g, y_c, c=3, \dots, n_h$  (when both the oil and gas phases are present),
- $p, S_o, x_c, c=1, \dots, n_h - 1$  (when no free gas is present), and
- $p, S_g, y_c, c=1, \dots, n_h - 1$  (when the oil phase disappears).

In general, adjacent gridblocks may have different sets of primary variables, and we need to switch variables when a hydrocarbon phase disappears or reappears. The variable selection (pressure, saturations and component mole fractions) in this model is normally called the natural variable selection, since the mass balance equations can be directly



expressed in terms of these variables and calculation of derivatives of the Jacobian matrix is very easy. This model was initially proposed as a FIM model, but it can also be reduced to an IMPES model by variable elimination.

**The Young and Stephenson Model (1983)** For this model the primary equations are the  $n_c$  mass balance equations and the saturation constraint equation, and the last hydrocarbon component mass balance equation is replaced by the total hydrocarbon mass balance equation. The primary variables are  $p$ ,  $F = \rho_o S_o + \rho_g S_g$  (total moles of hydrocarbon in one unit volume of fluid),  $W = \rho_w S_w$  (total moles of water in one unit volume of fluid) and  $z_c$  (overall mole fraction of component  $c$ ),  $c=1, \dots, n_h-1$ . The  $n_c$  mass balance equations are used to eliminate all of the primary variables except pressure from the saturation constraint equation. The saturation constraint equation is the final IMPES pressure equation. This approach does not depend on the presence of individual phases, since all of its primary variables are overall quantities. On the other hand, saturation is not one of the primary variables, so we have to use the chain rule to evaluate some of the derivatives of the flow term with respect to saturation, such as relative permeability.

For this model, there are  $n_c+1$  primary equations and variables, one more than necessary. Sometimes, this is done to facilitate the Jacobian matrix calculation, especially when certain constraint equation is nonlinear with respect to the selected primary variables (here, the saturation constraint equation is a nonlinear equation in terms of the primary variables). Actually each small  $(n_c+1) \times (n_c+1)$  block system can be reduced to a  $n_c \times n_c$  block system. An alternative approach is to treat this extra nonlinear constraint equation as a secondary equation, then there will be only  $n_c$  primary equations.

**The Chien, Lee, and Chen Model (1985)** For this model the primary equations are the  $n_c$  mass balance equations for each hydrocarbon component and for water. The primary variables are  $p$ ,  $W$  and  $r_c = z_c F$  (total moles of component  $c$  in one unit volume of fluid),  $c=1, \dots, n_h$   $c \neq \max$ . ( $r_{\max}$  is defined as the  $r_c$  with respect to which the derivative of the saturation constraint is largest.). This model also does not depend on the presence of individual phases. Also, this model needs to use the chain rule to evaluate some of the flow term derivatives. The equations of this model are normally solved fully implicitly.

Each of the models described above could be either fully implicit or only implicit in pressure. In the original Young and Stephenson model (Young and Stephenson, 1983), only pressure is treated implicitly, while in the original Coats model (Coats, 1980), all primary variables are treated implicitly. There are a lot of other models, and each of them is actually a variation of one of these basic models. The variations are obtained by changing the level of implicitness, changing intensive and extensive variables, and so on.

In summary, all of these isothermal compositional models (except the Fussel and Fussel model) select mass balance equations as the primary equations, and there are basically two types of primary variables (Aziz, 1996).

**Type A variables:**    one pressure  
                                $n_p - 1$  saturations  
                                $n_c - n_p$  component mole fractions

**Type B variables:**    one pressure  
                                $n_c - 1$  overall quantities, such as overall component mole fractions

The Coats model (Coats, 1980) belongs to Type A. The Young and Stephenson model (Young and Stephenson, 1983) and the Chien, Lee and Chen model (Chien et al., 1985) belong to Type B. Each type of variables has its own advantages and disadvantages. The common points and differences between these two types of variables are given below.

#### **Common Points:**

- Mass balance equations are always the primary equations.
- Phase equilibrium relations are always used to eliminate the secondary variables from the primary equations.
- After solving for the primary variables, the secondary variables are updated explicitly via the phase equilibrium relations gridblock by gridblock
- Both types of models can be either fully implicit (FIM) or only implicit in pressure (IMPES).

**Differences:**

- Type A variables are the natural variables, equations can be explicitly expressed in them, and the Jacobian matrix is easy to calculate. For Type B variables, we have to use the chain rule to calculate the derivatives for the Jacobian matrix. For example, the chain rule is required to calculate derivatives of phase relative permeability with respect to overall component mole fractions.
- In Type A, phase variables (saturation and component mole fractions in each phase) are used, so they depend on the appearance and disappearance of hydrocarbon phases. For each gridblock, if a hydrocarbon phase changes status, we need to switch the primary variables (replacing the non-existing phase variables with the corresponding variables for the existing phase). This generally leads to different primary variables in different gridblocks. Additional work is required to track the variables used in each gridblock, check the existence of individual hydrocarbon phases and perform the switch. Type B uses overall compositions as the primary variables, and the same primary variables can be used for all gridblocks.

**1.2.1.2 Volume Balance Type**

This type of models (Acs et al., 1982; Kendall et al., 1983; Watts, 1986; Aziz and Wong, 1988-1989) are based on the volume balance concept, and use the volume balance equation as the pressure equation to form IMPES type formulations. Since the pore space of porous medium must be fully filled with fluids present, the pore volume must be equal to the total fluid volume at any time and place. This can be expressed as

$$V_{\phi} = V_T = V_w + V_o + V_g \quad (1.9)$$

The pore volume  $V_{\phi}$  is solely a function of pressure:

$$V_{\phi}^{n+1} = V_{\phi}^n + V_{\phi}^o c_r (p^{n+1} - p^n) \quad (1.10)$$

where  $c_r$  is the rock compressibility.

The total fluid volume  $V_T$  is a function of both pressure and overall composition of each component:

$$V_T^{n+1} = V_T^n + \left( \frac{\partial V_T}{\partial p} \right)^n \bigg|_{M_w, M_c} (p^{n+1} - p^n) + \left( \frac{\partial V_T}{\partial M_w} \right)^n \bigg|_{p, M_c} (M_w^{n+1} - M_w^n) + \sum_{c=1}^{n_h} \left( \frac{\partial V_T}{\partial M_c} \right)^n \bigg|_{p, M_w, M_j, j \neq c} (M_c^{n+1} - M_c^n) \quad (1.11)$$

where,  $M_c$  and  $M_w$  are the overall mole amounts of hydrocarbon components and water.  $M_c^{n+1} - M_c^n$  and  $M_w^{n+1} - M_w^n$  are actually the accumulation terms in the mass balance equations, which can be written as

$$\frac{M_c^{n+1} - M_c^n}{\Delta t} = U_c^{n,n+1}, \quad c = 1, \dots, n_h \quad (1.12)$$

$$\frac{M_w^{n+1} - M_w^n}{\Delta t} = U_w^{n,n+1} \quad (1.13)$$

where:

$$U_c^{n,n+1} = \Delta [T(\lambda_o \rho_o x_c)^{n,n+1} \Delta \Phi_o^{n+1} + T(\lambda_g \rho_g y_c)^{n,n+1} \Delta \Phi_g^{n+1}] + Q_c^{W,n+1} \quad (1.14)$$

$$U_w^{n,n+1} = \Delta [T(\lambda_w \rho_w)^{n,n+1} \Delta \Phi_w^{n+1}] + Q_w^{W,n+1} \quad (1.15)$$

Note that, for IMPES, the transmissibility terms ( $\lambda_o$ ,  $\lambda_g$ ,  $\lambda_w$ ,  $\rho_o$ ,  $\rho_g$ ,  $\rho_w$ ,  $x_c$ ,  $y_c$ ) are fixed at the old time level. The time level for terms with superscript “ $n, n+1$ ” depends on the implicitness of the model.

The final pressure equation is obtained by substituting Eqn. 1.10, Eqn. 1.12 and Eqn. 1.13 into Eqn. 1.11. After rearranging, it can be written as

$$\left[ V_\phi^o c_r - \left( \frac{\partial V_T}{\partial p} \right)^n \bigg|_{M_w, M_c} \right] (p^{n+1} - p^n) = [V_T - V_\phi]^n + \Delta t \left[ \sum_{c=1}^{n_h} \left( \frac{\partial V_T}{\partial M_c} \right)^n \bigg|_{p, M_w, M_j, j \neq c} U_c^{n,n+1} + \left( \frac{\partial V_T}{\partial M_w} \right)^n \bigg|_{p, M_c} U_w^{n,n+1} \right] \quad (1.16)$$

After solving for pressure from Eqn. 1.16, other variables are updated explicitly gridblock by gridblock.

The key part of this method is to calculate the derivatives of individual phase volumes with respect to both pressure and overall moles of each component  $(\frac{\partial V_p}{\partial p}, \frac{\partial V_p}{\partial M_c}, \frac{\partial V_p}{\partial M_w})$ .

In order to calculate these derivatives analytically, we need to make full use of the phase equilibrium relations. Different authors have proposed different methods to calculate these derivatives. Models of this type were proposed by Acs et al., 1982; Kendall et al. 1983; Watts, 1986; Aziz and Wong, 1988-1989.

Comparing the mass balance type models with the volume balance type models, the volume balance type models are only implicit in pressure, basically it is an IMPES approach. Coats (1999) has pointed out that the IMPES pressure equation is unique, independent of the manner of derivation, choice and ordering of variables and equations. So no matter which type of model we use, as long as it finally reduces to the IMPES pressure equation, numerically they are equivalent to each other. They are only different in two aspects:

- The number of operations to generate the final IMPES pressure system is different. From this point of view, the Young and Stephenson model is more efficient than the Coats model (Coats, 1999).
- The Jacobian matrix is scaled differently from gridblock to gridblock, which could influence the convergence rate of iterative linear solvers (Coats, 1999).

In two papers, Aziz and Wong (1989) and Wong et al. (1990) showed the connection between these two types of models, and how to derive the volume balance model starting from the Jacobian matrix of a mass balance type model. In this research, we will just treat the volume balance type model as a special IMPES case of the mass balance type model.

Models that treat only pressure implicitly are referred to in the literature by different names: IMPES (implicit pressure and explicit saturations), IMPEM (implicit pressure and explicit overall component mass/mole). As stated earlier, regardless of the variables selected, the pressure equations are always equivalent to each other. In this work, we do not distinguish between them, and refer to them as IMPES models.

### 1.2.2 Flash Calculation

Compared to black-oil models, compositional models are not only more complicated in their formulations, they also require additional flash calculations. In black-oil models, the phase equilibrium relations can be reduced to a series of linear relations between component mole fractions and pressure, so there is no need for flash calculations. In compositional models, we need to do flash calculation for each gridblock at each iteration of each timestep. There are two types of flash calculations:

#### 1.2.2.1 Equilibrium Ratios ( $K_c$ ) Flash

This is the simplest approach (Young and Stephenson, 1983) to flash calculations. The equilibrium ratios are functions of both pressure and mole fractions, in this approach we assume that these functions are known. We have the following equilibrium relations:

$$y_c = K_c x_c \quad (1.17)$$

$$z_c = l x_c + (1-l) y_c \quad (1.18)$$

where  $l$  is the hydrocarbon liquid (oil) mole fraction in hydrocarbon fluid. From Eqn. 1.17 and Eqn. 1.18, we can further express  $x_c$  and  $y_c$  in terms of  $l$ ,  $z_c$  and  $K_c$ :

$$x_c = \frac{z_c}{l + (1-l)K_c} \quad (1.19)$$

$$y_c = \frac{K_c z_c}{l + (1-l)K_c} \quad (1.20)$$

From Eqn. 1.8e, Eqn. 1.8f, Eqn. 1.19 and Eqn. 1.20, we can derive the Rachford-Rice equation (Rachford and Rice, 1952), which is solely a function of  $l$  for a given  $z_c$  and  $K_c$ :

$$0 = 1 - 1 = \sum_c y_c - \sum_c x_c = \sum_c (y_c - x_c) = \sum_c \frac{(K_c - 1)z_c}{l + (1-l)K_c} \quad (1.21)$$

The solution procedure reduces to the following two steps:

1. Solve for  $l$  from Eqn. 1.21
2. Solve for  $x_c$  and  $y_c$  from Eqn. 1.19 and Eqn. 1.20

### 1.2.2.2 Equality of Fugacities Flash

At equilibrium,  $F_e = f_{c,o} - f_{c,g} = 0$ ,  $c = 1, \dots, n_h$ , which are a series of nonlinear equations. They can be solved by either the successive substitution method or the Newton-Raphson method (Fussel and Yanosik, 1978). The solution procedure for the successive substitution method is given below:

1. Solve for  $l$  from Eqn. 1.21
2. Solve for  $x_c$  and  $y_c$  from Eqn. 1.19 and Eqn. 1.20
3. Calculate the phase compressibility factor  $Z$  for each phase from an Equation of State (EOS)
4. Calculate the fugacity for each component in each phase
5. Check convergence ( $|f_{c,o} / f_{c,g} - 1.0| < \varepsilon$ )
6. If not converged, update equilibrium ratios by  $K_c^{new} = K_c^{old} \cdot (f_{c,o} / f_{c,g})$ , and repeat step 1 to 6

Appendix C derives all of the analytical expressions of derivatives needed for the Newton-Raphson method. The successive substitution method converges slowly, but convergence is guaranteed, the Newton-Raphson method converges quickly, but it only converges near the true solution. So, a typical approach is to use the successive substitution method first, after the error is reduced to a certain level, the Newton-Raphson method is used to finish it quickly (Nghiem et al., 1983).

For different Equation of State (EOS), we have different equations to calculate the component fugacities. Cubic Equations of State are commonly used in the petroleum industry, and among them the Peng-Robinson EOS (Peng and Robinson, 1976) is the most popular one. A properly tuned Cubic EOS is generally adequate for compositional simulation.

Besides phase equilibrium calculations, this procedure also yields phase properties. The oil phase molecular weight is calculated as

$$MW_o = \sum_{c=1}^{n_h} MW_c \cdot x_c \quad (1.22)$$

and the gas phase molecular weight is calculated as

$$MW_g = \sum_{c=1}^{n_h} MW_c \cdot y_c \quad (1.23)$$

The hydrocarbon phase densities are calculated as

$$\rho_p = \frac{P_p}{ZRT} \quad (1.24)$$

where compressibility factor  $Z$  is one of the three roots of the Cubic EOS, for the oil phase the smallest real root is used and for the gas phase the largest real root is used.

In the early days, most simulators used the successive substitution method, today most of them prefer the faster Newton-Raphson method or a combination of both methods. Some simulators separate flash calculation from flow calculation, especially for simulators using Type B variables.

### 1.2.3 Treatment of Phase Disappearance and Reappearance

In a gridblock, there may be up to two hydrocarbon phases (oil and gas). Whether both hydrocarbon phases exist or not, depends on pressure, temperature and overall composition. During simulation, we need to establish the existence of hydrocarbon phases in all gridblocks. If only a single hydrocarbon phase exists, equations and variables for some models may have to be changed.

**Phase Disappearance** When a gridblock has two hydrocarbon phases, we need to monitor its condition to find when it will change to a single hydrocarbon phase. Basically, there are three ways to do it:

- For each Newton iteration, we solve the flow equations, and get the saturation solutions. If either  $S_o$  or  $S_g$  is negative, it is set to zero before initializing for the next Newton iteration (Coats, 1980). In this case, the corresponding hydrocarbon phase disappears.
- A flash calculation is performed. If the liquid mole fraction,  $l$ ,  $<0$ , then the oil phase does not exist anymore, else if  $l > 1$ , then the gas phase disappears (Nghiem et al., 1983). This kind of flash is called negative flash (Whitson and Michelson, 1986).
- A new set of pseudo-equilibrium ratios (PER's) is used, and only one set of equations and constraints are used at all times to simulate the process regardless of the phase condition (Abou-Kassem and Aziz, 1985). This method is suitable for both compositional and thermal models.



**Phase Reappearance** When a gridblock has only a single hydrocarbon phase, we also need to monitor its condition. Because whenever it changes to two phases, we may need to change equations and variables again. The following three approaches can be used for this:

- For each Newton iteration, if either  $S_o$  or  $S_g$  is zero in a gridblock, then a saturation pressure is calculated for the single hydrocarbon phase fluid present in the block. If the calculated saturation pressure is less than the gridblock pressure, the gridblock remains in single hydrocarbon phase mode, otherwise, both hydrocarbon phases exist and the absent-phase saturation is set to 0.001 (Coats, 1980).
- A flash is performed, if  $l$  is between 0 and 1, then both hydrocarbon phases exist (Nghiem et al., 1983). We can accelerate the convergence of flash calculations by getting the initial values from the neighboring blocks (Young and Stephenson, 1983).
- The third treatment (Abou-Kassem and Aziz, 1985) of phase disappearance can be applied without any change, since the same set of equations and constraints is used at all times regardless of the phases present.

#### 1.2.4 Other Numerical Techniques

**Partial Jacobian Update** (Young and Stephenson, 1983) If a gridblock satisfies the convergence criteria, its old Jacobian values are retained. In other words, the Jacobian terms are updated only for un-converged gridblocks. This technique can save some time in Jacobian calculations, and it does not have a large influence on the convergence behavior of Newton iterations. At convergence, correct results are guaranteed.

**Relaxed Volume Balance Concept** (Coats et al., 1998) This technique can be used in both the FIM and the IMPES models, typically it is used in IMPES models. For the linear constraint equations, such as  $S_w + S_o + S_g - 1 = 0$ , we use

$$\begin{cases} S_g = 1 - S_w - S_o \\ \delta S_g = e - \delta S_w - \delta S_o \end{cases} \quad \text{where } e = 1 - S_w - S_o - S_g$$

to directly substitute for  $S_g, \delta S_g$  into the nonlinear equations in order to remove them. When  $e = 0$ , volume is conserved, when  $e$  is not zero, volume is not conserved, it is called the relaxed volume balance approach because some volume balance error (phase

saturations do not exactly sum to 1.0) is accepted. However, exact mass balance at each Newton iteration is achieved. In effect, this procedure amounts to iterating out volume balance error while keeping exact mass balance.

This relaxed balance concept can be used for any linear constraint equation. If exact balance is not achieved at the last timestep, then this balance error can be corrected in the current timestep.

**One Step Flash Calculation** Theoretically, we should do exact flash calculation for each Newton iteration. But at most times, it is worthwhile to do only a few flash iterations without full convergence. This is appropriate because the result of each Newton iteration is only an approximation to the final solution, and exact flash is not required. On the other hand, If the flash results are far from convergence, it may worsen the convergence rate of Newton iterations.

From the above discussion, we can see that compositional simulation is much more complex than black-oil simulation. There are a variety of compositional models, and they are quite different from each other. Some analysis and evaluation of the performance of individual compositional models has been published (Coats, 1999), but it is far from complete. There are two major reasons for the lack of appropriate comparisons:

- It is hard to implement different compositional models in one simulator. They just have too many differences. There is a lot of work involved in programming, before we can even start to evaluate different models.
- Compositional simulators are much harder to develop than black-oil simulators. This kind of development is time consuming, requires one to deal with complex issues associated with flash calculations and programming.

Since the fully implicit (FIM) models are too expensive for large problems and the timestep limits on the IMPES model are too strict for hard problems, we need a new model that is cheaper than the FIM model and much more stable than the IMPES model.

Another objective of this research, in addition to developing a General Purpose Research Simulator (GPRS), is to evaluate the performance of different isothermal compositional models, and where appropriate, develop new models. Before the work can be started, we need to develop a General Formulation Approach which can be used to

derive any kind of model. Then, during programming, differences between different models can be minimized, and we can quickly change from one model to another model. This should lead to a consistent evaluation of the performance of different models.

### **1.3 Outline Of Research Steps**

The major focus of this research is to develop a general-purpose simulation framework with emphasis on simulator design and compositional simulation. This will be achieved in four major steps:

**Design and Implementation of GPRS** As mentioned earlier, a well-designed simulator can improve group productivity and help preserve work of different investigators. A good design using a modular object-oriented approach is required. A lot of advanced techniques will be implemented and tested in this simulator, including unstructured grid handling, network modeling, multi-point flux calculation, linear equation solution techniques, etc (Chapter 2).

**General Formulation Approach and Its Implementation in GPRS** A General Formulation Approach will be developed, and this general approach should be capable of yielding any kind of model. Both variables of Type A and Type B should be available and the level of implicitness for each gridblock should be flexible (from 1 to number of components). The details of this General Formulation Approach are in Chapter 3.

**Development of New Compositional Models** Another purpose of this research is to develop new compositional models. A new model using implicit pressure, saturations and explicit component mole fractions (IMPSAT) is discussed in Chapter 4.

**Evaluation of Different Compositional Models** The final step of this research is to evaluate the performance of different compositional models under different conditions, which include large number of components, large systems, highly heterogeneous systems, etc. These aspects are discussed in Chapter 5. Our final goal is to find a way to minimize the computational time without sacrificing accuracy.



## Chapter 2

### Basic Aspects of GPRS

One of the objectives of GPRS is to build a general framework for research simulators, and allow a group of students to work with it and preserve the work of different investigators. In order to make this process easier, it is worthwhile to discuss the basic aspects of GPRS in detail, including gridding, formulation, flux calculation, well treatment, flash calculation, linear solvers, timestep control and reservoir initialization. Within GPRS, we intend to incorporate the most advanced simulation techniques developed in the last decade. So far, we have included unstructured grid, control volume formulation, network modeling, two-point and multi-point flux for unstructured grid, negative flash calculation, compositional well control, and advanced linear solver technology. In this chapter, we will discuss the basic aspects of GPRS, review the advanced techniques used in each part, and explain how these techniques are incorporated.

#### 2.1 Gridding

Today, the trend in simulator development is to separate the gridding part from the flow calculation (simulator) part, and use the output from the gridding part as input for the simulator. The gridding part is normally done by a gridding software. Through this approach, one simulator should be able to couple with different gridding software, and each gridding software should be able to provide service for multiple simulators. Also, this approach can lower the development cost and increase flexibility for both the gridding part and the simulator part. However it is not very suitable for dynamic gridding, where the grid changes with time. Fortunately, dynamic gridding is rarely used in today's reservoir simulators.

This separated approach is partially adopted in GPRS, and GPRS can get its grid information in two ways:

- **Internal generation** GPRS has a Cartesian grid module, which can generate all of the necessary information for a Cartesian grid. A similar model for Radial grid may be added later.
- **External read in** For more advanced grids, such as unstructured grids, the grid information is read from the output of a gridding software. Actually, this option is suitable for all kinds of grids, including Cartesian grid.

Basically, GPRS is a reservoir simulator without much gridding capability, and unstructured grid simulations are supported by interfacing with other gridding software.

In the petroleum industry, the available commercial gridding software includes FloGrid from Geoquest, GOCAD from the GOCAD group and GridPro from Program Development Corporation (PDC), etc. No matter which gridding software, they always share the following common functions:

- Read information (location, size and physical properties) from a geological fine grid, read major geological features, such as faults (location, length, direction) and read well specification (trajectory, diameter and skin)
- Perform automatic gridding according to user input (grid type, dimension and other options), and generate simulation grid
- Perform automatic upscaling from the geological fine grid to the simulation grid, calculate the simulation grid properties, build connections between gridblocks and calculate well indices (the connection between wellbore and well block). Note that some gridding packages do not perform upscaling, such as GridPro.
- Output all of the simulation grid properties and well information in a specific format, which can be read later by a simulator

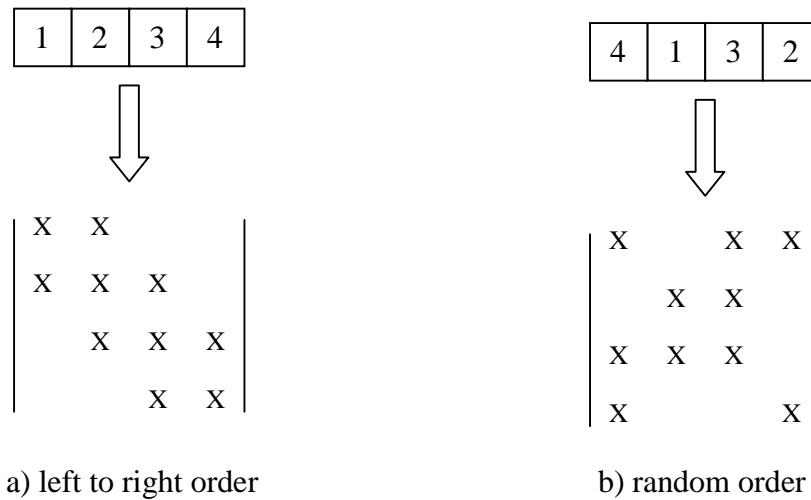
Normally, each gridding software is specially designed to serve several simulators, for example, FloGrid was initially designed to provide input for both ECLIPSE (Geoquest simulator) and FIRST! (MOBIL simulator). In order to do that, gridding software normally defines a specific output format for each served simulator. If the output file is a text file, it is possible to write an external program to read in and reformat the information and make it suitable for other simulators, such as GPRS.

In order to make the gridding software and the simulator work together, we have to define which information should be passed on from the gridding software to the simulator. The necessary information normally includes a list of cell properties, a list of connection properties, and a list of well data. On the other hand, the simulator should be able to handle the passed grid information, such as unstructured grid, multi-point flux. GPRS is designed to handle the most general grid type and grid connection data, and it should be able to interface with any gridding software. The gridding information that GPRS needs includes the following:

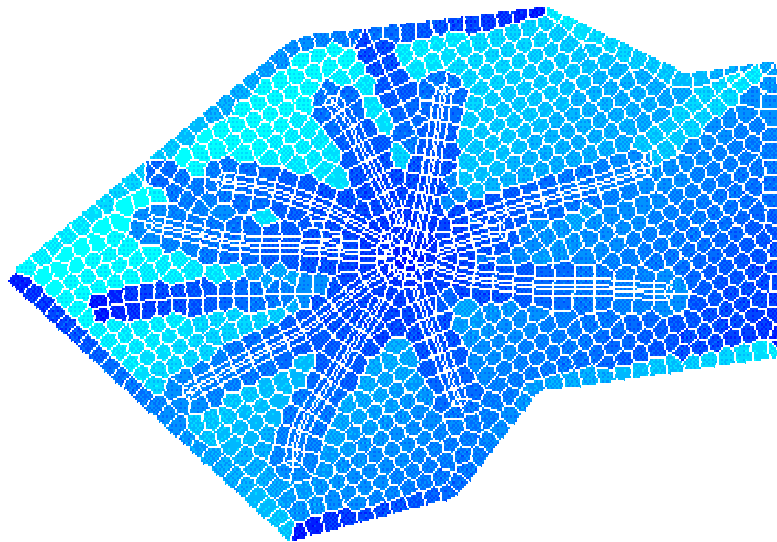
- List of cell properties. For each cell, we need,
  - dimensions (DX, DY, DZ) (not necessary),
  - depth,
  - porosity,
  - permeability (not necessary, included in the transmissibility and WI data), and
  - volume.
- List of connection properties. For each connection, we need,
  - number of gridblocks for each connection (for two-point flux, it is always 2),
  - gridblock index for each gridblock, and
  - transmissibility constant for each gridblock.
- List of well data. For each well, we need,
  - number of gridblocks that this well penetrates,
  - index for each gridblock penetrated by this well (well block), and
  - well index (WI) for each well block.

The ordering of the cell list and connection list is quite important to the efficiency of a simulator, especially in the linear solver part. Considering a simple  $4 \times 1 \times 1$  case, Figure 2.1 shows two different cell orderings and the corresponding Jacobian matrix structures. In Figure 2.1a, ordering is from left to right, and the Jacobian Matrix has a tridiagonal structure. In Figure 2.1b, ordering is random, and the corresponding Jacobian matrix has a random structure, which makes it much more difficult to solve than the tridiagonal matrix. For structured grids, the best way to order is to order in the smallest dimension

first and order in the largest dimension last. But for unstructured grids, such as the one shown in Figure 2.2, it is not so easy to decide which kind of ordering is the best. In general, when we order the cell list of an unstructured grid, we wish to preserve as much structure information as possible, and generate a Jacobian matrix with the smallest possible bandwidth. The ordering of the connection list does not influence the structure of the Jacobian matrix, but still, in order to make the best cache utilization in flux calculations, it is best to roughly follow the ordering of the cell list.



**Figure 2.1** Effect of gridblock ordering on structure of Jacobian matrix



**Figure 2.2** An unstructured grid with faults and multilateral wells (FloGrid, 2000)

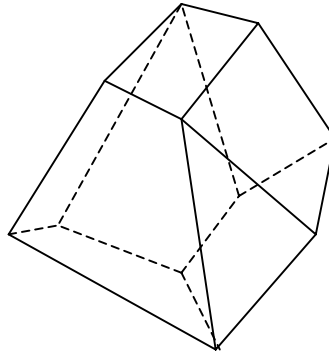


## 2.2 Control Volume Formulation and Structure of Jacobian Matrix

There are different types of numerical methods for the discretization of reservoir flow equations. Traditionally, finite difference methods have been used, in recent years, with the development of unstructured grids, more and more use is being made of control volume methods (Aavatsmark et al., 1997; Verma and Aziz, 1997), which make the handling of irregular shapes much easier. In case of Cartesian grids, control volume methods and finite difference methods produce the same discretized equations. Finite element methods (Young, 1978; Fung et al., 1991; Sukirman and Lewis, 1994) are another option, but they are not widely used in the petroleum industry. The control volume formulation is used in GPRS.

Reservoir flow equations are a series of nonlinear equations, which are normally solved by the Newton-Raphson method, where a Jacobian matrix and a right hand side (RHS) are calculated at each Newton iteration. In this section we will first introduce the control volume compositional formulation (since compositional simulation is one of the major focuses of this research), then we will show the general structure of the Jacobian matrix. In the following several sections, we will discuss the calculation of each individual part of the Jacobian matrix.

The control volume method is based on mass balance over a control volume. The mass accumulation in a control volume is equal to the flux (mass flowing in from its boundary) minus the production (mass produced from a well that maybe in the block, injection is negative). Control volume could be of any shape, for most unstructured grids, it is a polyhedron, like the one shown in Figure 2.3.



**Figure 2.3** A general polyhedron

The volume of this polyhedron is  $V$ , and it has  $ns$  surfaces. The rate of mass accumulation of component  $c$  can be expressed as

$$ACC_c = V \frac{\phi^{n+1} \sum_p (S_p \rho_p X_{cp})^{n+1} - \phi^n \sum_p (S_p \rho_p X_{cp})^n}{\Delta t} \quad (2.1)$$

where,  $X_{cp}$  is the mole fraction of component  $c$  in phase  $p$ .

The total flux is the sum of the fluxes across each of its  $ns$  surfaces, which can be written as

$$Flux_c = \sum_{s=1}^{ns} Flux_{s,c} \quad (2.2)$$

Flux across surface  $s$  is calculated as

$$Flux_{s,c} = \sum_p (v_p \rho_p X_{cp})_s \quad (2.3)$$

where,  $v_{p,s}$  is the phase volumetric flow rate across surface  $s$ . According to Darcy's law, it is equal to

$$v_{p,s} = (T \lambda_p \Delta \Phi_p)_s \quad (2.4)$$

where,  $T_s$  the transmissibility constant across surface  $s$ , and  $\Delta \Phi_{p,s}$  is the phase potential difference across surface  $s$ . By substituting Eqn. 2.4 and Eqn. 2.3 into Eqn. 2.2, we get the expression for the total flux:

$$Flux_c = \sum_{s=1}^{ns} [T_s \sum_p (\lambda_p \rho_p X_{cp} \Delta \Phi_p)_s] \quad (2.5)$$

The component mass produced from a well can be expressed as

$$Q_c^w = \sum_p (\rho_p X_{cp} q_p^w) \quad (2.6)$$

where,  $q_p^w$  is the phase volumetric production rate from the well, which can be written as

$$q_p^w = WI^w \cdot \lambda_p (p_p - p^w) \quad (2.7)$$

where,  $WI^w$  is the well index (Peaceman, 1978), a constant,  $p_p$  is the phase pressure of the well block and  $p^w$  is the wellbore pressure for the well in the well block. Substituting Eqn. 2.7 into Eqn. 2.6, we get

$$Q_c^w = WI^w \cdot \sum_p [\lambda_p \rho_p X_{cp} (p_p - p^w)] \quad (2.8)$$

Combining Eqn. 2.1, Eqn. 2.5 and Eqn. 2.8, we get the final control volume compositional formulation:

$$\begin{aligned} & V \frac{\phi^{n+1} \sum_p (S_p \rho_p X_{cp})^{n+1} - \phi^n \sum_p (S_p \rho_p X_{cp})^n}{\Delta t} \\ &= \left\{ \sum_{s=1}^{ns} [T_s \sum_p (\lambda_p \rho_p X_{cp} \Delta \Phi_p)_s] + WI^w \cdot \sum_p [\lambda_p \rho_p X_{cp} (p_p - p^w)] \right\}^{n,n+1} \end{aligned} \quad (2.9)$$

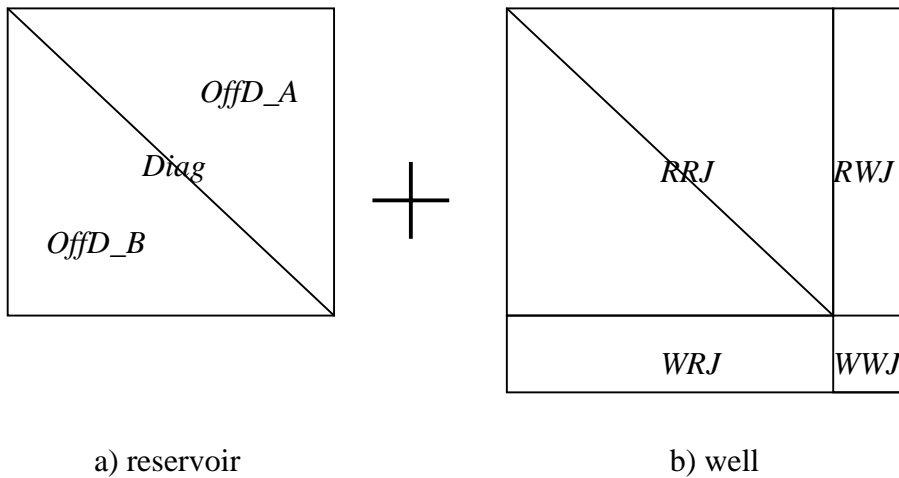
Note that each of the terms in the flux part and the well part can be evaluated at either timestep  $n$  or timestep  $n+1$ , depending on the implicitness of the model. Normally there is only one well passing through a control volume, if there are more, all of the well fluxes are summed.

Using the Newton-Raphson method, the Jacobian matrix is generated by calculating the derivatives of Eqn. 2.9 with respect to all of the unknowns (variables). In GPRS, the Jacobian matrix is calculated and stored separately for the reservoir part (accumulation and flux terms) and for the well part, and it is later pieced together only in the linear solver module.

The structure of the reservoir part of the Jacobian is shown in Figure 2.4a. Basically it consists of three arrays, *Diag*, *OffD\_A* and *OffD\_B*. *Diag* is the diagonal part, each of its entries is for one gridblock, and the location of each entry follows the cell list. *OffD\_A* records the sparse structure of the upper triangular part, and *OffD\_B* records the sparse structure of the lower triangular part. Each of their entries is for one connection, and the location of each entry follows the connection list. For example, for a connection between gridblock A and B ( $A < B$ ), *OffD\_A* stores the derivative of equations of gridblock A with respect to variables of gridblock B, and it is located at (A, B) of the Jacobian matrix. Similarly *OffD\_B* at (B, A) contains the derivatives of equations of gridblock B with

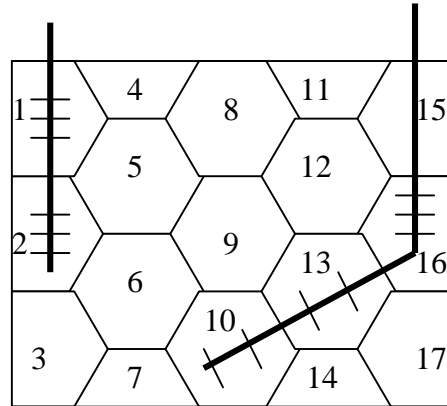
respect to variables of gridblock A. For multi-point flux,  $OffD\_A$  and  $OffD\_B$  will include more entries, we will discuss this in Section 2.4. For Jacobian calculations of the reservoir part, the accumulation part and the flux part are further separated. The accumulation part loops through the cell list, and it only adds terms in the  $Diag$  array; the flux part loops through the connection list, and it contributes to all three arrays. For multi-component system, each of the entries in these arrays is a small dense matrix.

The general structure of the well part of the Jacobian for a single well is shown in Figure 2.4b. Additional wells introduce additional rows and columns. Here besides the reservoir variables, each well may introduce an extra well variable, which is normally the well bottom hole pressure (BHP). From Figure 2.4b, we can see that the well part of the Jacobian consists of four arrays,  $RRJ$ ,  $RWJ$ ,  $WRJ$  and  $WWJ$ .  $RRJ$  represents the derivatives of well terms in Eqn. 2.9 with respect to the reservoir variables, which are always located at the diagonal, and only have entries at the well blocks.  $RWJ$  represents the derivatives of well terms in Eqn. 2.9 with respect to the well variable. This extra well variable is only necessary for some types of wells, such as wells under constant rate control, and in that case, an extra well equation is needed to complete the system.  $WRJ$  and  $WWJ$  represent the derivatives of this extra well equation with respect to the reservoir variables and the well variable respectively. Each entry in  $RRJ$  is normally a small dense matrix, for  $RWJ$ , it is a column vector, for  $WRJ$ , it is a row vector and for  $WWJ$ , it is a single value.



**Figure 2.4** General structure of the Jacobian matrix

In order to make the structure of the Jacobian matrix clearer, a simple two-well example is included here. The grid is shown in Figure 2.5. Well 1 is completed in gridblocks 1 and 2, and well 2 is completed in gridblocks 16, 13 and 10. The corresponding Jacobian matrix for this system is shown in Figure 2.6, where “A” represents a term generated by the accumulation part, “F” represents a term generated by the flux part, and “W” represented a term generated by the well part. The color in Figure 2.6 represents the storage, and same colored items are stored in the same array.



**Figure 2.5** A simple unstructured grid with wells

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	W1	W2
1	AFW	F		F	F													W	
2	F	AFW	F		F	F												W	
3		F	AF			F	F												
4	F			AF	F			F											
5	F	F		F	AF	F		F	F										
6		F	F		F	AF	F		F	F									
7			F			F	AF			F									
8				F	F			AF	F		F	F							
9					F	F		F	AF	F		F	F						
10						F	F		F	AFW			F	F				W	
11								F			AF	F			F				
12								F	F		F	AF	F		F	F			
13									F	F		F	AFW	F		F	F	W	
14										F			F	AF			F		
15											F	F			AF	F			
16												F	F		F	AFW	F	W	
17													F	F		F	AF		
W1	W	W																W	
W2								W			W				W				W

**Figure 2.6** Jacobian matrix structure for the system in Figure 2.5

## 2.3 Network Modeling

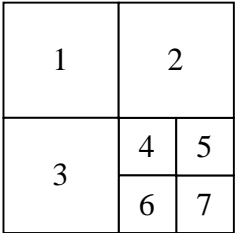
In Chapter 1, we have mentioned that there are two kinds of methods for grid representation, the block based approach and the connection based approach (Lim et al., 1994), also called network modeling. Traditionally, the block based approach has been used, where the connections (fluxes) are associated with blocks, but the number of connections for a gridblock may vary depending on the geometry of the gridblock, as in the case for unstructured grids. In addition, local grid refinement may increase the number of connections. As a result, each gridblock may have different number of neighbors. An array keeping track of the connected gridblocks has to be dynamically dimensioned for each gridblock, (or dimensioned for the largest possible case). Figure 2.7 shows a simple example of the conventional gridblock tracking array. The number of connections for each gridblock varies between two and four. From Figure 2.7, we can see that the block based connection information is also redundant, each of the connections actually appears two times. The number of connections is even higher if a nine-point scheme is used (Yanosik and McCracken, 1979; Sammon, 1991). In general, the block-based approach becomes very inconvenient for these cases.

		<u>Block</u>	<u>Connections</u>
1	2	1	2, 3
		2	1, 4, 5
3	4	3	1, 4, 6
		4	2, 3, 5, 6
	5	5	2, 4, 7
		6	3, 4, 7
	6	7	5, 6

**Figure 2.7** Gridblock connection array when the block based approach is used

The computation of flux terms is only necessary when pairs of gridblocks are connected. Furthermore, it is obvious that a positive flux into a given gridblock automatically implies a negative flux for the other connected block. This means that the flux is only a property of individual connections, not gridblocks. So the key idea in the connection based approach is to focus on connections instead of gridblocks when

computing flux terms. The first major task in the connection-based approach is to establish an array map or data structure containing all possible pairs of connections. As an example, the gridblock connections in Figure 2.7 can be rearranged into a list of connection pairs, shown in Figure 2.8. The first gridblock index in a connection pair is always smaller than the second gridblock index, in order to avoid redundancy. Once the connection array has been established, the computation algorithm in the flux part is exactly the same for any dimensional problems, since a connection is basically a line connecting two points, which is always one-dimensional. This connection-based approach is very convenient for unstructured grids, instead of tracking the number of connections for each gridblock, we track the connections for the whole domain.

	<u>Connection No.</u>	<u>Connected pairs</u>
	1	1, 2
	2	1, 3
	3	2, 4
	4	2, 5
	5	3, 4
	6	3, 6
	7	4, 5
	8	4, 6
	9	5, 7
	10	6, 7

**Figure 2.8** Connection array when the connection based approach is used

The connection-based approach is used in GPRS, and the following modifications are made to make it more suitable for unstructured grid simulation and modeling of wells and surface facilities:

- **Add in cell list** Traditionally, gridblocks (cells) are represented by three-dimensional (i, j, k) indices, instead of that, a one-dimensional index (cell list) is used in GPRS, which can be calculated as  $i + (j + k \cdot n_y) \cdot n_x$  for structured grids, where  $n_x$  and  $n_y$  are number of gridblocks in  $x$  and  $y$  directions. Doing that, the computational algorithm in the accumulation part is exactly the same for any dimensional problem, and the whole simulator becomes independent of the grid type and dimensions. This

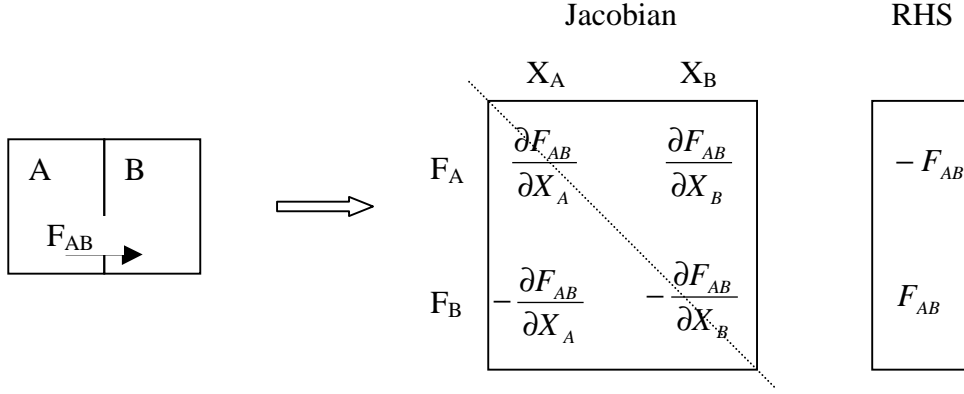


approach also saves some memory by reducing a three-dimensional indexing array to a one-dimensional indexing array.

- **Reservoir only connection list** In Lim et al. (1994), the connections are between two nodes, each node can be a reservoir node (gridblock), a well node (wellbore) or a surface node (surface facility), which is the most general connection based approach. But the reservoir, the well and the surface facilities, each have their own characteristics. This leads to different kinds of calculations and different kinds of structures in the Jacobian matrix for each part of the system. Hence the connection list in GPRS is only for the reservoir part, the well part and the surface facility part are treated separately. The advantage is that the structure of the Jacobian matrix is preserved, and this structure can be used later by the linear solver to improve its efficiency. This approach also makes the algorithm clearer and easier to understand.

The procedure for the reservoir part of the Jacobian matrix and residual calculation using this modified connection based approach is as follows:

1. First, loop through the cell list, and evaluate the accumulation terms and their derivatives for each gridblock. The accumulation terms are assigned to the residual, and the derivatives are assigned to the diagonal of the Jacobian matrix at the corresponding gridblocks.
2. Next, loop through the connection list, calculate the upstream direction for each phase and evaluate the flux terms and their derivatives for each connection. The flux terms are added to the residual of the connected blocks, one is positive and the other is negative, the derivatives are added to the correct locations in the Jacobian matrix, as shown in Figure 2.9 for a two-point flux calculation. Flux  $F_{AB}$  is from gridblock A to gridblock B,  $X_A$  and  $X_B$  are the variables at gridblock A and B, and  $F_A$  and  $F_B$  are the equations at gridblock A and B.  $\frac{\partial F_{AB}}{\partial X_A}$  and  $-\frac{\partial F_{AB}}{\partial X_B}$  are stored in *Diag*, and  $\frac{\partial F_{AB}}{\partial X_B}$  is stored in *OffD\_A*, and  $-\frac{\partial F_{AB}}{\partial X_A}$  is stored in *OffD\_B*.



**Figure 2.9** Jacobian structure for two-point flux derivatives

## 2.4 Multi-Point Flux Representation and Implementation

In the last decade, with the development of unstructured grids and upscaling methods, more and more simulations are using non-orthogonal grids and full tensor permeabilities. For these kinds of simulations, multi-point flux approximations (Gunasekera et al., 1998; Verma and Aziz, 1996) are necessary. At the same time, computers are becoming more powerful, and new techniques (e.g. Edwards, 1998) are being developed to reduce the cost of multi-point flux simulation, both of them make multi-point flux approximation more practical.

The connection-based approach was originally proposed for two-point flux. We generalize it to multi-point flux. In this section, first we will compare the differences between two-point flux and multi-point flux calculations, then we will show the algorithm used for multi point flux calculations. Two-point flux and multi-point flux are different in three areas: representation, upstream direction calculation and Jacobian matrix structure.

- **Representation** In two-point flux, a connection between two gridblocks only depends on these two gridblocks, generally, each connection can be represented in C++ by “*int A, int B, double T*”, which means that there is flow (connection) between gridblocks A and B and the transmissibility constant for this connection is T. Two-point flux can be calculated as

$$Flux_{c,p} = T_{AB} (\lambda_p \rho_p X_{cp})_{A|B} (\Phi_{p,B} - \Phi_{p,A}) \quad (2.10)$$

But in multi-point flux, a connection between two gridblocks may depend on  $np$  ( $\geq 2$ ) gridblocks. For structured grids,  $np$  is fixed, such as  $np=6$  for two-dimensional Cartesian grid and  $np=18$  for three-dimensional Cartesian grid (Verma, 1996). But for unstructured grids, it is not fixed, and there is no upper limit to it. A multi-point unstructured connection is represented by “*int np, int \*A, double \*T*” in GPRS, which means that there is a connection between gridblocks  $A[0]$  and  $A[1]$ , and its flux depends on  $np$  gridblocks, which are  $A[0], A[1] \dots A[np-1]$ , and the transmissibility constant assigned to each gridblock is  $T[0], T[1] \dots T[np-1]$ . Multi-point flux can be calculated as

$$Flux\_MP_{c,p} = (\lambda_p \rho_p X_{cp})_{A[0]:A[1]} \sum_{l=0}^{np-1} (T[l] \Phi_{p,A[l]}) \quad (2.11)$$

There is no flow through a connection when phase potentials are equal, in order to guarantee that, the transmissibility constants should satisfy

$$\sum_{l=0}^{np-1} T[l] = 0 \quad (2.12)$$

Using Eqn. 2.12 to remove  $T[0]$  from Eqn. 2.11, we can get

$$Flux\_MP_{c,p} = (\lambda_p \rho_p X_{cp})_{A[0]:A[1]} \sum_{l=1}^{np-1} (T[l] \cdot (\Phi_{p,A[l]} - \Phi_{p,A[0]})) \quad (2.13)$$

Each individual term in the summation of Eqn. 2.13 is called a sub-connection. Comparing Eqn. 2.13 with Eqn. 2.10, we notice that they are quite similar. If  $np=2$ , and we define  $A[0]$  as A,  $A[1]$  as B, and  $T[1]$  as T, then Eqn. 2.13 reduces to Eqn. 2.10. This is because two-point flux is just a special case of multi-point flux, where each connection only has one sub-connection.

- **Upstream direction calculation** In Eqn. 2.10 and Eqn. 2.13,  $\lambda_p \rho_p X_{cp}$  should be evaluated using conditions in the upstream block for each connection and for each phase. In two-point flux, the phase upstream direction is solely determined by the phase potential difference, and  $\lambda_p \rho_p X_{cp}$  is evaluated as

$$(\lambda_p \rho_p X_{cp})_{A|B} = \begin{cases} (\lambda_p \rho_p X_{cp})_A, & \text{if } \Phi_{p,B} - \Phi_{p,A} < 0 \\ (\lambda_p \rho_p X_{cp})_B, & \text{if } \Phi_{p,B} - \Phi_{p,A} \geq 0 \end{cases} \quad (2.14)$$

In multi-point flux, the phase upstream direction is determined by  $\sum_{l=0}^{np-1} (T[l] \cdot \Phi_{p,A[l]})$

or  $\sum_{l=1}^{np-1} (T[l] \cdot (\Phi_{p,A[l]} - \Phi_{p,A[0]}))$ , and  $\lambda_p \rho_p X_{cp}$  is evaluated as

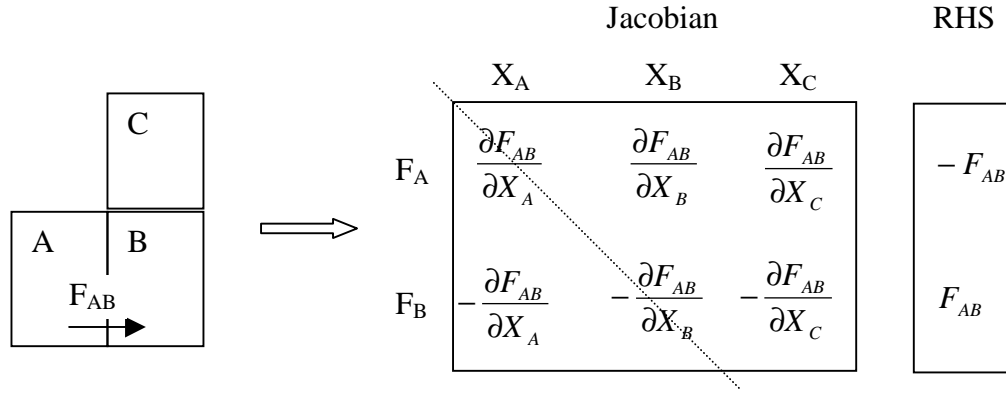
$$(\lambda_p \rho_p X_{cp})_{A[0]|A[1]} = \begin{cases} (\lambda_p \rho_p X_{cp})_{A[0]}, & \text{if } \sum_{l=0}^{np-1} (T[l] \cdot \Phi_{p,A[l]}) < 0 \\ (\lambda_p \rho_p X_{cp})_{A[1]}, & \text{if } \sum_{l=0}^{np-1} (T[l] \cdot \Phi_{p,A[l]}) \geq 0 \end{cases} \quad (2.15)$$

Note,  $\lambda_p \rho_p X_{cp}$  can only take the value at gridblock A[0] or A[1], not the other gridblocks. Eqn. 2.15 can be reduced to Eqn. 2.14 if there are only two points (T[1] is always positive for flux between A[1] and A[0]).

- **Jacobian matrix structure** If a multi-point flux approximation is used, we will have many more off diagonal terms in the Jacobian matrix, and the computational cost will be much higher. For structured grids, bands for the Jacobian matrix will increase from 5 to 9 in two dimension, and from 7 to 27 in three dimension. For unstructured grids, the structure of the Jacobian matrix becomes more complicated with more non-zero terms.

We have reviewed the differences between two-point flux and multi-point flux, next we will show the reservoir part of the Jacobian matrix and residual calculation for the multi-point flux approximation. This is done by looping through the connection list. For each connection, we first calculate the phase upstream directions, then loop through its sub-connections and evaluate the flux terms and their derivatives for each sub-connection. The flux terms are added to the residual of the connected blocks (A[0] or A[1]), one is positive and the other is negative, the derivatives are added to the correct locations in the Jacobian matrix, as shown in Figure 2.10 for a multi-point flux calculation. Flux  $F_{AB}$  is from gridblock A to gridblock B, and one of its sub-connections is related to gridblock C.  $X_A$ ,  $X_B$  and  $X_C$  are the variables at gridblock A, B and C.  $F_A$  and

$F_B$  are the equations at gridblock A and B.  $\frac{\partial F_{AB}}{\partial X_A}$  and  $-\frac{\partial F_{AB}}{\partial X_B}$  are stored in *Diag*,  $\frac{\partial F_{AB}}{\partial X_B}$  and  $\frac{\partial F_{AB}}{\partial X_C}$  are stored in *OffD\_A*, and  $-\frac{\partial F_{AB}}{\partial X_A}$  and  $-\frac{\partial F_{AB}}{\partial X_C}$  are stored in *OffD\_B*. Note, in the multi-point flux calculations, the number of entries in *OffD\_A* and *OffD\_B* equals the total number of sub-connections, and *OffD\_A* is no longer strictly upper triangular and *OffD\_B* is no longer strictly lower triangular.



**Figure 2.10** Jacobian structure for multi-point flux derivatives

## 2.5 Well Treatment

We have discussed how the reservoir part of the Jacobian matrix and RHS are calculated in GPRS, now we will discuss the well part. From Section 2.2, we know that the well part of the Jacobian includes four arrays, *RRJ*, *RWJ*, *WRJ* and *WWJ*. *RRJ* and *RWJ* are the derivatives of well terms in flow equations with respect to the reservoir variables and the well variable respectively. *WRJ* and *WWJ* are the derivatives of the extra well equation with respect to the reservoir variables and the well variable respectively. *RRJ* is always needed, while the existence of *RWJ*, *WRJ* and *WWJ* depends on well control. In this section, we will first discuss the calculation of *RRJ*, then we will discuss when an extra well equation and a corresponding well variable are needed. Well equations used in GPRS for the black-oil and the compositional models will be presented next.

*RRJ* always exists, and it is a block diagonal matrix with only non-zero terms at the well blocks. After it is calculated, it can be directly added to the reservoir part of the

Jacobian without introducing any extra non-zero terms. In Section 2.2, the well equation was written as

$$Q_c^W = WI^W \cdot \sum_p [\lambda_p \rho_p X_{cp} (p_p - p^W)] \quad (2.8)$$

where,  $WI^W$  is the well index,  $p_p$  is the phase pressure of the well block and  $p^W$  is the wellbore pressure. If this well has multiple perforations, we need to relate  $p^W$  to the pressure at a reference depth. We assume variable gradient and no friction in GPRS, and use the following relation to calculate the wellbore pressure at well block i:

$$\begin{aligned} p_i^W &= p_{i-1}^W + (\rho_{i-1}^W + \rho_i^W)g(D_i - D_{i-1}) \\ &= p_{i-2}^W + (\rho_{i-2}^W + \rho_{i-1}^W)g(D_{i-1} - D_{i-2}) + (\rho_{i-1}^W + \rho_i^W)g(D_i - D_{i-1}) \\ &= \dots\dots\dots \\ &= p_0^W + g \sum_{l=1}^i (\rho_{l-1}^W + \rho_l^W)(D_l - D_{l-1}) \end{aligned} \quad (2.16)$$

Here the reference depth is defined at the first well block, and the wellbore pressure at this depth ( $p_0^W$ ) is normally called the well bottom hole pressure (BHP).  $\rho_i^W$  is the total fluid density within the wellbore at well block i, which can be calculated as (Nolen, 1990)

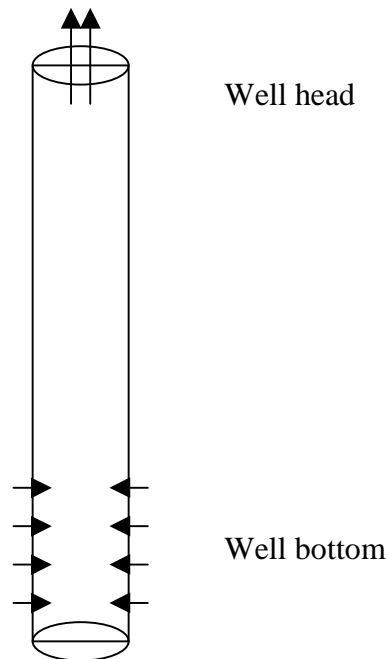
$$\rho_i^W = \frac{\sum_p (\rho_p q_p^W)_i}{\sum_p q_{p,i}^W} \quad (2.17)$$

$q_{p,i}^W$  is the phase volumetric production rate of this well at well block i, which is calculated from Eqn. 2.7. Combining Eqn. 2.8, Eqn. 2.16 and Eqn. 2.17, we can easily see that, if the well bottom hole pressure  $p_0^W$  is fixed, well term will only depend on the reservoir variables. If needed, the well bottom hole pressure will be the extra well variable.  $RRJ$  is the derivatives of Eqn. 2.8 with respect to the reservoir variables for a fixed well bottom hole pressure.

The existence of  $RWJ$ ,  $WRJ$  and  $WWJ$  depends on whether we need an extra well equation and well variable. For a BHP controlled well, where  $p_0^W$  is fixed, Eqn. 2.8 only depends on the reservoir variables,  $RWJ$  is zero, and there is no need for  $WRJ$  and  $WWJ$ .

*RRJ* by itself is already complete. After we solved for the reservoir variables, the well phase production rates can be explicitly calculated from Eqn. 2.7. For other types of well controls, such as constant phase volumetric flow rate control (at well head or well bottom), the BHP of a well is not fixed, we have the extra well variable, and we need an extra well equation to complete the system. The well equation depends on both the desired control at the well and the fluid model (black-oil or compositional). Here we only review the well equations used in GPRS.

Currently, only two types of well controls are implemented in GPRS, BHP control and constant phase volumetric flow rate (at standard conditions) control. An extra well equation and variable are needed for constant phase flow rate controlled wells. The extra well variable is always the well bottom hole pressure, the extra well equation is different for black-oil and compositional models, and both of them are based on component mass balance within the well bore. As shown in Figure 2.11, there is flow from the well bottom to the well head, and the well could have multiple perforations.



**Figure 2.11** Wellbore flow

Assuming steady state flow in the well, the component mass flow rate at the well bottom (summation for all well blocks) should be equal to the component mass flow rate at the well head, which can be expressed as

$$\sum_l \sum_p W I_l \lambda_{p,l} \rho_{p,l} X_{c,p,l} (p_{p,l} - p_l^w) = Q_c^{WH} \quad (2.18)$$

where,  $Q_c^{WH}$  is the mass flow rate of component  $c$  at the well head. Since the extra well equation is different for black-oil model and compositional model, they are discussed separately, using a constant oil phase flow rate controlled well as an example.

- **Black-oil model** Components in black-oil systems are defined as phases at standard conditions (well head). In order to avoid confusion between components and phases, we will use a superscript “-” to indicate components whenever necessary. At the well head, black-oil components and phases are equivalent to each other, and the component mass flow rate should equal the phase mass flow rate, which can be expressed as

$$Q_o^{WH} = Q_o^{WH} = \rho_o^{WH} q_o^{WH} = \rho_{o,SC} \cdot q_o^{WH} \quad (2.19)$$

Substituting Eqn. 2.18 into Eqn. 2.19 and rearranging, we get

$$\frac{\sum_l \sum_p W I_l \lambda_{p,l} \rho_{p,l} X_{o,p,l} (p_{p,l} - p_l^w)}{\rho_{o,SC}} - q_o^{WH} = 0 \quad (2.20)$$

where,  $q_o^{WH}$  is the fixed oil phase volumetric flow rate at the well head, and  $\rho_{o,SC}$  is the oil component density at the standard condition, which is an input property. For black-oil models, Eqn. 2.20 is the extra well equation used in GPRS.

- **Compositional model** Things become a bit more complicated for the compositional model. Here, well head is actually a separator which operates at standard conditions. The input to this separator is the total mass flow from the well (summation of  $Q_c^{WH}$  over all  $c$ ), the output is the individual phase flow at standard conditions. For constant oil phase flow rate, we have the following relation for the oil phase mass flow rate:

$$l^{SP} \cdot \sum_c Q_c^{WH} = \rho_o^{SP} \cdot q_o^{SP} \quad (2.21)$$



Substituting Eqn. 2.18 into Eqn. 2.21 and rearranging, we get

$$\sum_l \sum_p W I_l \lambda_{p,l} \rho_{p,l} (p_{p,l} - p_l^W) \cdot \left( \frac{l^{SP}}{\rho_o^{SP}} \right) - q_o^{SP} = 0 \quad (2.22)$$

where,  $q_o^{SP}$  is the fixed oil phase volumetric flow rate (output of the separator), and  $\rho_o^{SP}$  is the oil phase density at the separator, and  $l^{SP}$  is the hydrocarbon liquid (oil) mole fraction at the separator. To get  $\rho_o^{SP}$  and  $l^{SP}$ , the overall hydrocarbon component mole fractions at the separator are first calculated as

$$z_c = Q_c^{WH} / \sum_{i=1}^{n_h} Q_i^{WH} \quad (2.23)$$

then a flash calculation is performed at standard condition and fixed  $z_c$ . For compositional models, Eqn. 2.22 is the extra well equation used in GPRS.

Water component is fully separated from the hydrocarbon components in GPRS. So, for fixed water flow rate controlled wells, the extra well equation is always

$$\frac{\sum_l W I_l \lambda_{w,l} \rho_{w,l} (p_{p,l} - p_l^W)}{\rho_{w,SC}} - q_w^{WH} = 0 \quad (2.24)$$

For injectors, each phase mobility is replaced by the total mobility or the injected phase volume fraction weighted phase mobility (Almehaideb and Aziz, 1989; Nolen, 1990; Coats et al., 1998). In GPRS, for multiphase injection the mobility of the injected fluid is calculated as the volume fraction ( $E_p$ ) of that fluid multiplying the total mobility of fluid in the injector block (Almehaideb and Aziz, 1989), which can be written as

$$\lambda_p^{inj} = E_p \sum_p \lambda_p \quad (2.25)$$

The component mass balance within the wellbore is only for the injected components, and the total mass flow rate is only the summation over the injected components, correspondingly  $RRJ$  and  $RWJ$  are only calculated for the flow equations of the injected components.

Sometimes, after the reservoir part of the Jacobian and the well part of the Jacobian have been combined,  $RWJ$  can be eliminated without introducing any extra terms, such as for single block wells with constant rate control. With this, we can solve for the reservoir variables first and explicitly update the well variable later. The details of this operation are shown in Section 3.2. For multi-block wells with constant rate control,  $RWJ$  can also be eliminated, but it will introduce some extra non-zero terms in  $RRJ$ . For this reason, it is preferable not to perform elimination on  $RWJ$  for this kind of wells.

## 2.6 Flash Calculation and Treatment of Phase Disappearance and Reappearance

Flash calculations are necessary in compositional simulation, and for this we need to select an equation of state and phase equilibrium relations. In this section, first we will discuss the equation of state and phase equilibrium relations used in GPRS, then we will discuss the two different roles that flash calculations play in GPRS, finally we will show the treatment of phase disappearance and reappearance in GPRS.

The Cubic equation of state used in GPRS can be written as (e.g., Nghiem et al., 1983; Walas, 1985)

$$Z^3 + sZ^2 + qZ + r = 0 \quad (2.26)$$

$$\text{where, } \begin{cases} s = (u-1)B - 1 \\ q = A + (w-u)B^2 - uB \\ r = -AB - wB^2 - wB^3 \end{cases}$$

Parameters  $u$  and  $w$  depend on the cubic equation of state selected. For the Peng-Robinson EOS (Peng and Robinson, 1976),  $u=2$  and  $w=-1$ .  $Z$  is the phase compressibility factor, which is one of the three roots of Eqn. 2.26. For the oil phase, the minimum real root is used, and for the gas phase, the maximum real root is used.

At phase equilibrium, we have the following relation:

$$f_{c,o} = f_{c,g}, \quad c = 1, \dots, n_h \quad (2.27)$$

where,  $f_{c,o}$  and  $f_{c,g}$  are the fugacities of component  $c$  in the oil and gas phases. For the Cubic equation of state they are calculated as (Walas, 1985)

$$f_{c,p} = pX_{c,p}e^{\Phi_c}, \quad p = o \text{ or } g, \quad (2.28)$$

$$\Phi_c = \frac{b_c}{b}(Z-1) - \ln(Z-B) - \frac{A}{B\sqrt{u^2-4w}} \left( \frac{b_c}{b} - \frac{1}{a} \frac{\partial a}{\partial X_{c,p}} \right) \ln \left[ \frac{2Z+B(u+\sqrt{u^2-4w})}{2Z+B(u-\sqrt{u^2-4w})} \right] \quad (2.29)$$

$$\text{where, } \begin{cases} a = \sum_{j=1}^{n_h} \sum_{i=1}^{n_h} X_{i,p} X_{j,p} (1-k_{i,j}) \sqrt{a_i a_j} \\ b = \sum_{c=1}^{n_h} X_{c,p} b_c \end{cases} \quad \text{and} \quad \begin{cases} A = \frac{ap}{R^2 T^2} \\ B = \frac{bp}{RT} \end{cases} \quad (2.30)$$

Appendix C defines all parameters and provides the formulas and derivatives used for flash calculations.

Eqn. 2.27 is used in two ways in GPRS, as a constraint equation to relate the secondary variables to the primary variables and as a separate flash routine to check the state of hydrocarbon phase in a block. These two tasks are discussed next.

- **As a constraint equation** Flow equations are the primary equations, they depend on both the primary variables and the secondary variables. To form a complete system, we need some constraint equations to remove the secondary variables from the primary equations. Phase equilibrium relations (Eqn. 2.27) are our secondary equations, which also depend on both the primary variables and the secondary variables. They can be used as the constraint equations to build relations between the secondary variables and the primary variables. This role for flash calculations is only necessary when both hydrocarbon phases exist in a gridblock. For gridblocks with only one hydrocarbon phase, half of the phase variables disappear, and the primary equations depend only on the primary variables. For example, for a gridblock with only gas and water phases, the mass balance equations will only depend on  $p$ ,  $S_g$  and  $y_c$ ,  $c=1, \dots, n_h-1$ . When this is the case, there is no need for constraint equations.
- **As a separate flash routine** For gridblocks with only one hydrocarbon phase, the primary equations depend only on the primary variables, and there are no need of

secondary equations. But sometimes, a single hydrocarbon phase can change back to both hydrocarbon phases, and the secondary equations are needed again. So we need to monitor the state of hydrocarbon phases when a gridblock has only one hydrocarbon phase. This is done by a separate flash routine. A simple successive substitution flash routine is used in GPRS, which is similar to the one shown in Subsection 1.2.2. To speed up the convergence, the values of  $K_c$  and  $l$  at the old Newton iteration are used for the initial guess.

From the above discussion, we can see that flash calculations play different roles under different situations. In order to distinguish these situations, we need to detect the disappearance and reappearance of hydrocarbon phases. For models using the natural variables (variable Type A), treatment of phase disappearance and reappearance is also needed for choosing the correct phase variables. Phase disappearance and reappearance are detected in GPRS and treated as described below:

- **Phase disappearance** When a gridblock has two hydrocarbon phases, for each Newton iteration, we check the saturation solutions. If either  $S_o$  or  $S_g$  is negative, the corresponding hydrocarbon phase has disappeared. In this case, before initialing the next Newton iteration, we first set the negative saturation to zero, then assign the overall hydrocarbon component mole fractions ( $z_c$ ) to component mole fractions in the oil or gas phases ( $x_c$  and  $y_c$ ), finally mark this gridblock as a single hydrocarbon phase gridblock.
- **Phase reappearance** When a gridblock has only one hydrocarbon phase, at the end of each Newton iteration, we do a negative flash (Whitson and Michelson, 1986), pressure and overall hydrocarbon component mole fractions are the inputs. If  $l$  is between 0 and 1, then both hydrocarbon phases exist, and before initializing the next Newton iteration, we first assign the hydrocarbon phase saturations as

$$S_o = \frac{1 - S_w}{1 + \left(\frac{1}{l} - 1\right) \frac{\rho_o}{\rho_g}} \quad (2.31)$$

$$S_g = 1 - S_o - S_w \quad (2.32)$$

then set component mole fractions in the oil and gas phases ( $x_c$  and  $y_c$ ) from the results of flash calculation, finally mark this gridblock as a both hydrocarbon phase gridblock.

In compositional simulations, flash calculation is a very important part, and it could account for up to 30% of the total running time. The flash routine used in GPRS is quite simple, and a more robust and efficient flash routine is needed.

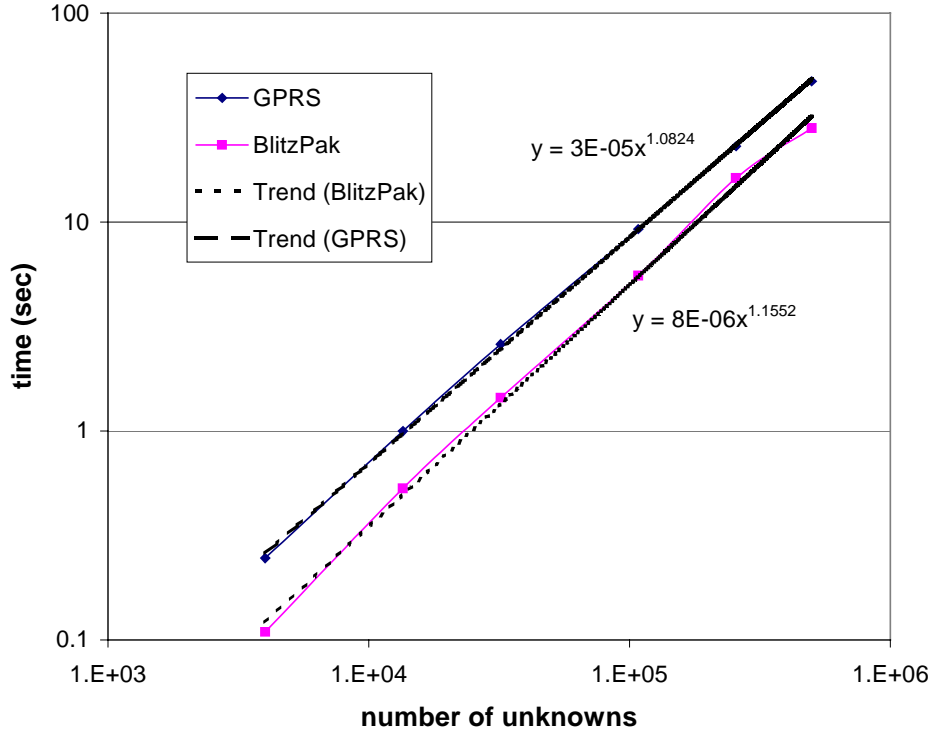
## 2.7 Linear Solver

In GPRS, the Jacobian matrix is calculated and stored separately for the reservoir part and for the well part, and it is later pieced together only in the linear solver part. This approach basically separates the solving of the linear system from the building of the linear system, and creates a separate linear solver module. All we need to do is to transform the Jacobian structure of GPRS into the specific format that a linear solver requires. In this way we can easily interface with available linear solvers, such as BlitzPak (Landmark, 1998), we can also develop and use our own solvers.

Both direct and iterative solvers are used in GPRS. The direct solvers included are a full matrix solver and a band matrix solver, both are quite slow and memory consuming, and can only be used for small systems. For iterative solvers, we use a Generalized Minimum Residual (GMRES) (Saad and Schultz, 1986) solver, which is suitable for any sparse matrix system. For large problems, iterative solvers are memory efficient and much faster than direct solvers. For iterative solvers, preconditioners are used to speed up the convergence. The performance of iterative solvers depends on the robustness of the preconditioners. The following preconditioners are included in GPRS, ordered according to their robustness: a) diagonal scaling; b) block diagonal scaling; c) Incomplete LU decomposition (ILU0) (e.g. Behie and Forsyth, 1983); d) Algebraic Multi-Grid (AMG) (Stueben, 1983), which is only suitable for the pressure system; and e) Constrained Pressure Residual (CPR) (Wallis et al., 1985), which is our best preconditioner. GPRS also has an interface to BlitzPak, which is a commercial linear solver package from Landmark Graphics Corporation. BlitzPak is only suitable for structured grids, where the

Jacobian matrix has up to 7 bands, while GMRES (Saad and Schultz, 1986) in GPRS is suitable for any grid and any sparse matrix system. For GPRS, BlitzPak is the fastest for structured grids, and CPR preconditioned GMRES is the best for unstructured grids.

For the preconditioners used in GPRS, only CPR is specially designed for reservoir simulation equations, and it is also the best preconditioner in GPRS. CPR is a two stage preconditioner. In the first stage we decouple the full system into a pressure part and a saturation part using an IMPES-like decoupling process, and solve for pressure from the pressure part (AMG preconditioned GMRES is used for the pressure solve), then we use the pressure solutions to reduce the residual of the full system. In the second stage we start from the reduced residual and work on the remaining decoupled system, and use a local preconditioner, such as ILU0, to get the full solutions for the reduced residual. The final solution is the first stage pressure solutions plus the second stage full solutions. CPR performs very well for all kinds of problems, including problems that are hard for IMPES. CPR is more suitable for fully implicit systems.



**Figure 2.12** Performance comparison between GPRS solver and BlitzPak

Here we compare the performance of CPR preconditioned GMRES (GPRS solver) with the performance of BlitzPak on different size problems. The results are shown in Figure 2.12. We use a Cartesian grid, 4 component compositional problem. The problem is solved fully implicitly. In Figure 2.12, we provide the trendlines using power relations. For the GPRS solver, the power is about 1.08, for BlitzPak, the power is about 1.15. The costs of both solvers are roughly linear with the number of unknowns, which is the best we can get for a linear solver. The constant for the GPRS solver ( $3E-5$ ) is larger than that for BlitzPak ( $8E-6$ ). This difference is due to the following two reasons: first GMRES in GPRS does not make use of the 7 band structure of the Jacobian matrix, second the components of the GPRS solver are from public domain software, which may not be very efficient. Anyway, for large models, GPRS solver only takes about 50% more time than BlitzPak. Considering that the GPRS solver is suitable for any grid system and any sparse matrix, we can conclude that the solver in GPRS is reasonably efficient and robust.

In the solver part of GPRS, most of the components are from different public domain software, such as the direct solvers are from LAPACK (Anderson, 1999), GMRES is from Iterative Math Library (IML) (Dongarra et al., 1996), ILU0 is from SparseLib++ (Pozo et al., 1996), a sparse matrix package, and AMG is from GMD software (Ritzdorf, 1991).

The solvers and preconditioners in GPRS are appropriate for fully implicit and IMPES systems (CPR is used to precondition the fully implicit system, and AMG is used to precondition the IMPES system). But, for an adaptive implicit system, such as AIM, most of the gridblocks are IMPES, only a small number of gridblock are fully implicit, CPR loses its efficiency (spends too much time on the pressure solve at the first stage, while the whole system is only a little bit larger than the pressure system), and AMG is only suitable for the pressure system. More works is needed to develop suitable solvers for AIM method.

For unstructured grids, the ordering of the cell list is quite important for the efficiency of linear solvers, as shown in Section 2.1. Currently, there is no reordering done to the cell list in GPRS. Additional work is needed to develop good ordering methods for the cell list.

## 2.8 Timestep Control and Reservoir Initialization

Fully implicit models are unconditionally stable, and we can have a timestep of any size. But in practice, in order to control the number of Newton iterations for each timestep, a reasonable timestep size is required. For models that are not fully implicit, such as IMPES, their maximum timestep sizes are limited by their stability. Section 3.3 derives all of the stability criteria used in GPRS.

In GPRS, a maximum timestep size  $\Delta t_{MAX}$  and a minimum timestep size  $\Delta t_{MIN}$  are specified by the user, and the timestep sizes for all models must satisfy Eqn. 2.33 at all times:

$$\Delta t_{MIN} \leq \Delta t \leq \Delta t_{MAX} \quad (2.33)$$

The increasing or decreasing of timestep size from timestep  $n$  to timestep  $n+1$  is governed by the following relation (Aziz and Settari, 1979):

$$\Delta t^{n+1} = \Delta t^n \left[ \frac{(1 + \omega)\eta_i}{\delta_i + \omega\eta_i} \right]_{\min \text{ over } i} \quad (2.34)$$

where  $i$  refers to the gridblock number,  $\eta_i$  is the specified desired change,  $\delta_i$  is the change over  $\Delta t^n$  and  $\omega$  is a tuning factor with a value between 0 and 1. Normally  $\eta_i$  and  $\delta_i$  have different values for different variables, so a different  $\Delta t^{n+1}$  is calculated for each variable and the minimum of all  $\Delta t^{n+1}$  is used for the next timestep. Typical values of  $\eta$  used in GPRS are:

$$\begin{aligned} \eta_p &= 200 \text{ Psia}, & \text{for pressure} \\ \eta_s &= 0.2, & \text{for saturations} \\ \eta_x &= 0.02 & \text{for component mole fractions} \end{aligned}$$

The timestep sizes of fully implicit models are decided solely by Eqn. 2.33 and Eqn. 2.34. The timestep sizes of partially implicit models are governed not only by Eqn. 2.33 and Eqn. 2.34, but also by their stability criteria.

At the start of simulation, we need to assign initial values to reservoir variables (pressure, saturations and component mole fractions). For black-oil models, the



procedure mentioned in Aziz (1996) is used, and it guarantees initial equilibrium of reservoir fluids. For compositional models, the following procedure is used:

1. Assign the same initial pressures (pressure at a reference depth) to all gridblocks.
2. Assign initial water saturations for all gridblocks. If a gridblock is below the water-oil contact (WOC), assign 1, otherwise assign connate water saturation.
3. Assign initial overall compositions  $z_c$  for all gridblocks. Initial overall compositions are inputted as an overall composition vs. depth table, and a table look up is performed for each gridblock to decide its initial overall compositions.
4. Assign initial component mole fractions in the oil and gas phases,  $x_c$  and  $y_c$  for all gridblocks. A flash calculation is performed at constant pressure, temperature and overall compositions  $z_c$  for each gridblock. The resulting  $x_c$  and  $y_c$  are assigned to each gridblock. At the same time, the oil and gas phase densities,  $\rho_o$  and  $\rho_g$ , and the hydrocarbon liquid mole fraction  $l$  are calculated for each gridblock.
5. Assign initial oil and gas saturations for all gridblock as

$$S_o = \frac{1 - S_w}{1 + \left(\frac{1}{l} - 1\right) \frac{\rho_o}{\rho_g}} \quad (2.35)$$

$$S_g = 1 - S_o - S_w \quad (2.36)$$

6. Reassign initial pressures for all gridblock according to depth:

$$p_i = p^0 + \sum_p (\rho_p S_p)_i g (D_i - D^0) \quad (2.37)$$

where  $p^0$  is the pressure at reference depth  $D^0$ .

If necessary, step 4-6 are repeated to achieve better initial equilibrium. The effect of capillary pressure is ignored in this procedure.

If the initial overall composition vs. depth table reflects real conditions in the reservoir, the calculated initial equilibrium is good. Otherwise the initial condition will be far from equilibrium, such as for the case when the same initial overall composition is used for all depths.



## Chapter 3

### General Formulation Approach

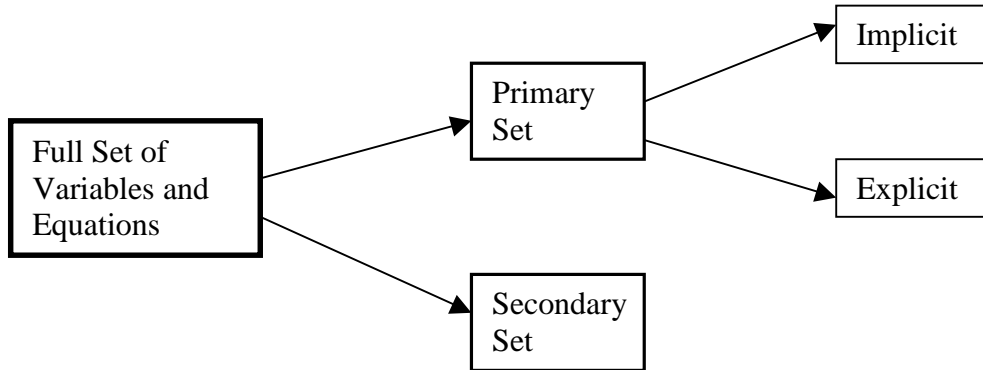
One of the objectives of this research is to evaluate the performance of different isothermal compositional models. In order to make the implementation of these models easy and consistent, a new General Formulation Approach is developed, which can be used to derive any kind of model (different equation and variable selections and different implicit levels). With this approach, we can minimize differences among different models, quickly change from one model to another model, and evaluate their performance in a consistent manner. This General Formulation Approach also enables us to develop and try out new models, and it is also suitable for other types of simulators, such as thermal simulators. This new approach is designed to solve a general nonlinear equation set. In this study, we have only explored its application to isothermal compositional models.

To better understand the steps involved in this General Formulation Approach, it is worthwhile to first review some of the important points about compositional simulation discussed in Chapter 1:

- Reservoir simulation involves a large number of coupled nonlinear equations. Complexity of the mathematical model increases dramatically from black-oil to compositional models. For compositional models, direct solution of this large system (full set of equations and variables) is almost impossible, due to the high computational and memory requirements.
- There is no need to solve all of the coupled equations together. For each type of simulation, there is a minimum number of independent variables and equations (primary variables and equations) that determine the thermodynamic state (intensive and extensive) of the system. For isothermal compositional system, this thermodynamic state is determined by  $n_c$  variables, or the number of components in the system. By solving for this minimum number of variables first (fully implicit

models), we can save a large amount of computational time. Even so, this approach is still too expensive for large problems.

- In fully implicit (FIM) models all of the primary variables are treated implicitly. To further reduce the cost of simulation, we can treat some primary variables explicitly, and reduce the number of implicit unknowns per gridblock. Normally, such models are much faster than FIM models, especially for problems with large number of components. However, explicit treatment of some primary variables will limit the maximum timestep size, if the maximum allowable timestep size is too small, such as less than one day, such models could be even slower than FIM models. Explicit treatment of some primary variables may also cause some balance errors (mass or volume). There are numerical techniques to reduce or switch the balance errors (Coats et al., 1998), but we cannot totally remove them.



**Figure 3.1** Reduction process from full set to implicit primary set

In general, we should start with the full set of equations and variables, first reduce them to the primary set (minimum number), then reduce the implicit level of the primary set to further speed up the simulation if necessary, as shown in Figure 3.1. This two step reduction process is included in the General Formulation Approach.

The full set of equations and variables are the starting point for this approach. In Chapter 1, we already pointed out that, the full set of equations are always the mass balance equations and the phase equilibrium relations, but the selection of the full set of variables is not unique. GPRS uses the natural variables as its basic full set of variables to

facilitate the calculation of derivatives. If other sets of variables are needed, such as Type B variables, we first build explicit relations between the new variables and the natural variables, then use the chain rule to switch from the natural variables to the new variables. This is actually the first step of the General Formulation Approach.

In summary, this General Formulation Approach includes three steps. First, we define a base model which uses the natural variables, and build other models from this base model by variable switching. Then, we reduce the full set to the primary set, and build the FIM model. Finally, if necessary we further reduce the implicit level of the primary set. Through these three steps, we control the variables used, and the implicitness of each variable. These three steps are performed gridblock by gridblock, so each gridblock can have different variables and different implicit levels. This provides a natural way to form adaptive implicit schemes, such as the AIM model (Forsyth and Sammon, 1986).

In this chapter, we will first take a look at the general nonlinear equation set that forms the basis of this approach. Then we will explain the operations involved in each of the above three steps. Next we will do some stability analysis for the explicit treatment of some primary variables. Finally we will discuss the consequences of using different kinds of equations and variables.

### 3.1 The General Nonlinear Equation Set

In order to make this approach suitable for any kinds of simulations, we base it on the most general nonlinear equation set, which can be expressed as

$$\vec{F}(\vec{X}) = 0 \quad (3.1)$$

where  $\vec{F}$  is the full set of equations, and  $\vec{X}$  is the full set of variables. This full set of equations and variables can be split into a primary part and a secondary part:

$$\begin{cases} \vec{F}_p(\vec{X}_p, \vec{X}_s) = 0 \\ \vec{F}_s(\vec{X}_p, \vec{X}_s) = 0 \end{cases} \quad (3.2)$$

where, subscript  $p$  stands for primary, and subscript  $s$  stands for secondary.

$\vec{F}_p = 0$  is the primary equation set, and we always select the mass balance equations for it. Its Jacobian matrix structure (derivatives of the mass balance equations with respect to the full set of variables) looks like the one shown in Figure 2.4 and Figure 2.6, here we have a reservoir Jacobian and a well Jacobian. The reservoir Jacobian has both diagonal terms and off-diagonal terms. The well Jacobian includes four arrays,  $RRJ$ ,  $RWJ$ ,  $WRJ$  and  $WWJ$ , and normally  $RRJ$  is added to the diagonal part of the reservoir Jacobian.  $\vec{F}_s = 0$  is the secondary equation set, we select the phase equilibrium relations (a series of nonlinear constraint equations) for it. Since it only depends on individual gridblocks, its Jacobian matrix will be a block-diagonal matrix. Combining the Jacobian matrix of  $\vec{F}_p$  and  $\vec{F}_s$ , we have the Jacobian matrix for the full set of equations and variables, which is shown in Figure 3.2.

diagonal                      Jacobian						RHS
A1	B1	.....	A12	B12	A <sub>RW</sub>	M1
C1	D1		0	0	0	N1
.....			.....			.....
A21	B21	.....	A2	B2		M2
0	0		C2	D2		N2
A <sub>WR</sub> B <sub>WR</sub>					A <sub>WW</sub>	M <sub>WW</sub>

**Figure 3.2**      Structure of the full set of Jacobian matrix and RHS

Here we define the following matrices:

$$A = \frac{\partial \vec{F}_p}{\partial \vec{X}_p}, \quad B = \frac{\partial \vec{F}_p}{\partial \vec{X}_s}, \quad C = \frac{\partial \vec{F}_s}{\partial \vec{X}_p}, \quad D = \frac{\partial \vec{F}_s}{\partial \vec{X}_s}, \quad (3.3)$$

and vectors:

$$\vec{M} = -\vec{F}_p, \quad \vec{N} = -\vec{F}_s. \quad (3.4)$$

where A, B and M are generated by the primary equations, and C, D and N are generated by the secondary equations. Each matrix or vector has its own storage array, and the storage for matrices A and B is further separated into three parts (*Diag*, *OffD\_A* and *OffD\_B*) as shown in Section 2.2. In this General Formulation Approach, we perform operations on this full set Jacobian matrix, eliminate terms to zero level by level, and finally extract a minimum set for the linear solver. All of the operations are done locally, no additional storage is required.

This General Formulation Approach is applied to isothermal compositional models in GPRS. We assume that the hydrocarbon components and water are totally separated, then  $\vec{F}_p = 0$  will be the mass balance equations for each hydrocarbon component and for water:

$$F_{c,i} = \frac{\partial}{\partial t} [V\phi(S_o\rho_o x_c + S_g\rho_g y_c)]_i - \sum_l [T(\lambda_o\rho_o x_c \Delta\Phi_o + \lambda_g\rho_g y_c \Delta\Phi_g)]_{l,i} + \sum_w (\rho_o x_c q_o^w + \rho_g y_c q_g^w)_i = 0 \quad (3.5)$$

$$c = 1, \dots, n_h$$

$$F_{w,i} = \frac{\partial}{\partial t} (V\phi S_w \rho_w)_i - \sum_l (T\lambda_w \rho_w \Delta\Phi_w)_{l,i} + \sum_w (\rho_w q_w^w)_i = 0 \quad (3.6)$$

Optionally, we can replace one of the hydrocarbon component mass balance equations by a total hydrocarbon mass balance equation:

$$F_{h,i} = \frac{\partial}{\partial t} [V\phi(S_o\rho_o + S_g\rho_g)]_i - \sum_l [T(\lambda_o\rho_o \Delta\Phi_o + \lambda_g\rho_g \Delta\Phi_g)]_{l,i} + \sum_w (\rho_o q_o^w + \rho_g q_g^w)_i = 0 \quad (3.7)$$

$\vec{F}_s = 0$  will be the phase equilibrium relations:

$$F_e = f_{c,o} - f_{c,g} = 0, \quad c = 1, \dots, n_h \quad (3.8)$$

This full set of equations is used for all of the isothermal compositional models.

The basic full set of variables,  $\vec{X}_{base}$ , are chosen to be

$$\begin{aligned} p &= p_g, \\ S_g, S_o, \\ y_c, x_c, c &= 1, 2, \dots, n_h - 1 \end{aligned}$$

which are actually the natural variables, or Type A variables. For other variables, a switch from the natural variables to the new variables is required. Here we have a total of  $2n_h + 1$  variables in  $2n_h + 1$  nonlinear equations. Capillary pressure constraints are used to remove water and oil pressures, the saturation constraint is used to remove water saturation, and component mole fraction constraints are used to remove the mole fractions of the last hydrocarbon component in the oil and gas phases.

Black-oil models are treated as special cases of the general compositional model, where secondary equations and variables are no longer needed and component mole fractions are either constants or only functions of pressure, refer to appendix B for detail. For thermal models,  $\vec{F}_p$  will include the energy balance equation, and  $\vec{X}_p$  will include temperature or internal energy.

So far, we have shown the base model (the full set of equations in terms of the base variables), which is also the starting point for the General Formulation Approach. In the following section, we will explain the three steps of this approach. For each step, the operations for the reservoir part and for the well part are discussed separately. Note that  $RRJ$  of the well Jacobian is already added to the diagonal part of the reservoir Jacobian, so it will not appear in the following discussion.

### 3.2 General Formulation Approach Steps

The first step of the General Formulation Approach is to switch from the base variables to the new variables, and this step is only necessary when variables other than the natural variables are used.



### 3.2.1 Switching From the Natural Variables to Other Variables

In GPRS, the base model uses the natural variables, and it is straightforward to calculate derivatives with respect to these variables, and they are  $\frac{\partial \vec{F}}{\partial \vec{X}_{base}}$ . If another model using a

new set of variables  $\vec{X}$  is desired, then we must compute derivatives with respect to the new variables  $\frac{\partial \vec{F}}{\partial \vec{X}}$ . We can do this by the chain rule:

$$\frac{\partial \vec{F}}{\partial \vec{X}} = \frac{\partial \vec{F}}{\partial \vec{X}_{base}} \frac{\partial \vec{X}_{base}}{\partial \vec{X}} \quad (3.9)$$

where  $\frac{\partial \vec{X}_{base}}{\partial \vec{X}}$  can be calculated from the relation between  $\vec{X}$  and  $\vec{X}_{base}$ . The whole

switching process is summarized below:

1. Build explicit relation  $\vec{X} = \vec{X}(\vec{X}_{base})$  between  $\vec{X}$  and  $\vec{X}_{base}$ , and calculate the transformation matrix  $TRAN = \frac{\partial \vec{X}_{base}}{\partial \vec{X}} = \left(\frac{\partial \vec{X}}{\partial \vec{X}_{base}}\right)^{-1}$  for each gridblock.
2. Calculate the new derivatives by  $\frac{\partial \vec{F}}{\partial \vec{X}} = \frac{\partial \vec{F}}{\partial \vec{X}_{base}} \cdot TRAN$ . This is done for all terms in the reservoir Jacobian and for  $WRJ$  in the well Jacobian, and the results are stored back in  $\frac{\partial \vec{F}}{\partial \vec{X}_{base}}$ .
3. Reduce the full set system to the final implicit primary set system (step 2 and 3 of the General Formulation Approach)
4. Solve the new linear system and update explicit variables, get  $\delta \vec{X}$  and update the new variables by  $\vec{X}^{v+1} = \vec{X}^v + \delta \vec{X}$
5. Update the base variables  $\vec{X}_{base}$  by solving the explicit relation  $\vec{X} = \vec{X}(\vec{X}_{base})$ , which may be nonlinear in some cases.

Different models will have different transformation matrices due to different relations between the new variables  $\vec{X}$  and the base variables  $\vec{X}_{base}$ , but the above five steps will be exactly the same. Next we will review examples of new variables, and express each of

them in terms of the base variables. Based on a review of existing compositional models, the possible choices for  $\vec{X}$  (besides the base model variables) are identified as:

$S_w$ ,	water saturation
$x_{n_h}, y_{n_h}$ ,	mole fraction of the last hydrocarbon component in the oil and gas phases
$z_c$ ,	overall mole fraction of each hydrocarbon component
$r_c$ ,	overall density of each hydrocarbon component
$r_w$ ,	overall density of water component
$F$	overall density of hydrocarbon components
$l$	hydrocarbon liquid (oil) mole fraction

and their relations to the base variables are given below:

$$S_w = 1 - S_g - S_o \quad (3.10)$$

$$x_{n_h} = 1 - \sum_{c=1}^{n_h-1} x_c \quad (3.11)$$

$$y_{n_h} = 1 - \sum_{c=1}^{n_h-1} y_c \quad (3.12)$$

$$z_c = \frac{\rho_o S_o x_c + \rho_g S_g y_c}{\rho_o S_o + \rho_g S_g} \quad (3.13)$$

$$r_c = \rho_o S_o x_c + \rho_g S_g y_c \quad (3.14)$$

$$r_w = \rho_w (1 - S_g - S_o) \quad (3.15)$$

$$F = \rho_o S_o + \rho_g S_g \quad (3.16)$$

$$l = \frac{\rho_o S_o}{\rho_o S_o + \rho_g S_g} \quad (3.17)$$

where,  $\rho_w = \rho_w(p)$ ,  $\rho_o = \rho_o(p, x_1, \dots, x_{n_h-1})$  and  $\rho_g = \rho_g(p, y_1, \dots, y_{n_h-1})$

Note that this process is equivalent to the use of chain rule to get the derivatives in Type B models. So it should not introduce any extra work. For Type A models, all of the transformation matrices are identity matrices, and this step is not necessary.

Besides the base model, a Type B model is implemented in GPRS, where the full set of variables is

$$\begin{aligned}
& p, \\
& r_c, c = 1, 2, \dots, n_h - 1 \\
& r_w, \\
& S_g, \text{ and} \\
& y_c, c = 1, 2, \dots, n_h - 1.
\end{aligned}$$

Within this set,  $p$ ,  $r_c, c = 1, 2, \dots, n_h - 1$  and  $r_w$  are the primary variables. The following relations are used to calculate the transformation matrix:

$$\begin{cases} p = p \\ r_c = \rho_o S_o x_c + \rho_g S_g y_c \\ r_w = \rho_w (1 - S_g - S_o) \\ S_g = S_g \\ y_c = y_c \end{cases} \quad (3.18)$$

The derivative of the overall component densities with respect to the base variables are calculated as

$$\begin{cases} \frac{\partial r_c}{\partial p} = \frac{\partial \rho_o}{\partial p} S_o x_c + \frac{\partial \rho_g}{\partial p} S_g y_c \\ \frac{\partial r_c}{\partial S_g} = \rho_g y_c, \quad \frac{\partial r_c}{\partial S_o} = \rho_o x_c, \\ \frac{\partial r_c}{\partial y_{c'}} = \rho_g S_g \delta_{c,c'} + \frac{\partial \rho_g}{\partial y_{c'}} S_g y_c, \quad \frac{\partial r_c}{\partial x_{c'}} = \rho_o S_o \delta_{c,c'} + \frac{\partial \rho_o}{\partial x_{c'}} S_o x_c \end{cases} \quad (3.19)$$

$$\text{and } \begin{cases} \frac{\partial r_w}{\partial p} = \frac{\partial \rho_w}{\partial p} (1 - S_g - S_o) \\ \frac{\partial r_w}{\partial S_g} = \frac{\partial r_w}{\partial S_o} = -\rho_w \\ \frac{\partial r_w}{\partial y_c} = \frac{\partial r_w}{\partial x_c} = 0 \end{cases} \quad (3.20)$$

The final structure of the inverse of the transformation matrix ( $TRAN^{-1}$ ) is shown in Figure 3.3. The transformation matrix is calculated as the inverse of  $TRAN^{-1}$  in GPRS,

regardless of its internal structure. But, for matrices with special structures, like the one in Figure 3.3, there are more efficient ways to calculate its inverse.

	$p$	$S_g$	$S_o$	$y_c$	$x_c$
$p$	1	0	0	0	0
$r_c$	$\frac{\partial \rho_o}{\partial p} S_o x_c + \frac{\partial \rho_g}{\partial p} S_g y_c$	$\rho_g y_c$	$\rho_g x_c$	$\rho_g S_g \delta_{c,c} + \frac{\partial \rho_g}{\partial y_c} S_g y_c$	$\rho_o S_o \delta_{c,c} + \frac{\partial \rho_o}{\partial x_c} S_o x_c$
$r_w$	$\frac{\partial \rho_w}{\partial p} (1 - S_g - S_o)$	$-\rho_w$	$-\rho_w$	0	0
$S_g$	0	1	0	0	0
$y_c$	0	0	0	I	0

**Figure 3.3** Structure of the inverse of the transformation matrix for a Type B model

Now that we have the full set of equations and variables, and we also have different choices of variables, the next step is to reduce the number of equations and variables to a set that is solved together. First we reduce it to the minimum number ( $n_c$ ) of equations and variables required to establish the thermodynamic state (the FIM model), then we can further reduce the implicit level.

### 3.2.2 Reduce Full Set $\vec{F}(\vec{X}) = 0$ to Primary Set $\vec{F}_p(\vec{X}_p) = 0$

In order to write an iterative process to solve  $\vec{F}(\vec{X}) = 0$ , we could use the Newton-Raphson method:

$$\left(\frac{\partial \vec{F}}{\partial \vec{X}}\right)^v \delta \vec{X} = -\vec{F}^v(\vec{X}), \quad \text{and} \quad \delta \vec{X} = \vec{X}^{v+1} - \vec{X}^v \quad (3.21)$$

or we can express  $\vec{F}(\vec{X})$  in Taylor series:

$$0 = \vec{F}^{n+1}(\vec{X}) \approx \vec{F}^{v+1}(\vec{X}) = \vec{F}^v(\vec{X}) + \left(\frac{\partial \vec{F}}{\partial \vec{X}}\right)^v \delta \vec{X} \quad (3.22)$$

Eqn. 3.21 and Eqn. 3.22 are equivalent. In the following analytical derivation, Taylor series are used to derive the primary set. In addition, we will also show a matrix manipulation method, which can achieve the same final goal as the analytical derivation method. The matrix manipulation method is used in GPRS, and we will discuss it in more detail later.

### Analytical Derivation

In the last section, we have defined the following matrices:

$$A = \frac{\partial \vec{F}_p}{\partial \vec{X}_p}, B = \frac{\partial \vec{F}_p}{\partial \vec{X}_s}, C = \frac{\partial \vec{F}_s}{\partial \vec{X}_p}, D = \frac{\partial \vec{F}_s}{\partial \vec{X}_s}$$

and vectors:

$$\vec{M} = -\vec{F}_p, \vec{N} = -\vec{F}_s$$

where matrices A, B, C and D together form the full set Jacobian matrix for Eqn.3.2, and  $\vec{M}$ ,  $\vec{N}$  are the corresponding right hand sides (RHS) for the primary part and for the secondary part, as shown in the first frame of Figure 3.4.

The first step is to relate the secondary variables to the primary variables. We start with the secondary equation set  $\vec{F}_s(\vec{X}_p, \vec{X}_s) = 0$ . Use of Taylor series on this equation set yields

$$\begin{aligned} 0 &\approx \vec{F}_s^{v+1}(\vec{X}_p, \vec{X}_s) = \vec{F}_s^v(\vec{X}_p, \vec{X}_s) + \left(\frac{\partial \vec{F}_s}{\partial \vec{X}_p}\right)^v \delta \vec{X}_p + \left(\frac{\partial \vec{F}_s}{\partial \vec{X}_s}\right)^v \delta \vec{X}_s \\ &= -\vec{N}^v + C^v \delta \vec{X}_p + D^v \delta \vec{X}_s \end{aligned} \quad (3.23)$$

Rearranging Eqn. 3.23, we get

$$\delta \vec{X}_s = (D^{-1} \vec{N})^v - (D^{-1} C)^v \delta \vec{X}_p \quad (3.24)$$

The second step is to write the primary equations in the primary variables. Use of Taylor series on the primary equation set  $\vec{F}_p(\vec{X}_p, \vec{X}_s) = 0$  yields

$$\begin{aligned}
0 &\approx \vec{F}_p^{\nu+1}(\vec{X}_p, \vec{X}_s) = \vec{F}_p^\nu(\vec{X}_p, \vec{X}_s) + \left(\frac{\partial \vec{F}_p}{\partial \vec{X}_p}\right)^\nu \delta \vec{X}_p + \left(\frac{\partial \vec{F}_p}{\partial \vec{X}_s}\right)^\nu \delta \vec{X}_s \\
&= -\vec{M}^\nu + A^\nu \delta \vec{X}_p + B^\nu \delta \vec{X}_s
\end{aligned} \tag{3.25}$$

Substituting Eqn.3.24 into the above equation, we get

$$0 = -\vec{M}^\nu + A^\nu \delta \vec{X}_p + B^\nu [(D^{-1}\vec{N})^\nu - (D^{-1}C)^\nu \delta \vec{X}_p] \tag{3.26}$$

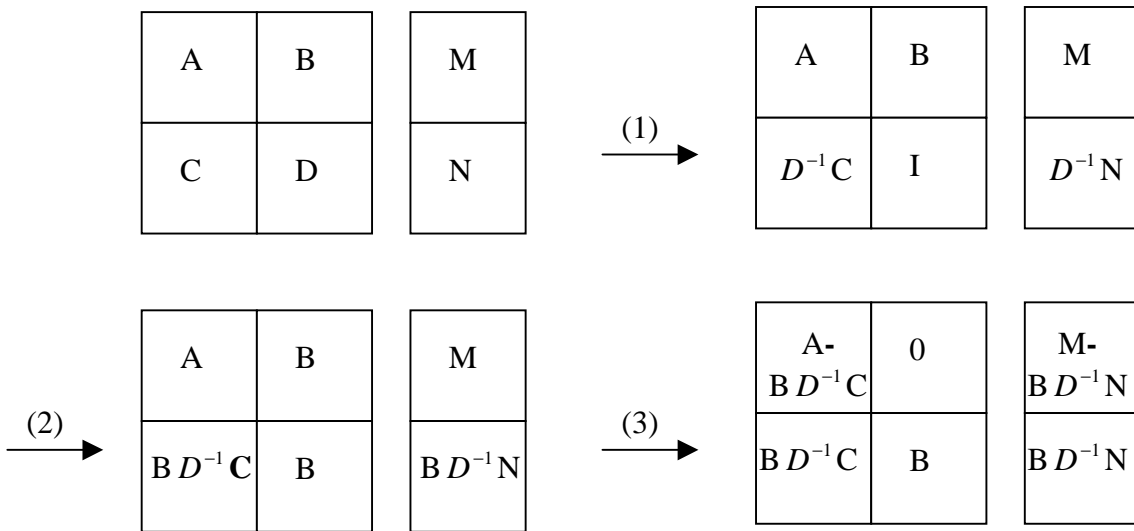
Rearranging terms, we have

$$(A - BD^{-1}C)^\nu \delta \vec{X}_p = (\vec{M} - BD^{-1}\vec{N})^\nu \tag{3.27}$$

This is the final primary system, which only includes the primary variables as unknowns. After we solve for the primary variables, the secondary variables can be updated explicitly from Eqn. 3.24 gridblock by gridblock. We can achieve the same goal by performing matrix manipulation directly on the full set Jacobian matrix and RHS.

### Matrix Manipulation (Gaussian Elimination)

Here we perform Gauss elimination on the blocks (A, B, C, D and M, N) of the full set Jacobian matrix and RHS. The goal is to eliminate B to zero, then the primary equations will depend only on the primary variables. This is illustrated in Figure 3.4.



**Figure 3.4** Reduce full set to primary set by Gaussian elimination

First, we multiply the secondary equation set by  $D^{-1}$ , then we multiply it again by B, and finally we subtract the secondary equation set from the primary equation set. The final primary equation set can be extracted and written as

$$(A - BD^{-1}C)\delta\vec{X}_p = (\vec{M} - BD^{-1}\vec{N}) \quad (3.28)$$

which is the same equation as Eqn. 3.27. We can also extract a secondary equation set from the second frame of Figure 3.4, after rearrange it, we have

$$\delta\vec{X}_s = (D^{-1}\vec{N})^v - (D^{-1}C)^v \delta\vec{X}_p \quad (3.29)$$

which is the same equation as Eqn. 3.24.

The three operations in Figure 3.4 are performed for each gridblock and for each part of the Jacobian matrix. From Figure 3.2, we know that matrices A and B exist on the diagonal part, the off-diagonal part and the well part of the full set Jacobian. Matrices C and D only exist on the diagonal part. We need to eliminate all parts of B. Note that matrices A, B, C and D are always in the same column (they are the derivatives with respect to the variables in the same gridblock), A, B and M are in the same row (equations for the same gridblock), and C, D and N are in the same row. Different parts of B are eliminated as discussed below:

- **Diagonal part of B** (*Diag* array) Loop through the cell list, select the diagonal A, B, C and D for the same block and their corresponding RHS, M and N, and perform the three step elimination process. In addition,  $D^{-1}$  is calculated and stored back in D,  $D^{-1}C$  and  $D^{-1}N$  are calculated and stored back in C and N to avoid repeating calculations and using extra memory space.
- **Off-diagonal part of B** (*OffD\_A* and *OffD\_B* arrays) Loop through the connection list, for each A and B in the off-diagonal part, select the C and D which are in the same column as A and B. M is corresponding to A and B, and N is corresponding to C and D. Only the last step of the elimination is needed here, since  $D^{-1}C$  and  $D^{-1}N$  have already been calculated. This process needs to be done for both the *OffD\_A* array and for the *OffD\_B* array.

- **Well part of B** If a well has the  $WRJ$  array, for each  $A_{WR}$  and  $B_{WR}$  in the  $WRJ$ , select the  $C$  and  $D$  which are in the same column as  $A_{WR}$  and  $B_{WR}$ .  $M_{WW}$  is corresponding to  $A_{WR}$  and  $B_{WR}$ , and  $N$  is corresponding to  $C$  and  $D$ . Only the last step of the elimination is needed here. If a well penetrates multiple gridblocks, there will be several  $A_{WR}$  and  $B_{WR}$ , and the elimination process will be performed for each of them.

To make the process clearer, for the system shown in Figure 3.2, the following operations are performed to reduce  $B$  to 0:

- Diagonal part operations:

$$\begin{array}{ll}
D1 \leftarrow D1^{-1} & D2 \leftarrow D2^{-1} \\
C1 \leftarrow D1^{-1} \cdot C1 & C2 \leftarrow D2^{-1} \cdot C2 \\
N1 \leftarrow D1^{-1} \cdot N1 & N2 \leftarrow D2^{-1} \cdot N2 \\
A1 \leftarrow A1 - B1 \cdot (D1^{-1} \cdot C1) & M1 \leftarrow M1 - B1 \cdot (D1^{-1} \cdot N1) \\
A2 \leftarrow A2 - B2 \cdot (D2^{-1} \cdot C2) & M2 \leftarrow M2 - B2 \cdot (D2^{-1} \cdot N2)
\end{array}$$

- Off-diagonal part operations:

$$\begin{array}{ll}
A12 \leftarrow A12 - B12 \cdot (D2^{-1} \cdot C2) & M1 \leftarrow M1 - B12 \cdot (D2^{-1} \cdot N2) \\
A21 \leftarrow A21 - B21 \cdot (D1^{-1} \cdot C1) & M2 \leftarrow M2 - B21 \cdot (D1^{-1} \cdot N1)
\end{array}$$

- Well part operations:

$$A_{WR} \leftarrow A_{WR} - B_{WR} \cdot (D1^{-1} \cdot C1) \quad M_{WW} \leftarrow M_{WW} - B_{WR} \cdot (D1^{-1} \cdot N1)$$

where  $A1 \leftarrow A1 - B1 \cdot (D1^{-1} \cdot C1)$  means that  $A1 - B1 \cdot (D1^{-1} \cdot C1)$  is calculated and stored back in  $A1$ . The updated  $A$  and  $M$  can later be extracted for fully implicit solve or for further reductions. After the above operations, the  $A$  and  $M$  of the full set Jacobian and RHS in Figure 3.2 are reduced to the primary set Jacobian and RHS (fully implicit). The structure of the resulting Jacobian is shown in Figure 3.5.



diagonal	Jacobian			RHS	
	A1	...	A12	A <sub>RW</sub>	M1
	...		...		...
	A21		A2		M2
		...			
	A <sub>WR</sub>			A <sub>WW</sub>	M <sub>WW</sub>

**Figure 3.5** Structure of the primary set Jacobian Matrix and RHS

After solving for the primary variables together, the secondary variables of each gridblock are updated explicitly by

$$\begin{aligned}\delta\vec{X}_{s,1} &= (D1^{-1} \cdot N1) - (D1^{-1} \cdot C1) \cdot \delta\vec{X}_{p,1} \\ \delta\vec{X}_{s,2} &= (D2^{-1} \cdot N2) - (D2^{-1} \cdot C2) \cdot \delta\vec{X}_{p,1}\end{aligned}$$

We have mentioned in Chapter 2 that, for single block wells with constant rate control, we can decouple the reservoir Jacobian from the well Jacobian by eliminating *RWJ*, then we can solve for the reservoir variables first and explicitly update the well variable later. If the well in Figure 3.5 only has single block penetration and *RWJ* exists, the following additional operations are performed in GPRS:

$$A1 \leftarrow A1 - A_{RW} \cdot (A_{WW}^{-1} \cdot A_{WR}) \quad M1 \leftarrow M1 - A_{RW} \cdot (A_{WW}^{-1} \cdot M_{WW})$$

After the primary reservoir variables have been solved, the well variable is updated by

$$\delta\vec{X}_w = (A_{WW}^{-1} \cdot M_{WW}) - (A_{WW}^{-1} \cdot A_{WR}) \cdot \delta\vec{X}_{p,1}$$

Here, Newton iterations are performed on both the primary equations and the secondary equations. If the secondary equations are only used as constraint equations, as mentioned in Chapter 2, then all of the N's in the above equations will be zero.

Some variations in the general operations are necessary as discussed below.

- If the transmissibility is treated explicitly, then there is no need to eliminate off-diagonal parts of B, because flux terms only have derivatives to pressure, which is always a primary variable, and all off-diagonal parts of B are zero.
- Matrix D is actually the Jacobian matrix for flash calculations. When we update the secondary variables, it is equivalent to doing one step of flash calculations for each Newton iteration. For exact flash calculations at each Newton step, we can use it as the initial guess for flash.
- For the black-oil model, there are no secondary variables, and this step is not needed. For compositional models, if a gridblock has only one hydrocarbon phase, then the primary equations will only depend on the primary variables as shown in Section 2.6, and this step is also not necessary.

So far, we have derived the fully implicit Jacobian matrix and RHS. For fully implicit simulations we can just stop here. But in a lot of situations, we desire to reduce the implicit level to speed up the calculations. The next subsection will show how this is done.

### 3.2.3 Reduce the Implicit Level of the Fully Implicit System $\vec{F}_p(\vec{X}_p) = 0$

To reduce the implicit level of the fully implicit system, we first split the primary variables into an implicit part and an explicit part, and set the derivatives of flux terms with respect to the explicit variables to zero, then we decouple the implicit variables from the explicit variables. The second step is basically a decoupling step. This process needs to be done for each gridblock that is not fully implicit. After the implicit variables have been solved for together, the explicit variables can be updated explicitly gridblock by gridblock. In this part, we will discuss the various decoupling methods available, their differences, and which one is used in GPRS.

For each gridblock, the fully implicit Jacobian matrix is shown in Figure 3.6, which is one row of the matrix equation in Figure 3.5.

	diagonal $V_1 V_2 V_3 P$		Off-diagonal $V_1 V_2 V_3 P$		$RWJ$ $p^w$	RHS
.....	<del> X X X X  X X X X  X X X X  X X X X </del>	.....	X X X X X X X X X X X X X X X X		X X X X	X X X X

**Figure 3.6** Fully implicit Jacobian matrix of one gridblock

Here the left part is the Jacobian matrix  $A$ , which includes a diagonal part, several off-diagonal parts and several well parts ( $RWJ$ ). The right part is the right hand side  $M$ .  $P$  is the pressure or the implicit primary variables,  $V_1$ ,  $V_2$  and  $V_3$  are the explicit primary variables.

We can reduce the implicit level of the fully implicit system in Figure 3.6 in two steps:

1. First, set all flux term derivatives with respect to the explicit variables to zero, or simply don't calculate them. Since the off-diagonal parts of  $A$  are the flux term derivatives, this treatment will leave parts (derivatives with respect to the explicit variables) of the off-diagonal  $A$  to zero. Diagonal  $A$  will also have some change, but it won't be zero. This is illustrated in Figure 3.7.

	diagonal $V_1 V_2 V_3 P$		Off-diagonal $V_1 V_2 V_3 P$		$RWJ$ $p^w$	RHS
.....	<del> X X X X  X X X X  X X X X  X X X X </del>	.....	0 0 0 X 0 0 0 X 0 0 0 X 0 0 0 X		X X X X	X X X X

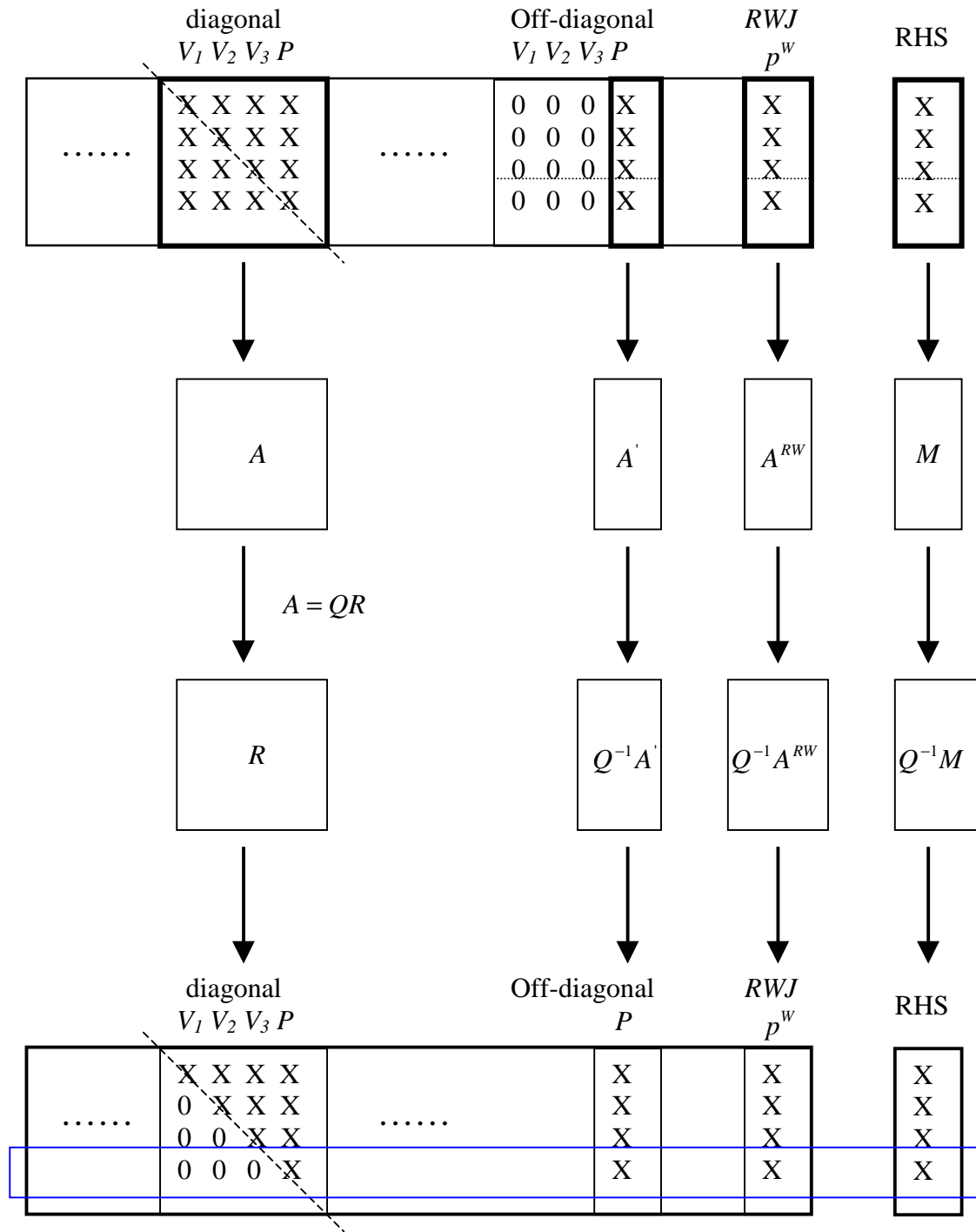
**Figure 3.7** Jacobian matrix of one gridblock after explicit treatment of flux terms

2. Next, decouple the implicit variables from the explicit variables. There are different kinds of decoupling methods available, two popular choices are Householder reflection (QR decomposition) and local inversion (Lacroix et al., 2000). Coats

(1999) has pointed out that the IMPES pressure equation is unique, independent of the manner of derivation, choice and ordering of variables and equations. Based on it, we can conclude that the choice of decoupling methods does not have too much influence on the final implicit system.

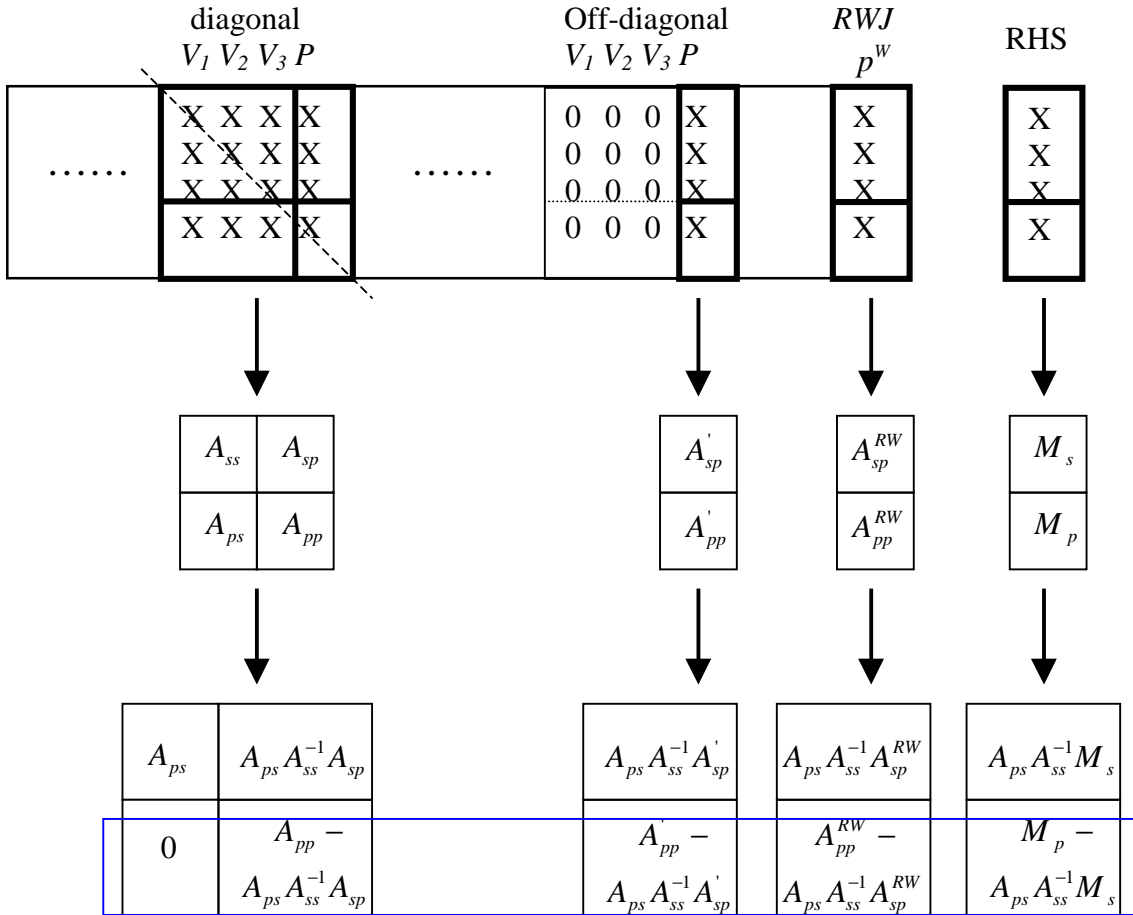
Here we only review the two most popular decoupling methods, Householder reflection and local inversion:

- **Householder reflection** The process of using Householder reflection (Lacroix et al., 1999) is summarized in Figure 3.8. We start from the system in Figure 3.7, and move each part (dark rectangles in Figure 3.8) into a matrix or column vector. Then we perform a QR decomposition ( $A=QR$ , where  $Q$  is an orthogonal matrix,  $Q^{-1}=Q^T$ , and  $R$  is an upper triangular matrix) on the diagonal matrix, and use the transpose of  $Q$  to multiply all of the matrices and vectors. After that, the diagonal matrix becomes  $R$ , which only has values in the upper triangular part, the last equation will be only in terms of the implicit variables, and it is extracted and passed to the linear solver. After it is solved, the explicit variables can be easily updated by making use of the upper triangular structure of the diagonal matrix. The same results can be achieved by performing Gaussian elimination on the system in Figure 3.7.



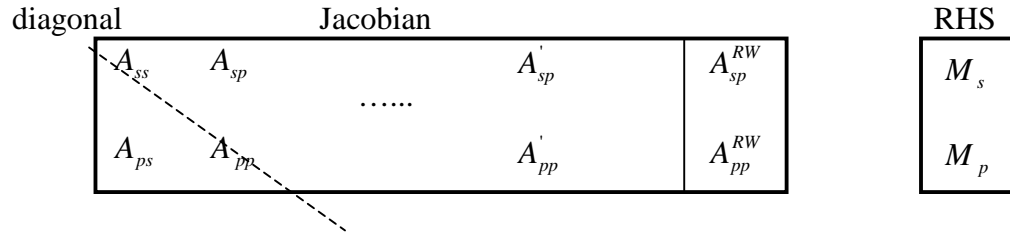
**Figure 3.8** Householder reflection decoupling

- Local inversion** The process of using local inversion (Lacroix et al., 2000) is summarized in Figure 3.9. The notation used here follows the normal notation used for black-oil IMPES models, subscript  $p$  stands for pressure or implicit variables, and subscript  $s$  stands for saturation or explicit variables. Here we also start from the system in Figure 3.7, but we do an additional splitting step to separate the implicit part from the explicit part, and move each part (dark rectangles in Figure 3.9) into a small matrix or column vector. Then we use the procedure in Figure 3.4 to reduce  $A_{ps}$  to zero. Finally the second equation can be extracted and solved for implicit variables. After that, the explicit variables are updated from the first equation. Note that we operate on the matrices and vectors in the same row (belong to the same gridblock). Like the procedure used in Figure 3.4, it is done separately for the diagonal part, the off-diagonal part, and the well part ( $RWJ$ ).



**Figure 3.9** Local inversion decoupling

For the IMPES model, numerical experiments (Vassilevski, 2001) show that for most cases, the pressure system from Householder reflection is more symmetric than the pressure system from local inversion, which is better for the linear pressure solver. But in GPRS, we use the local inversion method anyway, because this procedure is similar to the procedure used in Figure 3.4, and no additional work is needed to implement it. The method used for decoupling does not make much difference, since the final implicit systems are always equivalent.



**Figure 3.10** Illustration for local inversion

To make this step clearer, for the system shown in Figure 3.10, the following operations are performed to reduce  $A_{ps}$  to 0 in GPRS:

- Diagonal part operations:

$$\begin{aligned}
 A_{ps} &\leftarrow A_{ps} \cdot A_{ss}^{-1} \\
 A_{pp} &\leftarrow A_{pp} - (A_{ps} \cdot A_{ss}^{-1}) \cdot A_{sp} \\
 M_p &\leftarrow M_p - (A_{ps} \cdot A_{ss}^{-1}) \cdot M_s
 \end{aligned}$$

- Off-diagonal parts operations:

$$A_{pp}' \leftarrow A_{pp} - (A_{ps} \cdot A_{ss}^{-1}) \cdot A_{sp}'$$

- Well parts (RWJ) operations:

$$A_{pp}^{RW} \leftarrow A_{pp}' - (A_{ps} \cdot A_{ss}^{-1}) \cdot A_{sp}^{RW}$$

$A_{pp}'$ ,  $A_{pp}'$ ,  $A_{pp}^{RW}$  and  $M_p$  form the final implicit system, they are extracted and passed to the linear solver. After getting the implicit variables, the explicit variables are updated explicitly by

$$\delta \vec{X}_{\text{exp}} = A_{ss}^{-1} \cdot (M_s - A_{sp} \cdot \delta \vec{X}_{\text{imp}} - A_{sp}' \cdot \delta \vec{X}_{\text{imp}}' - A_{sp}^{RW} \cdot \delta \vec{X}^W)$$

Coats et al. (1995) proposed using IMPES reduction factor to generate the pressure system. Basically it is the application of the procedure in Figure 3.9 for the case where only the pressure is treated implicitly, such as in IMPES. In that approach, each equation (one row) in Figure 3.7 is multiplied by a scalar, and the resulting  $n_c$  equations are added, and the IMPES reduction factor is a vector consisting of these scalars. The values of these scalars are determined so that the addition removes all of the variables except pressure. The IMPES reduction factor is actually the  $-A_{ps}A_{ss}^{-1}$  matrix in the General Formulation Approach. For IMPES, it is a vector, since  $A_{ps}$  is a row vector and  $A_{ss}^{-1}$  is a matrix. Coats (1999) showed that IMPES reduction factor leads directly to the value of total compressibility in multiphase gridblocks, and this in turn provides an error check for black-oil PVT data, such as if the calculated total compressibility is negative, then there must be some inconsistency in the black-oil PVT data.

Before any decoupling, parts of the transmissibility are treated explicitly. For each explicit part, we can either fix it at the old time level or update it iteration by iteration. The first approach has less numerical diffusion. The second approach is actually an inexact Newton method, which upon convergence gives the same solution as the FIM method. From Eqn. 2.5, we can see that transmissibility has three non-constant parts (phase mobility, phase density and component mole fraction), and each part can be treated differently. For each gridblock, we have the following four options for the treatment of transmissibility in GPRS:

- **Option 1** All three parts are fixed at the old time level, and only derivatives with respect to the implicit variables are calculated. This option is used for the IMPES model.
- **Option 2** Component mole fractions are fixed at the old time level, phase mobility and phase density are updated iteration by iteration, and only derivatives with respect to the implicit variables are calculated. This option is used for the IMPSAT model, where both pressure and saturation are treated implicitly, and component mole fractions are treated explicitly.



- **Option 3** All three parts are updated iteration by iteration, and only derivatives with respect to the implicit variables are calculated. This option can be used for any models, but it performs poorly for IMPES, which is actually the IMPES model proposed for an adaptive implicit scheme by Thomas and Thurnau (1983). Russell (1989) showed that in order to make this method stable, the CFL number must be less than 0.05.
- **Option 4** All three parts are updated iteration by iteration, and all derivatives are calculated regardless of the implicit level. This option is used for the FIM model. For the IMPES and IMPSAT models, this option can also be used, the reductions in implicit level are performed for all non-zero terms, and finally some terms are ignored to form the implicit system. The corresponding models are normally called quasi IMPES and quasi IMPSAT models. Generally, quasi IMPES is not used as a formulation, and it is only used as a way to generate a pressure system, which can be used later for preconditioning purpose.

In summary, the proposed General Formulation Approach can be used to obtain any desired formulation, regardless of the type of model, type of variables and level of implicitness. All of the operations are performed on individual gridblocks, so the extension to the use of different implicit levels for different gridblocks (Adaptive Implicit Method, AIM) is straightforward. This approach also allows the use of different models for different reservoirs (reservoirs are only connected through well groups and boundary interfaces, and each reservoir is handled by a single processor in parallel computation).

This General Formulation Approach only defines the general steps and methods to reach the final goal, and it is suitable for any model. But in order to make the simulator more efficient, we need to consider the details of each specific model. For example:

- In many cases, matrices are banded or sparse, we should only perform operations on the non-zero terms.
- For some models, such as variable Type B models, which use  $z_c$  as the primary variables, we know the analytical form of accumulation term derivatives with respect

to them, and we can just bypass the variable switching part, and directly calculate them from the start.

- Water equation only has derivatives with respect to pressure and saturations, which are both primary variables for the base model. This means that water equation is in terms of the primary variables only from the start, and we need not do any reduction for it, as long as both pressure and saturations are implicit.
- The matrix  $C$  is generated from the phase equilibrium relations, and it does not have any derivative with respect to saturations, so some columns of  $C$  are always zero. We can take advantage of this, and save some calculations.

These are some general considerations that reduce calculations, and they are far from complete. Furthermore, each specific model has its own characteristics, and we should use them to further simplify calculations. However, all of the operations of this approach are of first order to the number of gridblocks, so the cost should be small compared to the cost of linear solver, especially for large models. With proper tuning, the speed of this General Formulation Approach should be comparable with that of commercial simulators.

### 3.3 Stability Analysis

After the secondary variables are eliminated from the primary equations, we get the FIM model, which is unconditionally stable. In practice, in order to control the number of Newton iterations for each timestep, a reasonable timestep size is required, normally it is between 30 days and 100 days. If we further reduce the implicit level of the primary set, the explicit treatment of primary variables will make the model conditionally stable, and the timestep size will have to satisfy some stability criteria. In real simulations, we desire to use the maximum allowable (stable) timestep size, and stability analysis is needed to correctly identify it. The natural variables are used in GPRS, within them, pressure is always treated implicitly, and saturations and component mole fractions can be treated either implicitly or explicitly. Next, we will derive the stability criteria due to the explicit treatment of saturations and component mole fractions. Assuming one-dimensional two-

phase (gas/oil) flow, no source and sink, the hydrocarbon component mass balance equations can be written as

$$\frac{\partial}{\partial t}(\phi \rho_o S_o x_c + \phi \rho_g S_g y_c) = \frac{\partial}{\partial x}(\rho_o u_o x_c + \rho_g u_g y_c) \quad (3.30)$$

$$c = 1, \dots, n_h$$

where  $u_o$  and  $u_g$  are the phase velocities for the oil and gas phases.

Summing Eqn. 3.30 over all components, we get the total hydrocarbon mass balance equation:

$$\frac{\partial}{\partial t}(\phi \rho_o S_o + \phi \rho_g S_g) = \frac{\partial}{\partial x}(\rho_o u_o + \rho_g u_g) \quad (3.31)$$

By assuming incompressible flow, no gravity and constant viscosity, the total flow velocity  $u_T$  becomes a constant, and Eqn. 3.31 can be simplified as

$$\phi \rho_o \frac{\partial S_o}{\partial t} + \phi \rho_g \frac{\partial S_g}{\partial t} = \frac{\partial}{\partial x}(\rho_o f_o u_T + \rho_g f_g u_T) = \rho_o u_T \frac{\partial f_o}{\partial x} + \rho_g u_T \frac{\partial f_g}{\partial x} \quad (3.32)$$

$$\phi(\rho_g - \rho_o) \frac{\partial S_g}{\partial t} = u_T(\rho_g - \rho_o) \frac{\partial f_g}{\partial x} = u_T(\rho_g - \rho_o) \frac{df_g}{dS_g} \frac{\partial S_g}{\partial x} \quad (3.33)$$

where  $f_o$  and  $f_g$  are the fractional flows of the oil and gas phases, and they are only functions of saturation. After rearranging Eqn. 3.33, we get

$$\frac{\partial S_g}{\partial t} = \left( \frac{u_T}{\phi} \frac{df_g}{dS_g} \right) \frac{\partial S_g}{\partial x} \quad (3.34)$$

Eqn. 3.34 is the Buckley-Leverett wave equation, its stability is determined by the CFL number, which can be written as

$$CFL_{S,x} = \frac{u_T \cdot \Delta t}{\Delta x \cdot \phi} \cdot \frac{df_g}{dS_g} \leq 1 \quad (3.35)$$

For three-dimensional flow, the total CFL number is the summation of CFL numbers in all three directions (Russell, 1989):

$$\begin{aligned}
CFL_S &= CFL_{S,x} + CFL_{S,y} + CFL_{S,z} \\
&= \frac{q_{T,x} \cdot \Delta t}{A_x \Delta x \cdot \phi} \cdot \frac{df_g}{dS_g} + \frac{q_{T,y} \cdot \Delta t}{A_y \Delta y \cdot \phi} \cdot \frac{df_g}{dS_g} + \frac{q_{T,z} \cdot \Delta t}{A_z \Delta z \cdot \phi} \cdot \frac{df_g}{dS_g} \\
&= \frac{q_T \cdot \Delta t}{V \cdot \phi} \cdot \frac{df_g}{dS_g} \leq 1
\end{aligned} \tag{3.36}$$

where  $q_T$  is the total volumetric flow rate,  $q_{T,x}$ ,  $q_{T,y}$  and  $q_{T,z}$  are the volumetric flow rates along x, y and z directions, and  $q_T = q_{T,x} + q_{T,y} + q_{T,z}$ .  $V$  is the volume of a gridblock, and  $V = A_x \Delta x = A_y \Delta y = A_z \Delta z$ . The physical meaning of Eqn. 3.36 is that a phase front can not move more than one gridblock in a stable timestep.

Coats (2001) derived the same stability criteria for the general multi-dimensional three-phase flow, including both gravity and capillary pressure. For two-phase (gas/oil) flow without capillary pressure, it can be reduced to

$$CFL_S = \frac{\Delta t}{V\phi} \cdot \frac{\frac{\lambda_o}{\lambda_g} \frac{d\lambda_g}{dS_g} q_g - \frac{\lambda_g}{\lambda_o} \frac{d\lambda_o}{dS_g} q_o}{\lambda_o + \lambda_g} \leq 1 \tag{3.37}$$

By ignoring gravity, we have

$$\begin{cases} q_g = f_g q_T = \frac{\lambda_g}{\lambda_g + \lambda_o} q_T \\ q_o = f_o q_T = \frac{\lambda_o}{\lambda_g + \lambda_o} q_T \end{cases} \tag{3.38}$$

Substituting Eqn. 3.38 into Eqn. 3.37 and rearranging, we get

$$\begin{aligned}
CFL_S &= \frac{q_T \Delta t}{V\phi} \cdot \frac{\lambda_o \frac{d\lambda_g}{dS_g} - \lambda_g \frac{d\lambda_o}{dS_g}}{(\lambda_o + \lambda_g)^2} \\
&= \frac{q_T \Delta t}{V\phi} \cdot \frac{(\lambda_o + \lambda_g) \frac{d\lambda_g}{dS_g} - \lambda_g \left( \frac{d\lambda_o}{dS_g} + \frac{d\lambda_g}{dS_g} \right)}{(\lambda_o + \lambda_g)^2} \\
&= \frac{q_T \Delta t}{V\phi} \cdot d \left( \frac{\lambda_g}{\lambda_o + \lambda_g} \right) / dS_g \\
&= \frac{q_T \Delta t}{V\phi} \cdot \frac{df_g}{dS_g} \leq 1
\end{aligned} \tag{3.39}$$

which is the same equation as Eqn. 3.36. Eqn. 3.37 is used as the stability criterion for the explicit treatment of saturations in GPRS.

If any component mole fraction is treated explicitly, we can get its stability criterion from Eqn. 3.30. Assuming incompressible flow and constant equilibrium ratio  $K_c$ , we have

$$\begin{aligned} & (\phi \rho_o S_o / K_c + \phi \rho_g S_g) \frac{\partial y_c}{\partial t} + \phi (\rho_g y_c - \rho_o x_c) \frac{\partial S_g}{\partial t} \\ &= (\rho_o u_o / K_c + \rho_g u_g) \frac{\partial y_c}{\partial x} + u_T (\rho_g y_c - \rho_o x_c) \frac{df_g}{dS_g} \frac{\partial S_g}{\partial x} \end{aligned} \quad (3.40)$$

Substituting Eqn. 3.34 into Eqn. 3.40, we can eliminate the second terms on both sides, and after rearranging, we get

$$\frac{\partial y_c}{\partial t} = \frac{\rho_o u_o x_c + \rho_g u_g y_c}{\phi \rho_o S_o x_c + \phi \rho_g S_g y_c} \frac{\partial y_c}{\partial x} \quad (3.41)$$

Dividing by  $K_c$  on both sides of Eqn. 3.41, we get

$$\frac{\partial x_c}{\partial t} = \frac{\rho_o u_o x_c + \rho_g u_g y_c}{\phi \rho_o S_o x_c + \phi \rho_g S_g y_c} \frac{\partial x_c}{\partial x} \quad (3.42)$$

The stability criterion for both Eqn. 3.41 and Eqn. 3.42 is

$$CFL_{x,x} = \frac{\Delta t}{\Delta x \phi} \frac{\rho_o u_o x_c + \rho_g u_g y_c}{\rho_o S_o x_c + \rho_g S_g y_c} \leq 1 \quad (3.43)$$

Similarly, for three-dimensional flow, the stability criterion is

$$CFL_x = \frac{\Delta t}{V \phi} \frac{\rho_o q_o x_c + \rho_g q_g y_c}{\rho_o S_o x_c + \rho_g S_g y_c} \leq 1 \quad (3.44)$$

Coats (1999) mentioned that the stability criterion due to the explicit treatment of component mole fractions is  $CFL_x = \frac{\Delta t}{V \phi} \cdot \frac{q_T}{1 - S_w} \leq 1$ , which is only a simplified version of Eqn. 3.44 for  $S_o=0$  or  $S_g=0$ . In Coats' new paper (Coats, 2001), the same stability criterion as developed here (Eqn. 3.44) was proposed, but he got it by first assuming

$q_o=0$  or  $q_g=0$ , and getting the stability criterion for the case where only single hydrocarbon phase can flow, then finally generalizing it to the case where both hydrocarbon phases flow. Our derivation is more general.

The physical meaning of Eqn. 3.44 is, for a stable timestep, the amount of component  $c$  flowing out of a gridblock can not exceed the amount of component  $c$  in place in that gridblock. In other words, a component can not move more than one gridblock in a stable timestep. Eqn. 3.44 is for each individual component, and it only needs to be satisfied when either  $y_c$  or  $x_c$  is treated explicitly. Eqn. 3.44 is used as the stability criterion for the explicit treatment of component mole fractions in GPRS. We will further analyze and compare these two stability criteria (Eqn. 3.37 and Eqn. 3.44) in Chapter 4.

### 3.4 Influence of Equation and Variable Selections

Compositional models are different in both their implicit levels and their equation and variable selections. In the last section we have discussed the stability criteria due to the explicit treatment of variables. In this part, we consider the influence of equation and variable selections on the efficiency of compositional simulations. First we discuss the effect of selecting primary equations, then we compare the differences between different variable selections. For each we will discuss their influence on Newton iterations and linear solver iterations.

#### 3.4.1 Selection of Primary Equations

In reservoir simulation, we can select different equations, but no matter what equations we select, each equation is basically a linear combination (first assign a coefficient to each equation, then sum them up) of all of the equations shown in Subsection 1.2.1. For example, the total mass balance equation is the summation of all of the component mass balance equations, in this case the vector of coefficients is  $(1, 1, 1, \dots, 1)$ . If we consider the ordering of equations, a new ordering can be viewed as a special linear combination of the old ordering. Such as, coefficients  $(0, 0, 0, 1, \dots, 0)$  move the fourth equation to the current location. Equation selection is done gridblock by gridblock, the Jacobian matrix and the RHS for one gridblock can be written as

$$A_i \cdot \delta X = B_i \quad (3.45)$$

where  $A_i$  is one row of the full Jacobian matrix  $A$ ,  $B_i$  is one row of the RHS  $B$ , and  $\delta X$  is all of the unknowns. Any combination of equations of gridblock  $i$  can be represented by pre-multiplying Eqn. 3.45 by a matrix on both sides:

$$M_i \cdot (A_i \cdot \delta X) = M_i \cdot B_i \quad (3.46)$$

Eqn. 3.46 is only for one gridblock. Equations for individual blocks can be combined to obtain the final full system:

$$M \cdot (A \cdot \delta X) = M \cdot B \quad (3.47)$$

$$\text{where, } M = \begin{bmatrix} M_1 & & & \\ & M_2 & & \\ & & \dots & \\ & & & M_n \end{bmatrix}, A = \begin{bmatrix} A_1 \\ A_2 \\ \dots \\ A_n \end{bmatrix}, \text{ and } B = \begin{bmatrix} B_1 \\ B_2 \\ \dots \\ B_n \end{bmatrix}$$

Eqn. 3.47 can be recast as

$$(M \cdot A) \cdot \delta X = (M \cdot B) \quad (3.48)$$

Here, the Jacobian matrix changes to  $MA$ , the RHS changes to  $MB$ , but the solution vector is still  $\delta X$ .

The solution vector is always  $\delta X$  no matter what the  $M$  is, the same solution is guaranteed as long as Eqn 3.48 is solved accurately. Same solutions for  $X$  lead to the same residuals at each Newton iteration. Since the convergence of Newton iterations depends on the reduction of residuals, the number of Newton iterations will stay the same. In other words, equation selection will not influence Newton iterations.

On the other hand, the change of the Jacobian matrix will influence linear solver iterations, because  $A$  and  $MA$  have different condition numbers. Alternatively, we can view Eqn. 3.48 as a left preconditioning of the old linear system  $A \cdot \delta X = B$ , where  $M$  is the preconditioning matrix. In this case  $M$  is a block diagonal matrix, so it is some kind of block diagonal scaling left preconditioning. In this sense, the best equation selection is

the one that makes the block diagonals of  $MA$  as close to a diagonal matrix as possible, just as the block diagonal scaling preconditioner does. In other words, we like to select and align the equations in such a way that equation  $i$  is most influenced by variable  $i$ , then the block diagonals of  $MA$  will be as near to diagonal as possible (diagonal dominance).

Based on these observations, we can conclude that, at the Jacobian level, different equation selections are equivalent to performing different block diagonal left preconditionings to the old system, which only influence linear solver iterations and has no influence on Newton iterations. A good equation selection should roughly align the equations in the way that variable  $i$  has the biggest influence on equation  $i$  (diagonal dominance). Applying a block diagonal scaling left preconditioning to the Jacobian matrix could achieve the same effect. Of course, different equation selections will influence the amount of work in calculating the Jacobian matrix and the RHS.

### 3.4.2 Different Variable Selections

Different models can also use different variables. In this part, we will first consider the influence of different variable selections on Newton iterations and linear solver iterations, then we will discuss the differences between using intensive variables and extensive variables, finally we will compare the performance of models with Type A and Type B variables in terms of their mass and volume balance errors. In Subsection 3.2.1, we have mentioned that we can change the variable selection by multiplying the equation by a transformation matrix. This transformation matrix is for each gridblock, and this process is done gridblock by gridblock. We can combine all of them together, then at the full Jacobian level, this process looks like

$$(AM) \cdot (M^{-1}\delta X) = B \quad (3.49)$$

$$\text{where, } M = \begin{bmatrix} M_1 & & & \\ & M_2 & & \\ & & \dots & \\ & & & M_n \end{bmatrix}.$$



Here, the Jacobian matrix changes to  $AM$ , the solution vector changes to  $\delta Y = M^{-1}\delta X$ , but the right hand side is still  $B$ .

The change of the Jacobian matrix will influence linear solver iterations, because  $A$  and  $AM$  have different condition numbers. Alternatively we can view Eqn. 3.49 as a right preconditioning of the old system  $A \cdot \delta X = B$ , where  $M$  is the preconditioning matrix. In this case  $M$  is also a block diagonal matrix, so it is some kind of block diagonal scaling right preconditioning. For the same reason as for the selection of equations, it is best to align the variables in the way that makes variable  $i$  to have the biggest influence on equation  $i$  (diagonally dominant).

By preconditioning, we solve for  $X$  by  $X^{v+1} = X^v + \delta X^v = X^v + M\delta Y^v$  and we get the same solution as for the old system,  $A \cdot \delta X = B$ . But for the new variables, we must first solve for  $Y$  by  $Y^{v+1} = Y^v + \delta Y^v$ , then  $X$  of each gridblock can be updated explicitly by the relation  $X = f^{-1}(Y)$  that was built between  $X$  and  $Y$  for the calculation of transformation matrix. In this case, we will get a different solution  $X$  if the relation  $X = f^{-1}(Y)$  is nonlinear. Such as for  $F = X^2 - 2 = 0$  with initial guess  $X^0 = 1$ , after the first Newton iteration, we get  $\delta X^0 = 0.5$  and  $X^1 = X^0 + \delta X^0 = 1.5$ , if we change  $X$  to  $Y = X^2$ , then the equation becomes  $F = Y - 2 = 0$  with initial guess  $Y^0 = 1$ , after the first Newton iteration, we have  $\delta Y^0 = 1$ ,  $Y^1 = Y^0 + \delta Y^0 = 2$  and  $X^1 = \sqrt{Y^1} = \sqrt{2}$ , and these two solutions for  $X^1$  are different. Different solutions for  $X$  lead to different residuals at each Newton iteration. Since the convergence of Newton iterations depends on the reduction of residuals, variable selection will influence Newton iterations. If the equations are more linear with respect to the new variables, then the number of Newton iterations will decrease (for linear system, only one Newton iteration is needed to converge). Variable selection will also influence the amount of work in calculating the Jacobian matrix and the RHS. The natural variables are the cheapest ones for building a fully implicit system.

For the same type of variable, it is also possible to use an extensive variable, such as the overall component molar density  $r_c$  vs. the overall component moles  $m_c$ . In order to understand the difference between these two types of variables, we need to recall that, no matter which kinds of variables are used, the number of independent variables is always

the same. Consider a compositional model using the natural variables, which are all intensive variables, after the natural variables have been calculated and fixed, all of the extensive variables of this system will also be fixed. For example the overall component mole  $m_c$  can be calculated as

$$m_c = V\phi(\rho_o S_o x_c + \rho_g S_g y_c) \quad (3.50)$$

But sometimes, models using extensive variables use one more equation and one more variable than necessary. Such as for a three-phase black-oil model, if we use pressure and saturations as variables, one of the saturations can be easily removed by using the saturation constraint equation, since it is a linear equation with respect to saturations. But if we use pressure and overall component moles  $m_c$  as variables, we can not easily remove one of the overall component moles, since the relation between them is nonlinear. For example in the case of black-oil models, we have

$$V\phi = \frac{B_w}{\rho_w} m_w + \frac{B_g}{\rho_g} m_g + \frac{B_o - R_{g,o} B_g}{\rho_o} m_o \quad (3.51)$$

Eqn. 3.51 is actually the volume balance equation for black-oil models. For convenience this volume balance equation is treated as part of the nonlinear equation set to complete the system when calculating the Jacobian matrix and the RHS. This does not mean that the degrees of freedom of this system is 4, in fact, it is only 3. The volume balance equation is only a constraint equation (only has derivatives in the diagonal part of the Jacobian matrix), and each of the  $4 \times 4$  small matrices can be reduced to a smaller  $3 \times 3$  matrix.

Is there any benefit in using extensive variables? Yes, sometimes, but not too much. If the overall component moles  $m_c$  are used, then the accumulation terms,  $(m_c^{n+1} - m_c^n) / \Delta t$  will be linear with respect to these variables, and the mass balance is exact at each Newton iteration (Farkas, 1997). If the overall component molar densities  $r_c$  are used, then the accumulation term,  $[(\phi r_c)^{n+1} - (\phi r_c)^n] \cdot V / \Delta t$  is nearly linear with respect to these variables if we consider that porosity changes are small with pressure, and the mass balance will be nearly exact at each Newton iteration (Farkas, 1997). But if the overall component mole fractions  $z_c$  are used, then the accumulation term,

$[(\phi \rho_T z_c)^{n+1} - (\phi \rho_T z_c)^n] \cdot V / \Delta t$  will be nonlinear, since the total fluid density  $\rho_T = \rho_o S_o + \rho_g S_g$  changes a lot with both pressure and overall component mole fractions, this will cause mass balance errors at each Newton iteration. In summary, if the rock compressibility is much smaller than the fluid compressibility, such as for gas flow, a model using the overall component molar densities  $r_c$  should have similar performance as a model using the overall component moles  $m_c$ . If the rock is incompressible,  $m_c = V \phi r_c$  and the only difference between  $r_c$  and  $m_c$  is a constant scalar ( $V \phi$ ), and both models should have the same performance.

In Chapter 1, we mentioned that there are two types of variables, Type A and Type B. Next we will compare the models using these two types of variables in terms of mass balance and volume (saturation) balance errors.

- **Type A models** There is mass balance error at each Newton iteration, due to the nonlinearity of the accumulation terms with respect to these variables. The volume balance is exact at all times, since the saturation constraint equation is a linear equation. Both balances will be exact when the model converges at the end of each timestep (Farkas, 1997).
- **Type B models** If  $r_c$  and  $m_c$  are used, there is virtually no mass balance error at each Newton iteration, but there is a volume balance error, now the saturation constraint equation is a nonlinear equation with respect to these variables. Again, both balances will be exact when the model converges at the end of each timestep (Farkas, 1997).

In reservoir simulation, mass balance is a much more desirable property than volume balance. For Type A variables, Coats (1999) proposed a new approach to achieve exact mass balance at each Newton iteration. In each iteration step, the Jacobian variables are used to calculate well rate and inter-block fluxes, and the component mass changes are calculated from the mass balance equations. As a result, there is no mass balance error. But the mass balance errors are transformed into volume balance error in each iteration step, and the relaxed volume balance method is used to control the volume balance error.

Is there a strong connection between mass balance and Newton iteration numbers? No. Mass balance at each Newton iteration depends on the linearity of the accumulation term. The Newton iteration number depends on the linearity of the flow equations, which

include three parts, the accumulation part, the flux part and the well part. For exact mass balance, the accumulation part must be linear with respect to the selected variables, but the degree of nonlinearity of the flux and well parts with respect to the selected variables are still unknown. In compositional simulations, we are solving the mass balance equations and the phase equilibrium relations together, and both of them are nonlinear. The Newton iteration number depends on the nonlinearity of both sets of equations, the nonlinearity of phase equilibrium relations with respect to different variables is even more unpredictable. Due to these considerations, we conclude that for compositional simulations, the Newton iteration number is not strongly influenced by the mass balance. The results in Chapter 5 further validate this point.

Based on the above discussion, we can draw the following conclusions for different equation and variable selections:

- Different equation and variable selections will change linear solver iterations, and a good equation and variable selection should align the equations and variables in such a way that variable  $i$  has the biggest influence on equation  $i$  (diagonal dominance). The same effect can be achieved by applying block diagonal preconditioners for the Jacobian matrix.
- Different equation selections will not change the behavior of Newton iterations, while different variable selections can change the behavior of Newton iterations, which depend on the nonlinearity of the equations with respect to different variables.
- No matter what types of variables are used, intensive or extensive, the degrees of freedom of the system are always the same.
- Mass balance is a nice property to have, but it does not have a direct connection with the convergence of the non-linear system (Newton iteration numbers).

Of course, when selecting equations and variables, the cost of calculating the Jacobian matrix and the RHS should also be considered.

## Chapter 4

### IMPSAT Model and IMPSAT Based AIM Model

The General Formulation Approach has the advantage of allowing selection of any level of implicitness. For isothermal compositional models, the implicit level (number of implicit variables per gridblock) ranges from 1 in IMPES models to  $n_c$  in fully implicit (FIM) models. With the increase of implicit level, stability increases, and we can use larger timestep size with fewer Newton iterations. However with the increase in number of implicit variables, the computational cost per Newton iteration also increases dramatically. In IMPES models, the cost of each linear solve is fixed relative to  $n_c$ , because we always solve for one variable per gridblock, but in FIM models, the cost of each linear solve increases as  $O(n_c^{1.1 \sim 1.2})$ . The question here is how to balance these two factors, and select a suitable implicit level that is fast enough and stable enough.

To make a simulation run fast, we need to reduce the number of unknowns per gridblock from  $n_c$ . This number is minimum for IMPES and maximum for FIM. We also like a model to only solve for a fixed number (relative to  $n_c$ ) of unknowns per gridblock as is the case for the IMPES model. Under these constraints, implicit pressure and implicit saturations appear to be desirable, because both variables are independent of the number of components. Here we propose the IMPSAT model, where only pressure and saturations are treated implicitly, and all of the component mole fractions are treated explicitly. Since one of the saturations can be removed using the saturation constraint equation, we only have  $n_p$  unknowns per gridblock. In reservoir simulation, typically we have two or three phases, so the number of unknowns per gridblock for the IMPSAT model is three or less. Compared to  $n_c$  unknowns per gridblock for the FIM model. Thus the IMPSAT model has the potential to yield big savings in computational cost per linear solve, especially for problems with large number of components.

In fact, the IMPSAT model is not new, Quandalle and Savary (1989) and Branco and Rodriguez (1995) also proposed similar ideas. In Quandalle and Savary's paper, it was

called IMPIMS (implicit pressure and implicit saturations) model, and they used it to simulate a laboratory experiment. In Branco and Rodriguez's paper, it was called semi-implicit model, and several small test problems were used for validation. Neither paper provided a detailed analysis of the model, no stability criteria were derived and no comparisons with the IMPES and FIM models were provided. Furthermore all of the tests were on small simple problems. In order to prove that IMPSAT works in practice, more work was needed in all of these areas. As a matter of fact IMPSAT is an option in at least one of the commercial simulators, but it does not work properly. In this study we will show that when properly implemented, IMPSAT works very well for most cases. Hopefully this will lead to greater use of this method.

In this chapter, we will first analyze various aspects of the IMPSAT model, then we will introduce a new IMPSAT based AIM model, where IMPSAT formulation is added into the traditional AIM model, IMPES+FIM (Forsyth and Sammon, 1986). We believe that this new AIM model can replace the traditional AIM model, especially for large field case studies.

## **4.1 IMPSAT Model**

In this section, we will answer the following four questions about the IMPSAT model: Why does it work, what is its stability criterion, what is its cost and how to build it most efficiently? In answering each question, we will discuss advantages and disadvantages of IMPSAT compared to the IMPES and FIM models.

### **4.1.1 Why Does IMPSAT Work?**

In order to explain why IMPSAT works, we need to take a look at the variable coupling between gridblocks. For the natural variable selection, we have three types of variables, pressure, saturation and mole fractions. Each of them is coupled from gridblock to gridblock due to the flux terms in the mass balance equations. For one-dimensional flow without gravity and capillary pressure, each component flux within a phase can be expressed as the product of a transmissibility term and a pressure difference between two neighboring gridblocks:

$$Flux_{c,p} = T_{cp} \cdot \Delta p, \quad \text{and } T_{c,p} = C \cdot \lambda_p \rho_p x_{cp} \quad (4.1)$$

The transmissibility term has four parts, a constant  $C$ , a phase mobility  $\lambda_p$  which only depends on pressure and saturations, a phase density  $\rho_p$  which only depends on pressure and component mole fractions, and a component mole fraction  $x_{cp}$ . In reservoir simulation, this transmissibility term is always evaluated at upstream conditions. Depending on the flow directions, the flux term of gridblock  $i$  could depend on the variables in gridblock  $i$ ,  $i+1$  and  $i-1$ :

$$Flux_{c,p,i} = Flux_{c,p,i}(p_{i-1}, p_i, p_{i+1}, S_{p,i-1}, S_{p,i}, S_{p,i+1}, x_{c,p,i-1}, x_{c,p,i}, x_{c,p,i+1}) \quad (4.2)$$

Due to this, all of the variables are coupled from gridblock to gridblock, and we need to solve for all of them together, as is the case for the FIM model.

The degree of coupling is different for each variable. The coupling between pressures exists in both the transmissibility part and the pressure difference part. While the coupling between saturations and between mole fractions only exists in the transmissibility part. If we fix the transmissibility term, the flux term will become a linear function of gridblock pressures:

$$Flux_{c,p,i} = Flux_{c,p,i}(p_{i-1}, p_i, p_{i+1}) \quad (4.3)$$

From Eqn. 4.3, we can see that, pressure is always strongly coupled from gridblock to gridblock, while saturations and mole fractions are loosely coupled from gridblock to gridblock. The IMPES model is based on this observation, it ignores the coupling of saturations and mole fractions between gridblocks by treating the transmissibilities explicitly. Once this is done, pressures are decoupled and can be computed implicitly followed by explicit updating of saturations and mole fractions, gridblock by gridblock.

Here, we make a new claim: the coupling between mole fractions is even weaker than the coupling between saturations. For saturations and mole fractions, the ratios of the flux term derivatives to the accumulation term derivatives are compared, which are also the ratio of the off-diagonal term to the diagonal term in the Jacobian matrix, and roughly

represent the degree of coupling between gridblocks for each variable. For saturations, this ratio can be expressed as

$$Ratio_s = \frac{\Delta(Flux)}{\Delta(ACC)} = \frac{\rho_p \frac{nS_p^{n-1}}{\mu_p} x_{cp} \Delta p}{\phi \rho_p x_{cp}} = n \cdot \left( \frac{S_p^{n-1}}{\phi \mu_p} \Delta p \right) \quad (4.4)$$

For mole fractions, this ratio can be expressed as

$$Ratio_x = \frac{\Delta(Flux)}{\Delta(ACC)} = \frac{\rho_p \frac{S_p^n}{\mu_p} \Delta p}{\phi \rho_p S_p} = \left( \frac{S_p^{n-1}}{\phi \mu_p} \Delta p \right) \quad (4.5)$$

Here we assume a power relation ( $k_{rp} = S_p^n$ ,  $n \geq 1$ ) for phase relative permeabilities and ignore the dependency of phase densities on mole fractions. Comparing Eqn. 4.4 with Eqn. 4.5, we can see that the coupling between saturations is approximately  $n$  times stronger than the coupling between mole fractions. For three-dimensional problems, due to the gravity effect, the large density and viscosity difference between the gas and the oil phases will enhance the movement of phase front, and this coupling difference could be even larger. The existence of capillary pressure will also increase this difference, since capillary pressure is assumed to be only a function of saturations. Based on these observations, we propose the IMPSAT model, where the phase density and the component mole fraction in the transmissibility part are treated explicitly, and the flux terms are only functions of pressures and saturations:

$$Flux_{c,p,i} = Flux_{c,p,i}(p_{i-1}, p_i, p_{i+1}, S_{p,i-1}, S_{p,i}, S_{p,i+1}) \quad (4.6)$$

In the IMPSAT model, both pressure and saturations are simultaneously solved for first, then component mole fractions are updated explicitly gridblock by gridblock. Since we only ignore the coupling between mole fractions, which is generally weak, the IMPSAT model should be more stable than the IMPES model. Hopefully this improved stability will allow approximately the same size timesteps as in the FIM model.

Next we will derive the stability criterion for the IMPSAT model and compare it with the stability criterion of the IMPES model.



#### 4.1.2 IMPSAT Stability Criterion

In order to reduce the number of timesteps and Newton iterations, a more stable model than IMPES is desirable. In the IMPSAT model, only the mole fractions are treated explicitly, so its stability criterion only needs to satisfy Eqn. 3.44, which is

$$CFL_{IMPSAT} = \frac{\Delta t}{V\phi} \frac{\rho_o q_o x_c + \rho_g q_g y_c}{\rho_o S_o x_c + \rho_g S_g y_c} \leq 1 \quad (4.7)$$

In the IMPES model, both the saturations and the component mole fractions are treated explicitly, its stability criterion should satisfy both Eqn. 3.37 and Eqn. 3.44, which can be written as

$$CFL_{IMPES} = \frac{\Delta t}{V\phi} \cdot \text{MAX} \left( \frac{\frac{\lambda_o}{\lambda_g} \frac{d\lambda_g}{dS_g} q_g - \frac{\lambda_g}{\lambda_o} \frac{d\lambda_o}{dS_g} q_o}{\lambda_o + \lambda_g}, \frac{\rho_o q_o x_c + \rho_g q_g y_c}{\rho_o S_o x_c + \rho_g S_g y_c} \right) \leq 1 \quad (4.8)$$

From these two equations, we can see that, the IMPSAT model is always more stable than the IMPES model, but by how much? For a one-dimensional problem without gravity, Eqn. 4.7 and Eqn. 4.8 can be simplified as

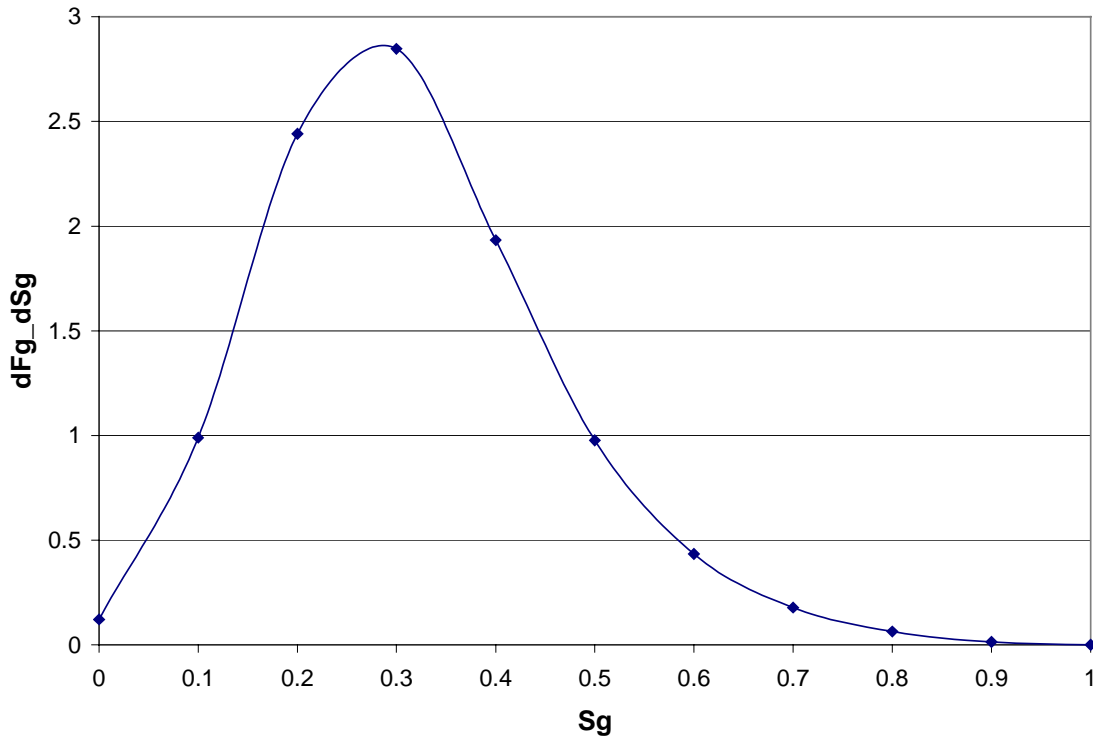
$$CFL_{IMPSAT} = \frac{q_T \Delta t}{V\phi} \cdot V_c \leq 1 \quad (4.9)$$

$$CFL_{IMPES} = \frac{q_T \Delta t}{V\phi} \text{MAX} \left( \frac{df_g}{dS_g}, V_c \right) \leq 1 \quad (4.10)$$

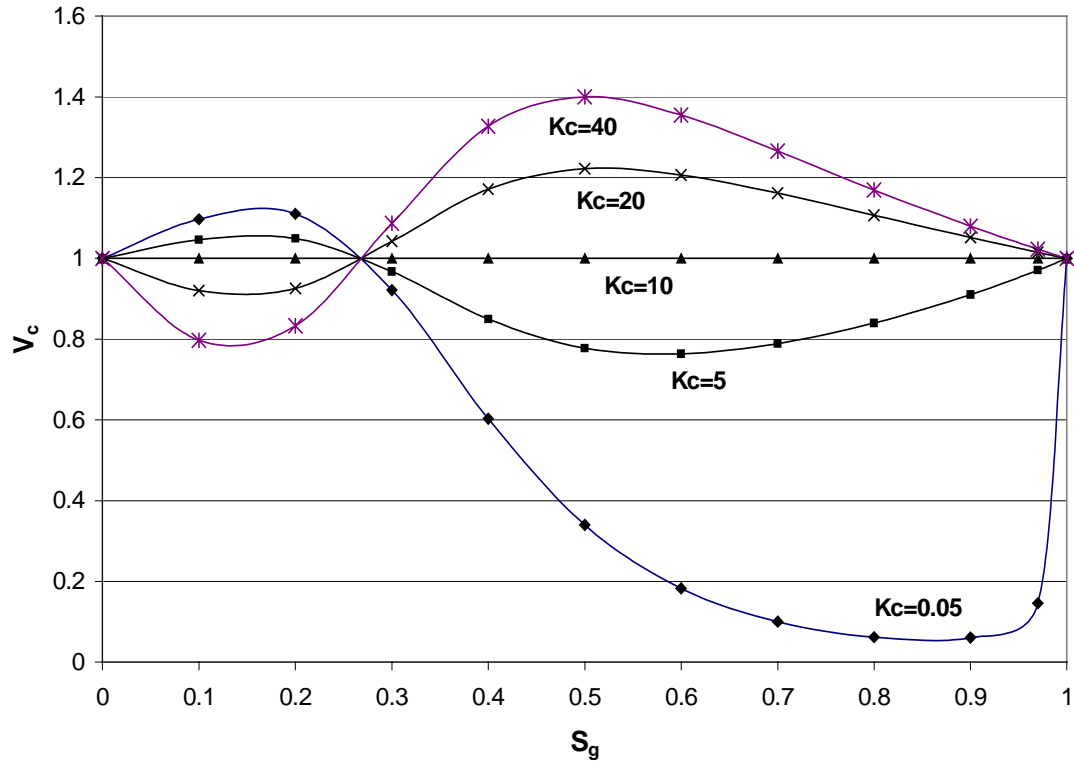
$$\text{where } V_c = \frac{\rho_o + f_g (\rho_g K_c - \rho_o)}{\rho_o + S_g (\rho_g K_c - \rho_o)} = \begin{cases} f_o / S_o, & \text{when } K_c \rightarrow 0 \\ 1, & \text{when } K_c = \rho_o / \rho_g \\ f_g / S_g, & \text{when } K_c \rightarrow \infty \end{cases} \quad (4.11)$$

The difference between these two CFL numbers depends directly on the relative size of  $df_g / dS_g$  and  $V_c$ . We can plot these two functions for a typical fractional flow curve ( $k_{rg} = S_g^3$ ,  $k_{ro} = S_o^2$ ,  $\mu_o / \mu_g = 10$  and  $\rho_o / \rho_g = 10$ ).  $df_g / dS_g$  is shown as a function of  $S_g$  in Figure 4.1. Note that  $df_g / dS_g$  has a maximum value of 2.8.  $V_c$  is shown as a function of  $S_g$  for different  $K_c$  values in Figure 4.2, different  $K_c$  values correspond to

different components, and lighter components have larger  $K_c$  values since lighter components are more likely to stay in the gas phase. The maximum value in Figure 4.2 is around 1.4. For this example, we can say that the IMPSAT model is about two times more stable than the IMPES model. From Figure 4.2, we notice that lighter components have larger  $K_c$  and  $V_c$  values, so they cause more instability compared to the heavier components. This is because lighter components are more likely to stay in the gas phase, which has a much higher flow rate than the oil phase due to its small viscosity. If we include two extra variables  $x_l$ ,  $y_l$  (component  $l$  is the lightest one) in the implicit variables, this model (IMPSAT++) will become even more stable.

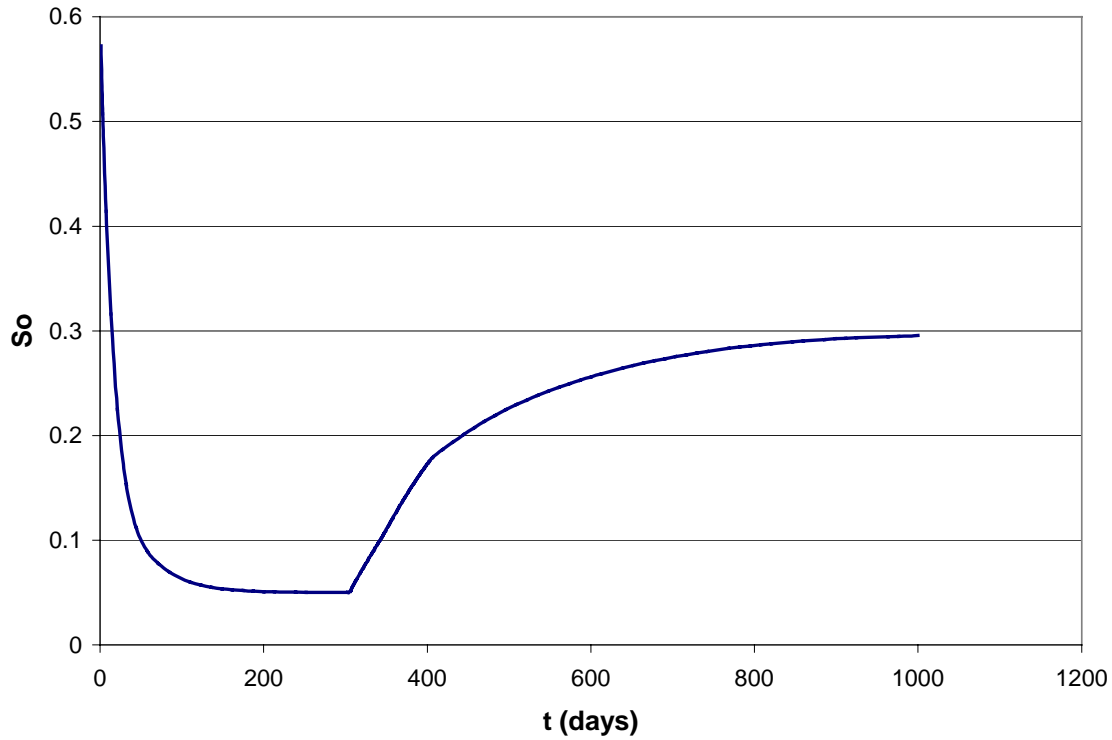


**Figure 4.1**  $df_g / dS_g$  as a function of  $S_g$



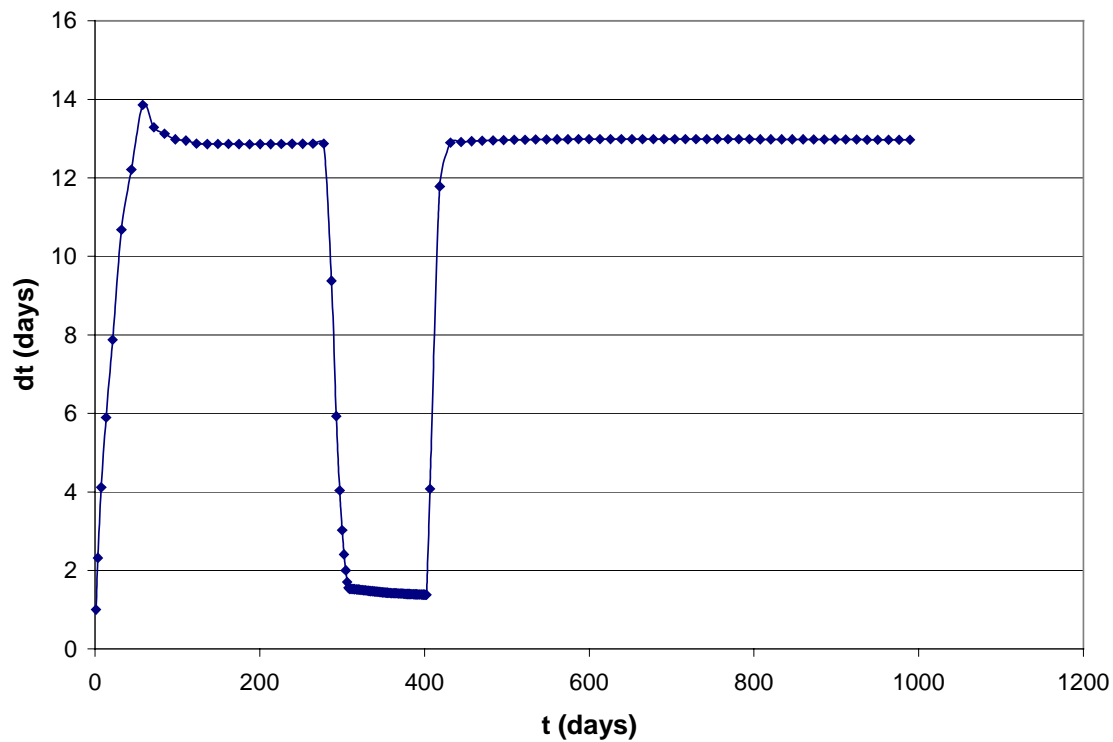
**Figure 4.2**  $V_c$  as a function of  $S_g$  for difference  $K_c$  values

For three-dimensional problems, where gravity is important, the large density and viscosity difference between the gas and the oil phases will enhance the movement of phase front, and the stability difference between the IMPSAT and the IMPES models will be much larger than two times. Here we compare their stable timestep sizes for a three-dimensional (5x5x5) four-component (C1, C2, C4 and C7) compositional simulation with one corner producer completed only in the top layer (Cao, 2002). Figure 4.3 shows the oil saturation at the well block. Initially the reservoir was not at equilibrium, since the same initial overall composition was used for all depths. Up to 300 days gravity dominated the process, causing phase segregation. After about 400 days gravity segregation is complete and the oil phase starts to cone around the producer.

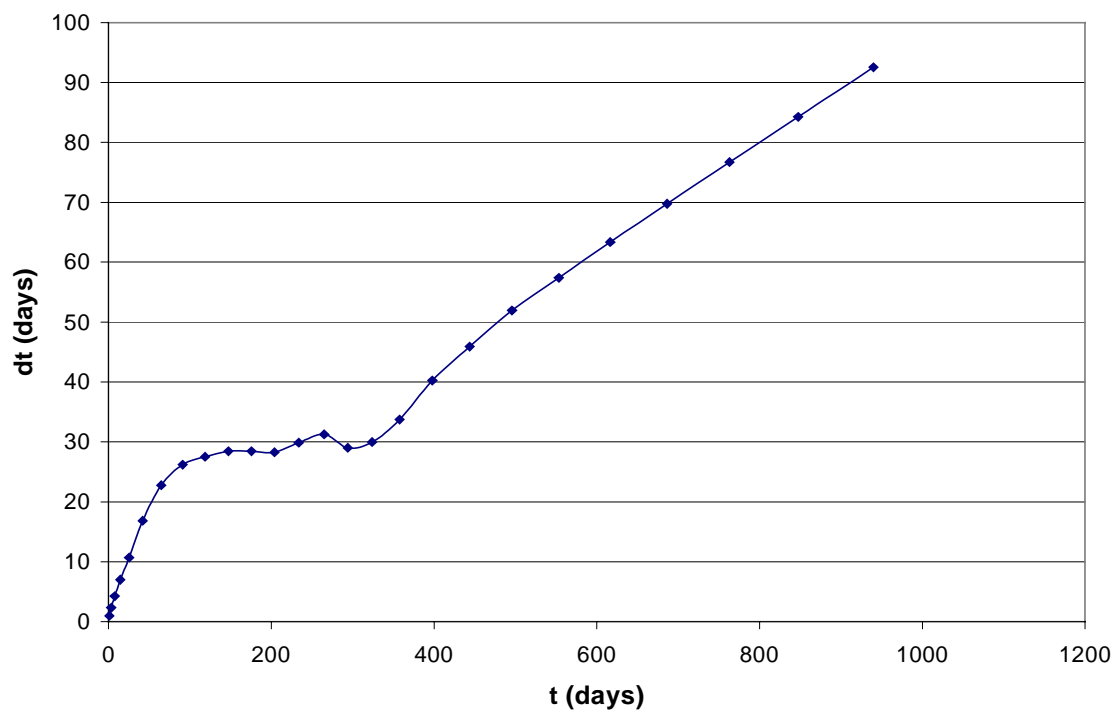


**Figure 4.3** Oil saturation at the well block

Figure 4.4 reports the maximum stable timestep size of the IMPES model, which was calculated from Eqn. 4.8. Here the maximum stable timestep size was around 13 days, and during the transition period between 300 and 400 days, it was smaller than 2 days. Figure 4.5 reports the maximum stable timestep size of the IMPSAT model, which was calculated from Eqn. 4.7. Here, the maximum stable timestep size ranged from 30 days to over 90 days, also no problems were observed at the transition period. The ratio between these two stable timestep sizes is from 2 to 7 times.



**Figure 4.4** Maximum stable timestep size for the IMPES model



**Figure 4.5** Maximum stable timestep size for the IMPSAT model

The results in Figure 4.4 and Figure 4.5 are for CFL=1. In practice, we can use a larger CFL number, for IMPES models, normally CFL=2 is fine (Coats, 2001). For the IMPSAT model, we have found that CFL=3 or 4 is generally a safe limit. This means that in practice, the IMPSAT model has even greater advantage than that demonstrated by the example. The comparison between the IMPES and the IMPSAT models for larger CFL numbers are included in Chapter 5. In general, for most three-dimensional problems, the stable timestep size of the IMPSAT model can be one order of magnitude larger than the stable timestep size of the IMPES model. We have demonstrated that the IMPSAT model is much more stable than the IMPES model, how about the cost of the IMPSAT model compared to the costs of the FIM and IMPES models? In the next subsection, we will analyze the cost of different compositional models.

#### 4.1.3 Cost of IMPSAT Model

In order to compare the cost (CPU time) of the IMPSAT model with the costs of the FIM and IMPES models, it is necessary to analyze the cost of simulation for models with different implicit levels. In compositional simulations, the most time consuming parts for each Newton iteration are the Jacobian construction, the flash calculation and the linear solve. The cost of each part depends on implementation details of the individual simulators, here we provide some rough estimates for each model.

The cost of the Jacobian construction depends on both the selection of primary variables and the implicit level, and it changes significantly from model to model. For FIM models, it can be expressed as

$$T_{Jac,FIM} = N_{Newton} [C_{Jac,block}^{FIM} N_{block} N_c^2 + C_{Jac,conn}^{FIM} N_{conn} N_c^2] \quad (4.12)$$

where  $N_{block}$  is the total number of gridblocks,  $N_{conn}$  is the total number of connections between gridblocks, and  $C_{Jac,block}^{FIM}$  and  $C_{Jac,conn}^{FIM}$  are the coefficients (constants). The first term in Eqn. 4.12 is the cost of calculating each of the  $N_c \times N_c$  diagonal entries of the Jacobian matrix, and the second term in Eqn. 4.12 is the cost of calculating each of the  $N_c \times N_c$  off-diagonal entries of the Jacobian matrix.  $C_{Jac,block}^{FIM}$  and  $C_{Jac,conn}^{FIM}$  are different for different simulators and different variable selections, and generally  $C_{Jac,block}^{FIM}$  and

$C_{Jac,conn}^{FIM}$  of the FIM model with Type B variables are larger than those of the FIM model with Type A variables, because we need to use the chain rule to calculate the derivatives in the Type B FIM model. For the IMPES model with Type A variables, the cost of the Jacobian construction is approximately given by

$$T_{Jac,IMPES\_A} = N_{Newton} [(C_{Jac,block}^{IMPES\_A} N_{block} N_c^2 + C_{Jac,conn}^{IMPES\_A} N_{conn} N_c) + C_{GFA} N_{block} (N_c - 1)^3] \quad (4.13)$$

where the first two terms are the costs of calculating the full Jacobian matrix, where all of the off-diagonal entries are  $N_c \times 1$  matrices, the last term is the cost of reducing the implicit level from  $N_c$  to 1 (inverting each small matrix D, whose size is  $(N_c - 1) \times (N_c - 1)$ ), and  $C_{GFA}$  is the corresponding coefficient. For the IMPES model with Type B variables, since the accumulation term of the mass balance equations can be simply expressed as  $[(\phi r_c)^{n+1} - (\phi r_c)^n] \cdot V / \Delta t$ , all diagonal entries of the full Jacobian will only have non-zero terms at the diagonal and one column, and the cost of the Jacobian construction is only

$$T_{Jac,IMPES\_B} = N_{Newton} [(C_{Jac,block}^{IMPES\_B} N_{block} N_c + C_{Jac,conn}^{IMPES\_B} N_{conn} N_c) + C_{GFA} N_{block} (N_c - 1)] \quad (4.14)$$

For the IMPSAT model (Type A variables), the cost of the Jacobian construction can be expressed as

$$T_{Jac,IMPSAT} = N_{Newton} [(C_{Jac,block}^{IMPSAT} N_{block} N_c^2 + C_{Jac,conn}^{IMPSAT} N_{conn} N_c N_p) + C_{GFA} N_{block} (N_c - N_p)^3] \quad (4.15)$$

Similarly, the first two terms are the costs of calculating the full Jacobian matrix, where all of the off-diagonal entries are  $N_c \times N_p$  matrices, the last term is the cost of reducing the implicit level from  $N_c$  to  $N_p$  (inverting each small matrix D, whose size is  $(N_c - N_p) \times (N_c - N_p)$ ), and  $C_{GFA}$  is the corresponding coefficient. For models with  $N_i$  ( $N_i \leq N_c$ ) implicit variables, Eqn. 4.15 (cost of the Jacobian construction) can be generalized as

$$T_{Jac} = N_{Newton} [(C_{Jac,block} N_{block} N_c^2 + C_{Jac,conn} N_{conn} N_c N_i) + C_{GFA} N_{block} (N_c - N_i)^3] \quad (4.16)$$

The cost of the flash calculation is independent of the implicit level, and it can be expressed as

$$T_{flash} = N_{Newton} [C_{Jac}^{flash} N_{block} N_c^2 + C_{solve}^{flash} N_{block} N_c^{n_{flash}}] \quad (4.17)$$

where the first term is the cost of calculating the  $N_c \times N_c$  flash Jacobian matrix for each gridblock and the second term is the cost of solving it, and  $C_{solve}^{flash}$  and  $n_{flash}$  depend on the solver used.

The cost of each linear solve depends on the total number of implicit unknowns ( $N_{block} N_i$ ), and it can be written as

$$T_{linear\_solve} = N_{Newton} [C_{solver} (N_{block} N_i)^{n_{solver}}] \quad (4.18)$$

where  $C_{solver}$  and  $n_{solver}$  depend on the linear solver used. Typically  $n_{solver} = 1.1 \sim 1.2$  for good iterative linear solvers.

The total cost of simulation is the summation of the costs of these three parts.

$$T_{total} = T_{Jac} + T_{flash} + T_{linear\_solve} \quad (4.19)$$

From Eqn. 4.13 to Eqn. 4.19, we can see that if we know the total number of Newton iterations, we can roughly compare the costs of different models. We have demonstrated that the IMPSAT model is much more stable than the IMPES model in Subsection 4.1.2 and compared its cost with the costs of the FIM and IMPES models in Subsection 4.1.3. Next we will introduce an efficient way to construct the IMPSAT model.

#### 4.1.4 How to Build IMPSAT Efficiently?

There is a very simple way to build the IMPSAT model. In both the water mass balance equation (Eqn. 1.2) and the total hydrocarbon mass balance equation (Eqn. 1.6), only pressure and saturations appear explicitly. Since the water phase density depends only on pressure, the water mass balance equation only includes pressure and saturation dependent terms, and it can be directly used as an IMPSAT equation without any reduction. For the total hydrocarbon mass balance equation, if we ignore the dependency of hydrocarbon phase densities on component mole fractions, then it will only include



pressure and saturation dependent terms and can be directly used as another IMPSAT equation without any reduction.

IMPSAT needs three equations for three-phase flow, and we already have two. So we only need to use the reduction steps in Chapter 3 on the remaining  $n_c - 2$  hydrocarbon component mass balance equations to get one more IMPSAT equation, and the cost of this operation is about 3 times the cost of building an IMPES model. If we consider the dependency of hydrocarbon phase densities on component mole fractions, the reduction steps in Chapter 3 need to be performed on the remaining  $n_c - 1$  hydrocarbon component mass balance equations to get two more IMPSAT equations, and the cost is doubled.

## 4.2 IMPSAT Based AIM Model

For large and difficult problems, IMPES timestep size is too small to be practical, and even IMPSAT is not stable enough. While FIM is always stable, it is just too expensive, especially for problems with a large number of components. In such cases, we propose adding the IMPSAT model into the traditional AIM model (Forsyth and Sammon, 1986), we call the new AIM model the IMPSAT based AIM model. Compared to the traditional AIM (IMPES+FIM) model, it is more flexible, more stable and less expensive. There are three variations of the new IMPSAT based AIM model:

- **IMPES+IMPSAT** Here we use the IMPSAT model to replace the FIM model, and this model is the least expensive one in terms of cost per Newton iteration.
- **IMPSAT+FIM** Here we use the IMPSAT model to replace the IMPES model, and this model is the most stable one.
- **IMPES+IMPSAT+FIM** Here we use the IMPSAT model as the third formulation in the new AIM model, and this model is the most general one. Actually, the first two AIM models and the traditional AIM model are all special cases of this general AIM model.

All of these AIM models are very easy to build using the General Formulation Approach proposed in Chapter 3, where all of the operations are done gridblock by gridblock, and we can control the implicit level at each gridblock separately.

Depending on different types of flow problems, one of these new AIM models will outperform the traditional AIM model. For easy problems, IMPES+IMPSAT is the best to use, for hard problems, IMPSAT+FIM is a better choice. With properly tuned percentages of IMPES, IMPSAT and FIM gridblocks, IMPES+IMPSAT+FIM should be the most efficient model to use, especially for large field case studies.

# Chapter 5

## Model Validation and Performance

In this chapter, we will present two sets of results, the first set is used to validate individual models and techniques implemented in GPRS. The results from this set are compared with the results from existing simulators. The second set is used to evaluate the performance of different compositional models. Here we compare the performance of IMPSAT and IMPSAT based AIM models with the traditional models (IMPES, FIM and AIM).

Before we start showing the results from GPRS and comparing them with the results from other simulators, it is worthwhile to discuss the convergence controls used in GPRS. In GPRS, the convergence of Newton iterations is governed by both the maximum residual and the maximum variable change. At convergence, the following condition should be satisfied:

- Scaled mass balance equation residuals are less than 0.1. Equations are scaled by the accumulation terms (component mass dividing timestep size).
- Scaled phase equilibrium relation residuals are less than 0.02. These are scaled by the component fugacities in the gas phase.
- The maximum scaled pressure change (absolute pressure change divided by the average reservoir pressure) is less than 0.0001.
- The maximum absolute saturation change is less than 0.005.
- The maximum absolute component mole fraction change is less than 0.001.

The convergence of iterative linear solvers is governed by the normalized linear equation residual ( $\|B - AX\|_2 / \|B\|_2$ ). This limit is set to  $10^{-4}$ .

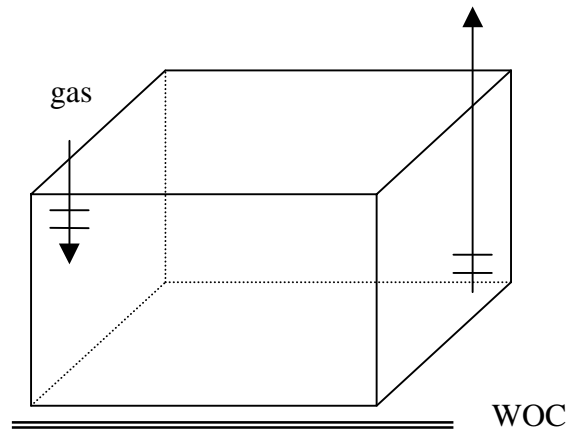
### 5.1 Validation of GPRS

Here we show validation results for the black-oil and the compositional models in GPRS. The black-oil model results are compared with the results from Eclipse 100 (Geoquest, 2000A), and the compositional model results are compared with the results from Eclipse 300 (Geoquest, 2000A). We have also tested the unstructured grid handling and multi-

point flux approximations in GPRS, and compared our results with the results from FLEX (Verma, 1996). Besides that, we will also show validation results for the General Formulation Approach (different variable selections and implicit levels). Finally, we include a non-conventional well problem, whose results are compared with the results from Eclipse 100.

### 5.1.1 Black-oil Model

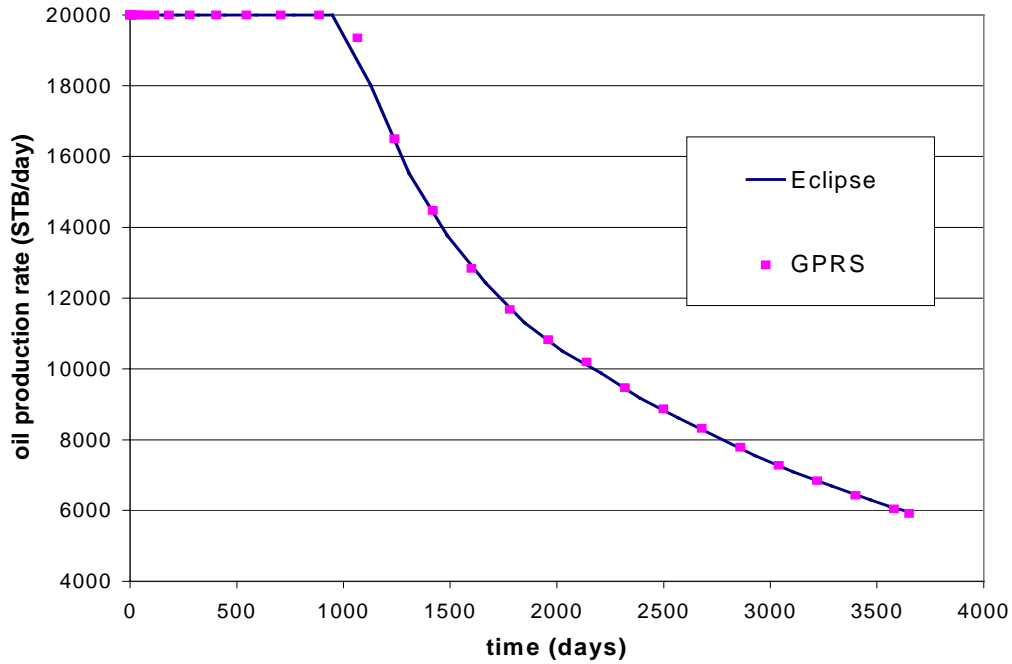
This is the first SPE comparative project (Odeh, 1981), a three-phase black-oil simulation of a quarter of a 5-spot. This grid is  $10 \times 10 \times 3$ . Each layer has different properties. One gas injector is located at (1, 1, 1), and one oil producer is located at (10, 10, 3), and both wells are under constant rate control, shown in Figure 5.1. Initially, the reservoir is full of undersaturated oil and connate water. The simulation was run to 10 years with a maximum timestep size of 180 days, and the results from GPRS are compared with the results from Eclipse 100. Compositional formulation was used to simulate this black-oil problem in GPRS, and both simulators were run in fully implicit mode. The gas was not allowed to re-dissolve.



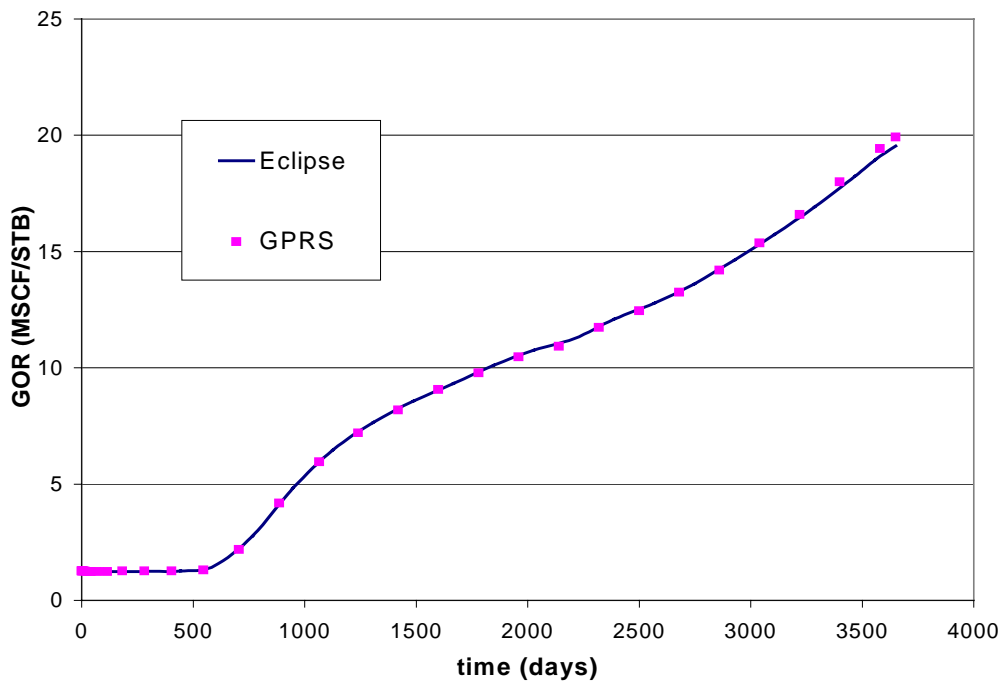
**Figure 5.1** Black-oil reservoir

Both simulators used the same number of timesteps (32) and Newton iterations (114). Figure 5.2 and Figure 5.3 show the results from GPRS and Eclipse 100. GPRS results are shown in dots, and Eclipse 100 results are shown in solid lines. Figure 5.2 shows the oil production rate, plateau production up until around 1000 days, then the producer changes to bottom hole pressure (BHP) control. Figure 5.3 shows the gas oil ratio (GOR) at the

producer, initially, there is no free gas, GOR is constant, after around 660 days, free gas appears, and GOR increases. Basically, the results from these two simulators are the same, which means that we have excellent agreement between GPRS and Eclipse 100.



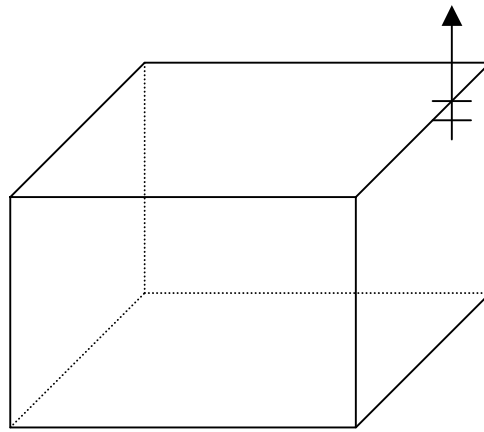
**Figure 5.2** Oil production rate at the producer from GPRS and Eclipse 100



**Figure 5.3** Gas oil ratio at the producer from GPRS and Eclipse 100

### 5.1.2 Compositional Model

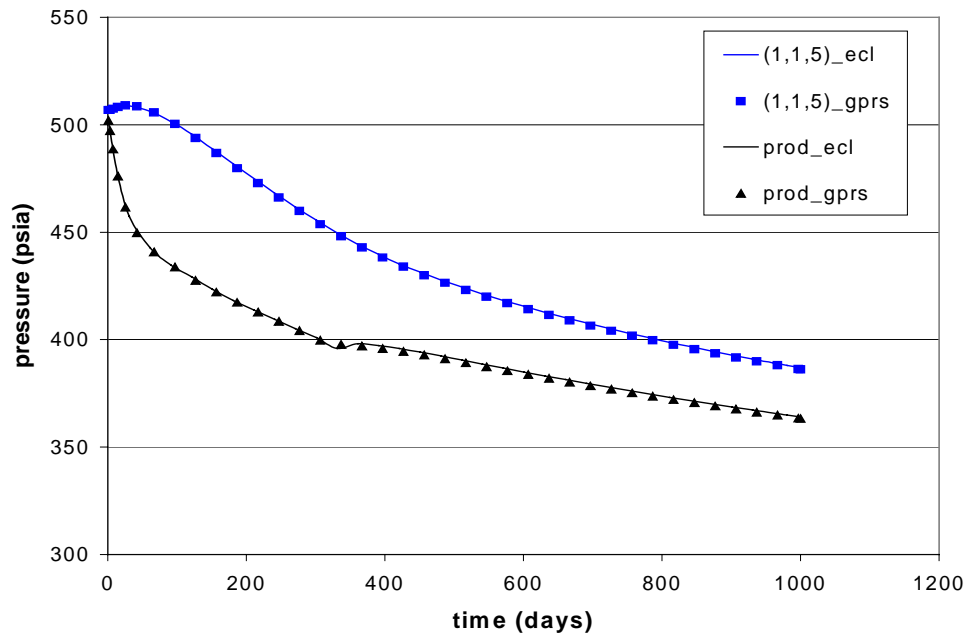
This is a four-component, two-phase (gas-oil) compositional simulation (Cao, 2002). The grid is  $5 \times 5 \times 5$ . One producer is located at (5, 5, 1), which is under bottom hole pressure (BHP) control (shown in Figure 5.4). The Peng-Robinson Equation of State (Peng and Robinson, 1976) was used for flash calculations. Initially the reservoir has both the oil and gas phases. The simulations were run to 1000 days with a maximum timestep size of 30 days, and the results from GPRS are compared with the results from Eclipse 300 (Geoquest, 2000A).



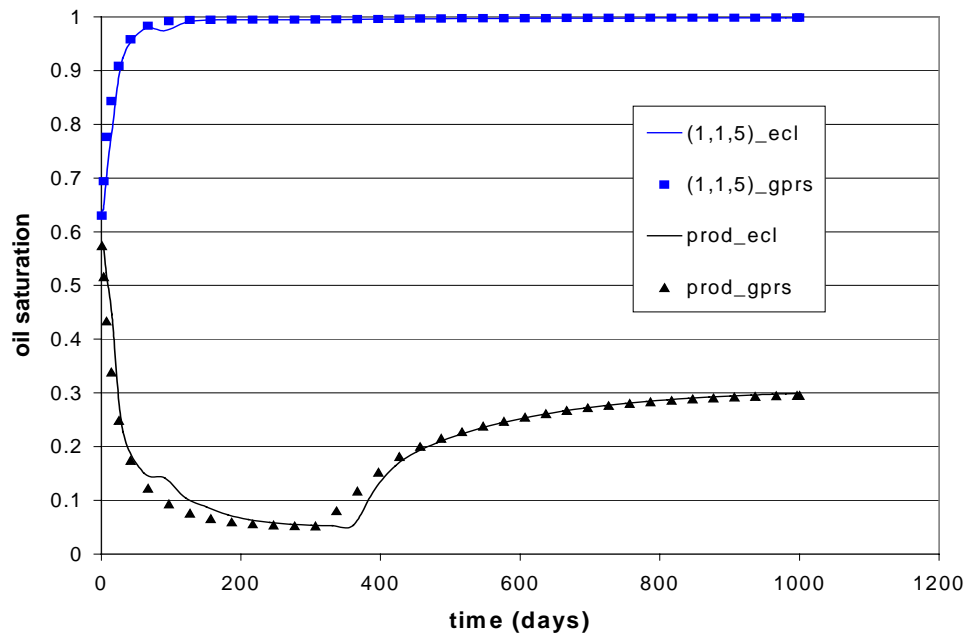
**Figure 5.4** Compositional reservoir

GPRS uses Type A variables, while Eclipse 300 uses Type B variables (pressure and overall molar density of each component). Both simulators were run in fully implicit mode. Both simulators quickly reached the maximum timestep size of 30 days, and used roughly the same number of timesteps (GPRS used 39 timesteps, while Eclipse 300 used 38 timesteps). GPRS took an average of 2.5 Newton iterations per timestep, while Eclipse 300 took an average of 2.1 Newton iterations per timestep. The slight increase of Newton iteration number in GPRS was caused by the difference in Newton iteration convergence control in the two simulators. In Eclipse 300, only the maximum change of pressure and effective saturations (the effective saturation change for a given molar density change) are used for convergence control (Geoquest, 2000A). Also their limits are larger than the limits used in GPRS, Eclipse 300 uses 0.1 atm for absolute pressure change and 0.01 for effective saturation change, while GPRS uses 0.0001 for maximum scaled pressure

change and 0.005 for absolute saturation change. Figure 5.5 and Figure 5.6 compare the results from GPRS with the results from Eclipse 300.

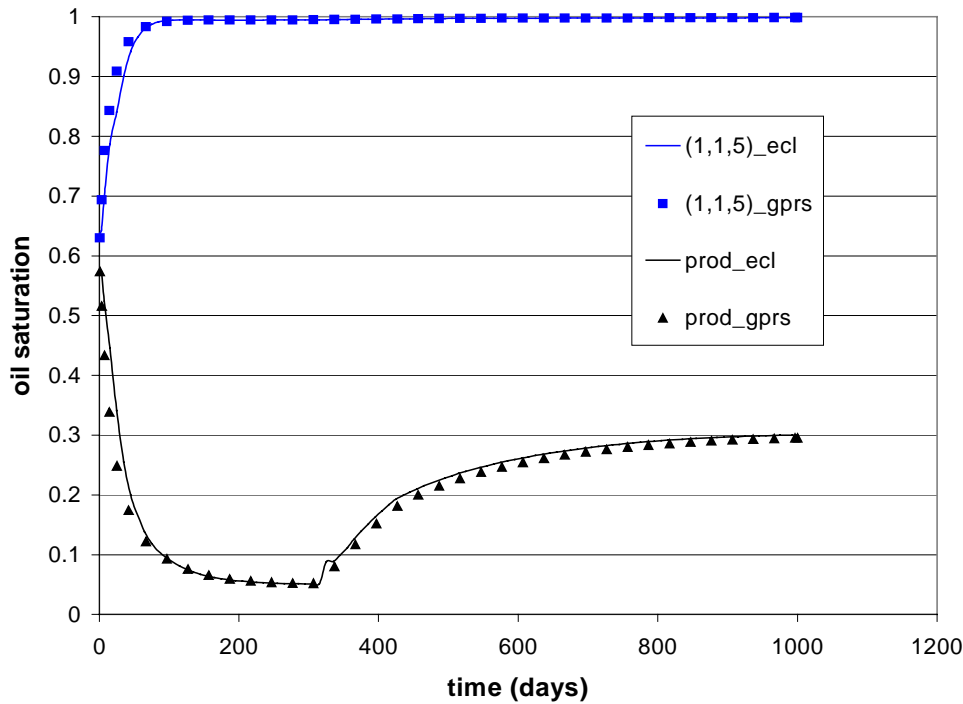


**Figure 5.5** Well block and block (1,1,5) pressures from GPRS and Eclipse 300 with  $\Delta t_{\max}=30$  days



**Figure 5.6** Well block and block (1,1,5) oil saturations from GPRS and Eclipse 300 with  $\Delta t_{\max}=30$  days

Similar to the results in Figure 5.2 and Figure 5.3, GPRS results are shown in dots, and Eclipse 300 results are shown in solid lines. The black lines are for producer (5,5,1), and the blue lines are for block (1,1,5). Figure 5.5 shows the pressure at block (1,1,5) and at the producer (5,5,1). Figure 5.6 shows the oil saturation at these two blocks. The oil saturation at the producer block from the two simulators does not match very well, the reason is that Eclipse 300's convergence criteria (pressure and effective saturation changes) are not as strict as those used in GPRS. Actually Eclipse 300 goes to the next timestep without full convergence in the current timestep. From Figure 5.6 we can notice a glitch (around 90 days) in Eclipse 300 results. By reducing the maximum timestep size from 30 days to 10 days in Eclipse 300, the results improve as shown in Figure 5.7. Eclipse 300 used a maximum timestep size of 10 days, and GPRS still used 30 days. Now these two results are almost on top of each other. Hence we can conclude that the compositional model in GPRS works well, and at times it performs even better than the model in Eclipse 300. From Figure 5.7, we notice that, even for 10 days, there is still a small glitch (around 330 days) in Eclipse 300 results, probably still due to the loose convergence control.

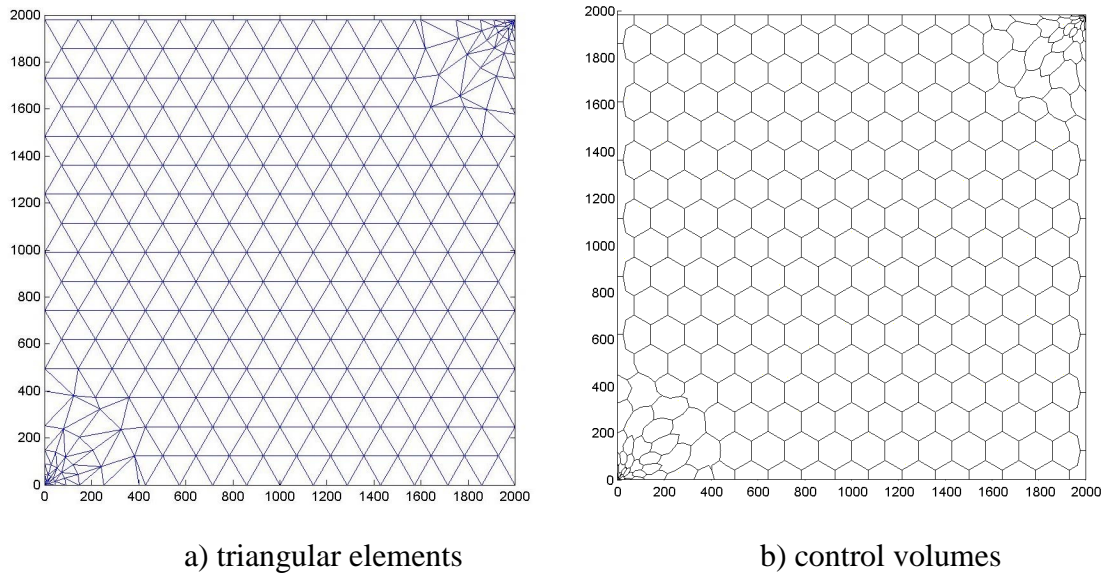


**Figure 5.7** Well block and block (1,1,5) oil saturations from GPRS and Eclipse 300 with  $\Delta t_{\max}=30$  days for GPRS and  $\Delta t_{\max}=10$  days for Eclipse 300



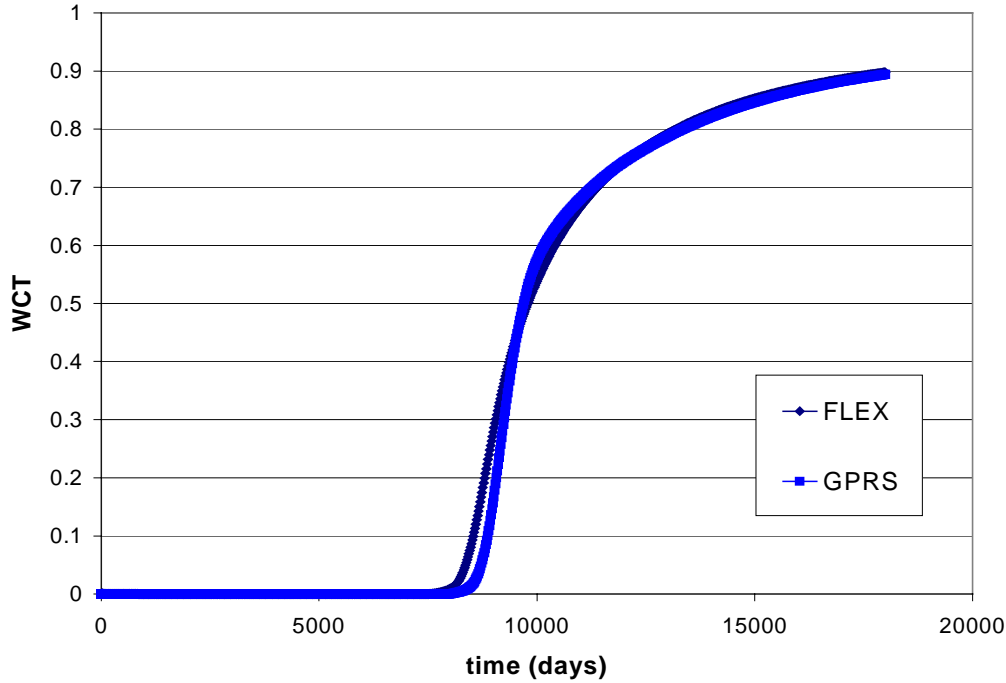
### 5.1.3 Unstructured Grid and Multi-Point Flux

A test problem pieced together from Verma (1996) was used to validate the implementation of unstructured grids and multi-point flux in GPRS. This is a quarter of a five spot simulation of a homogeneous black-oil (water-oil) reservoir (Cao, 2002), one water injector is located at the lower-left corner and one producer is located at the upper-right corner. Both wells are at constant rate control. The grid is a two-dimensional (x-y) triangular grid with local grid refinements around the wells, Figure 5.8a shows the triangular elements, and Figure 5.8b shows the control volumes (gridblocks). Most of the multi-point fluxes are around the wells, where the gridblocks are irregular. This problem has 301 gridblocks, 838 connections, and about 30% of the connections use multi-point flux calculations.



**Figure 5.8** Unstructured grid (Verma, 1996)

We ran this problem using GPRS and compared its results with the results from FLEX (Prevost, 2002). Figure 5.9 shows the water cut at the producer. Considering the differences in these two simulators, the results compare well. There is only a small discrepancy at the water breakthrough time. We can conclude that the unstructured grid handling and multi-point flux approximations in GPRS work well.

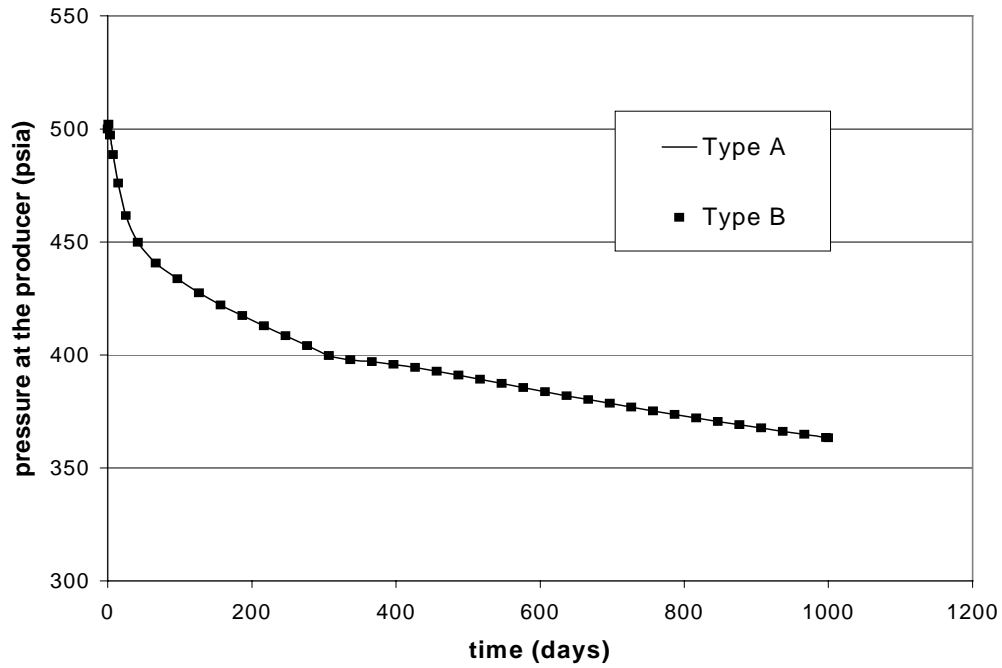


**Figure 5.9** Water cut results from GPRS and FLEX for unstructured grid

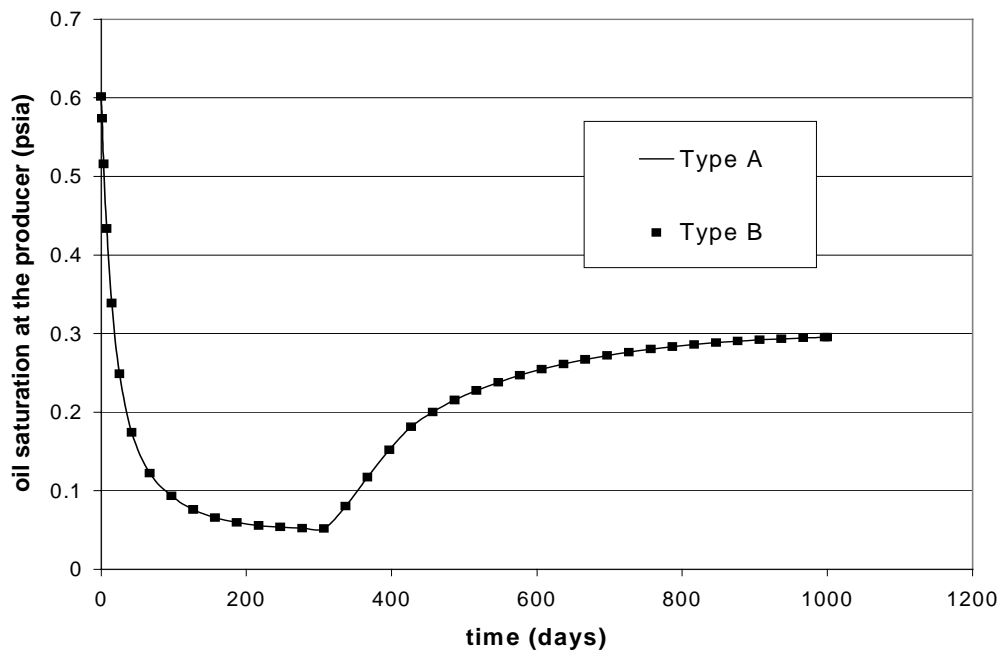
#### 5.1.4 General Formulation Approach

This is the same homogeneous  $5 \times 5 \times 5$  four component (C1, C2, C4, C7) two hydrocarbon phase problem that was used to validate the compositional modeling in Subsection 5.1.2. Here we compare the results of models with different variable selections and implicit levels. The maximum timestep size of 30 days was used for all runs. Figure 5.10 and Figure 5.11 compare the results of the FIM model with Type A and Type B variables. Type A results are shown in solid lines, and Type B results are shown in dots. Figure 5.10 shows the gridblock pressure at the producer and Figure 5.11 shows the oil saturation at the producer. The results from both FIM models match exactly, this is because they are solving the same equations and at convergence they should have the same solutions. Figure 5.12 and Figure 5.13 compare the results of models of Type A variables with different implicit levels, and they are the FIM model (all four primary variables are implicit), the IMPSAT++ model (the first three primary variables are implicit), the IMPSAT model (the first two primary variables are implicit) and the IMPES model (only pressure are implicit). Figure 5.12 shows the gridblock pressure at the producer and Figure 5.13 shows the oil saturation at the producer. From these results, we can see that

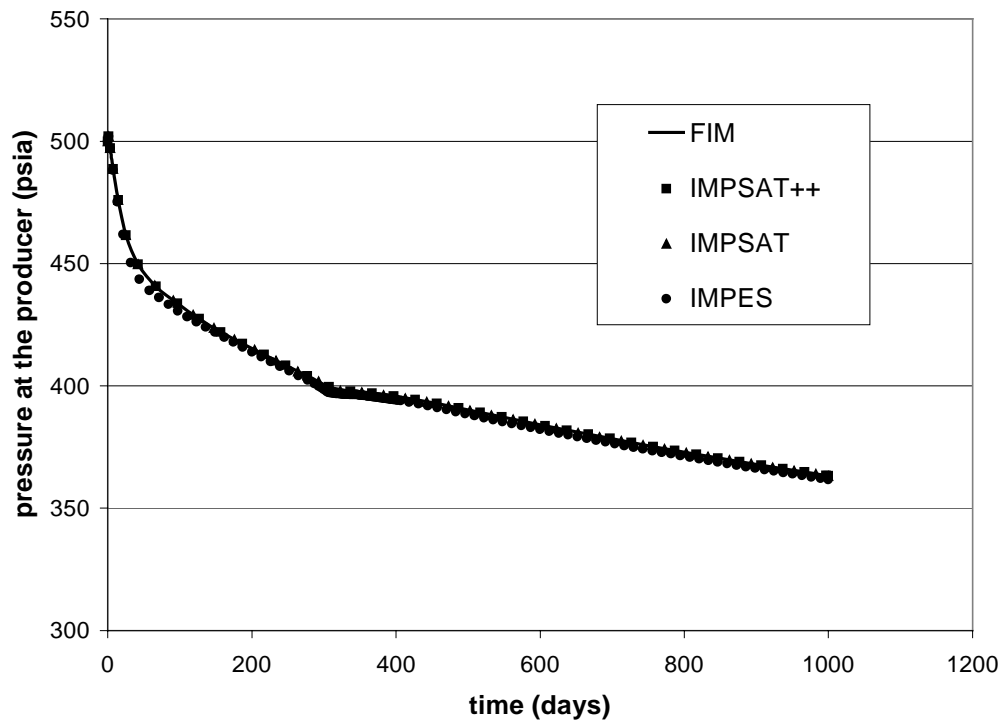
the IMPES results have less numerical diffusion, and the results from the other three models are basically the same.



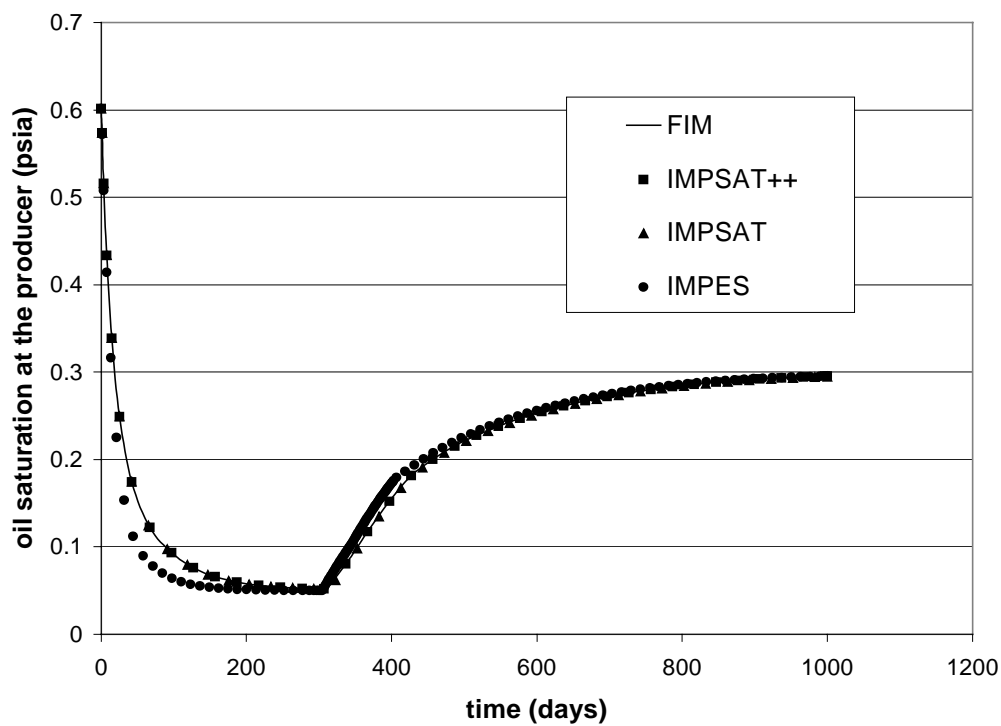
**Figure 5.10** Well block pressure for the FIM model with Type A and Type B variables



**Figure 5.11** Well block oil saturation for the FIM model with Type A and Type B variables



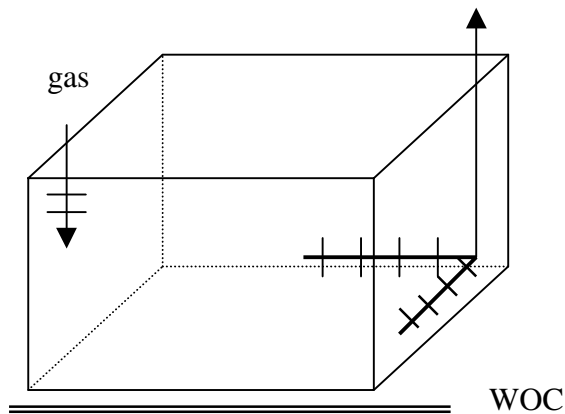
**Figure 5.12** Well block pressure for models with different implicit levels



**Figure 5.13** Well block oil saturation for models with different implicit levels

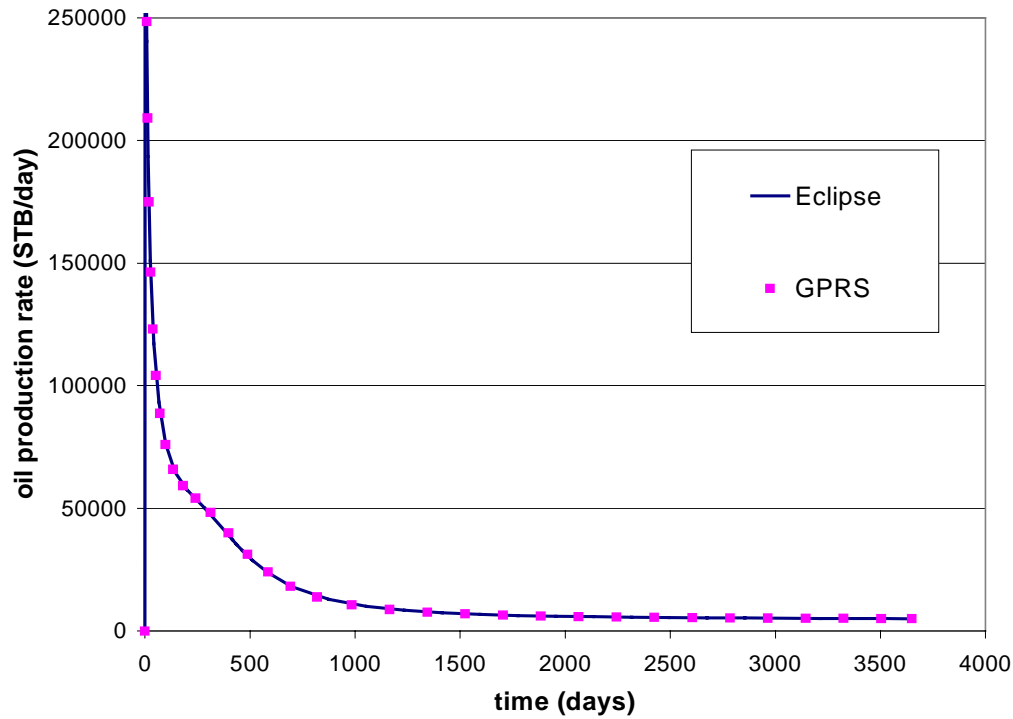
### 5.1.5 Non-conventional Well

This is the same  $10 \times 10 \times 3$  three-phase black-oil problem that was used to validate the black-oil modeling in Subsection 5.1.1. Here, the producer is replaced by a BHP controlled dual-lateral well, which is located at (6-10, 10, 3) and (10, 6-10, 3), shown in Figure 5.14. Initially, the reservoir is full of undersaturated oil and connate water. The simulation was run to 10 years with a maximum timestep size of 180 days, and the results from GPRS are compared with the results from Eclipse 100. Compositional formulation was used to simulate this black-oil problem in GPRS, and both simulators were run in fully implicit mode. The gas was not allowed to re-dissolve.

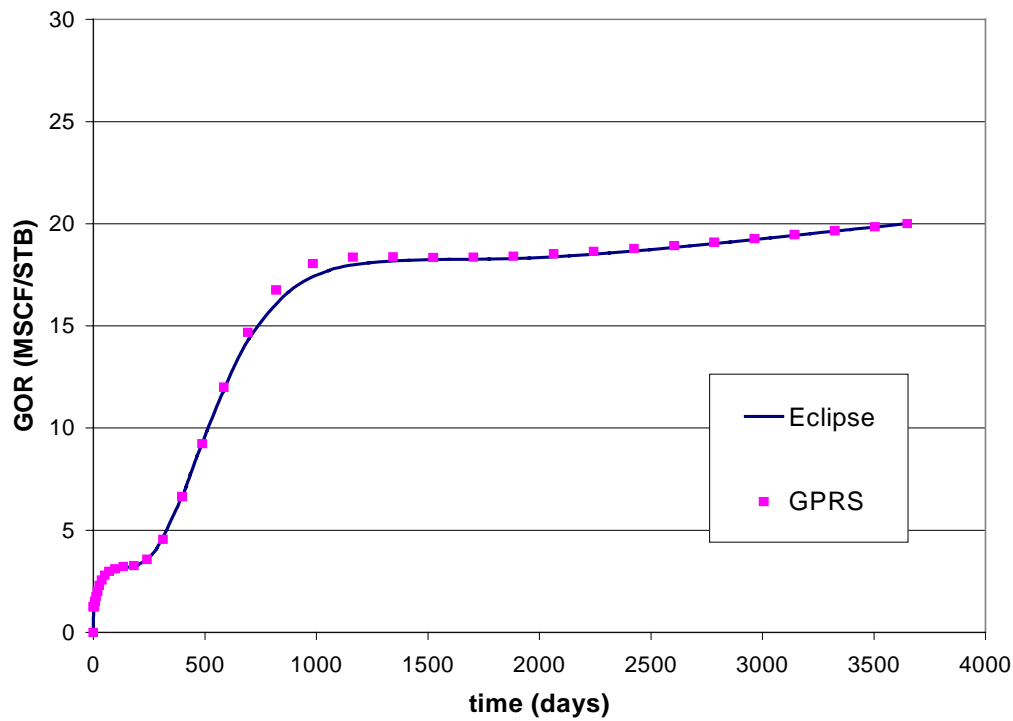


**Figure 5.14** Black-oil reservoir with a dual-lateral producer

Figure 5.15 and Figure 5.16 show the results from GPRS and Eclipse 100. GPRS results are shown in dots, and Eclipse 100 results are shown in solid lines. Figure 5.15 shows the oil production rate and Figure 5.16 shows the gas oil ratio (GOR) at the producer. The results from these two simulators are almost the same.



**Figure 5.15** Oil production rate at the producer from GPRS and Eclipse 100 for a dual-lateral well problem



**Figure 5.16** Gas oil ratio at the producer from GPRS and Eclipse 100 for a dual-lateral well problem

So far, we have shown that the basic modules of GPRS work well, we can now start to evaluate the performance of different compositional models using various compositional problems. The performance of individual compositional models is independent of the grid type and the method of flux approximation, so in this part we will use Cartesian grids and two-point flux approximations for all problems.

## 5.2 Performance of Compositional Models in GPRS

A series of compositional problems were used to evaluate the performance of different compositional models. The size of these problems ranges from 125 gridblocks to 100,000 gridblocks, number of components ranges from 4 to 9, with both homogeneous and highly heterogeneous domains. The compositional models evaluated here include fully implicit (FIM) models with both types of variables (Type A and Type B), IMPES models with both types of variables, the IMPSAT model with Type A variables and the traditional and new AIM models. Besides the simulation results, we also report the iteration counts (number of timesteps, number of total Newton iterations and number of total linear solver iterations) and the timing results (solver time and total running time) for each problem and each model. All of the simulations were performed on a SGI origin 200 workstation.

Here both Type A and Type B variables were used. For a four hydrocarbon component and two hydrocarbon phase problem, the full set of equations (total 8) include the mass balance equations and the phase equilibrium relations for each hydrocarbon component, with the four mass balance equations as the primary equations. For Type A variables, the full set of variables is  $p_g, S_g, y_1, y_2, y_3, x_1, x_2, x_3$ , where  $y_i$  and  $x_i$  are the component mole fractions in the gas and oil phases, and  $p_g, S_g, y_1, y_2$  are the primary variables. For Type B variables, the full set of variables is  $p_g, r_1, r_2, r_3, S_g, y_1, y_2, y_3$ , where  $r_i$  are the overall densities of hydrocarbon components, and  $p_g, r_1, r_2, r_3$  are the primary variables.

In this section, we will first introduce compositional problems used, then evaluate the performance of individual compositional models. The performance of compositional models is evaluated under different conditions. Each problem is designed for one specific condition. These conditions include injection of components, heterogeneity, large number

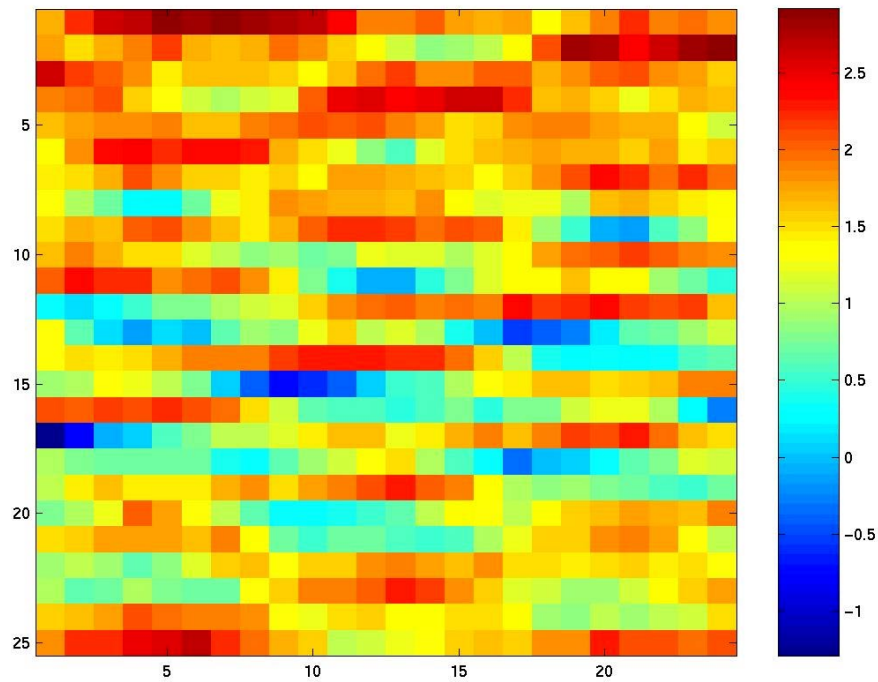
of components and large systems. The five compositional problems used here are discussed below (Cao, 2002):

- **Case 1** This is the same homogeneous  $5 \times 5 \times 5$  four component (C1, C2, C4, C7) two hydrocarbon phase problem that was used to validate the compositional modeling in GPRS. One bottom hole pressure (BHP) controlled producer is located at (5, 5, 1). This is a relatively easy problem, and both IMPES and IMPSAT should perform well.
- **Case 2** This is the same problem as in Case 1, but here an injector is added to increase the difficulty. The injector is located at (1, 1, 1). It is under bottom hole pressure control and C1 is injected. This problem is difficult for the IMPSAT model, because a component is injected when the component mole fractions are treated explicitly.
- **Case 3** This is a heterogeneous problem, with four components (C1, C2, C4, C7) and two hydrocarbon phases. The grid is  $24 \times 1 \times 25$ . One bottom hole pressure controlled producer is located at (1, 1, 1-3), and it penetrates only the top three layers. The permeability field is from the 9<sup>th</sup> SPE comparative project (Killough, 1995). Only the top layer, shown in Figure 5.17, is used. The permeability changes over four orders of magnitude and it is correlated along the x direction. The primary objective here is to test the performance of different models on a difficult heterogeneous problem. Due to high flow rates in the high permeability regions, this problem is hard for both IMPES and IMPSAT.
- **Case 4** This is a nine-component problem. The fluid description is from the 3<sup>rd</sup> SPE comparative project (Kenyon and Behie, 1987), and the nine components are CO<sub>2</sub>, N<sub>2</sub>, C1, C2, C3, C4-6, C7<sub>+1</sub>, C7<sub>+2</sub> and C7<sub>+3</sub>. The grid is  $5 \times 5 \times 5$  with homogeneous permeability, and one bottom hole pressure controlled producer is located at (5, 5, 1). Due to the large number of components, this problem is hard for both IMPES and IMPSAT.
- **Case 5** This is a large homogeneous problem with 100,000 gridblocks ( $100 \times 100 \times 10$ ) and four components (C1, C2, C4, C7). One bottom hole pressure controlled producer is located at (100, 100, 1-2), and it penetrates only the top two layers. Compared to other problems, this problem has much smaller gridblocks ( $100\text{ft} \times 100\text{ft} \times 10\text{ft}$  compared to  $1000\text{ft} \times 1000\text{ft} \times 10\text{ft}$  for other problems). This small gridblock size limits the maximum stable timestep size of IMPSAT to around 0.5 days, and it is virtually



impossible to run the problem with this model. The stability restriction is even worse for IMPES models. So for this problem, only the FIM and AIM models were used. The stability of the IMPES+IMPSAT model is even worse than the stability of the IMPSAT model, so it was also not used.

All of the simulations were run to 1000 days, and the Peng-Robinson EOS (Peng and Robinson, 1976) was used for flash calculations. Now that we have introduced all of the problems, next we will show the performance of GPRS for each type of compositional model, starting from the FIM model.



**Figure 5.17** Heterogeneous permeability field ( $\ln k$ ) for Case 3

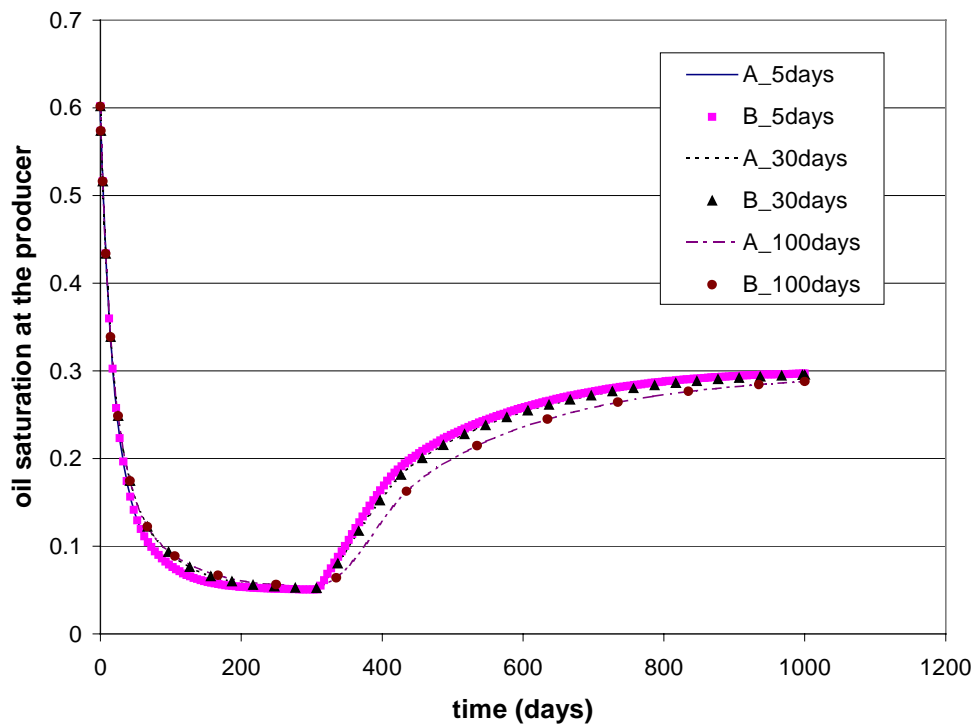
### 5.2.1 Fully Implicit Models

The FIM model with both Type A and Type B variables was used. Model performance is compared for different problems and different timestep sizes. The maximum timestep sizes tested were 5, 30 and 100 days (For Case 5, only timestep sizes of 30 and 100 days were used). Figure 5.18 to Figure 5.22 show the well block oil saturation for each

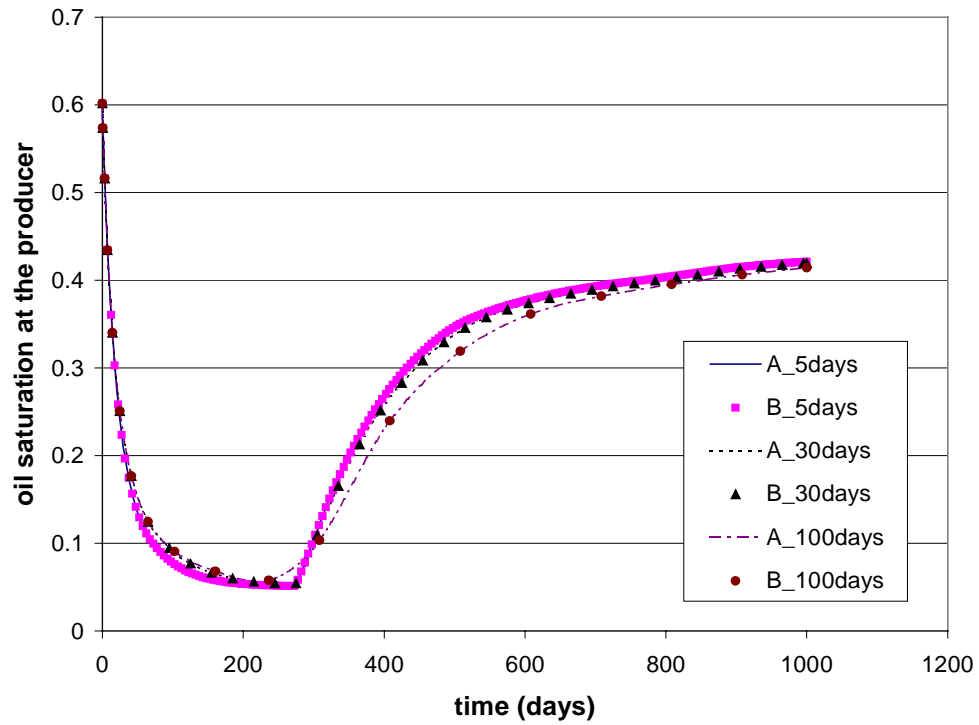
problem, and the legend in each figure marks the model used, such as B\_30days stands for Type B FIM model with a maximum timestep size of 30 days. Type A variable results are drawn in lines, and Type B variable results are drawn in dots. From these results, we can draw the following conclusions:

- For the same maximum timestep size, both FIM models match exactly, this is because they are solving the same equations and at convergence they should have the same solutions.
- With the increase of timestep size, there is more numerical diffusion for both FIM models and the results are more smoothed out.
- Both FIM models are always stable.

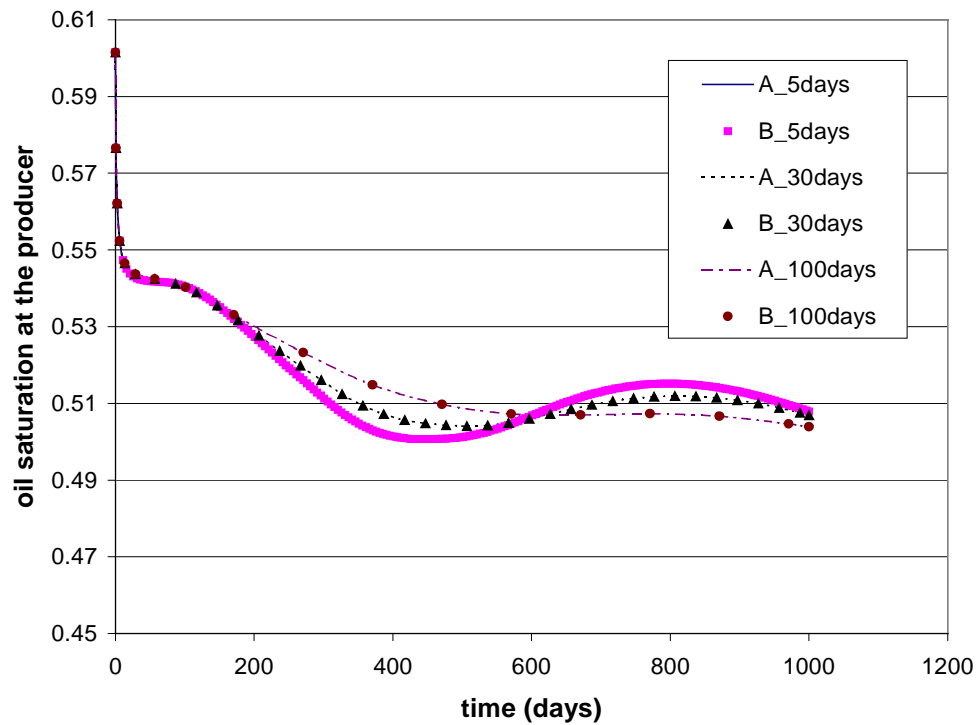
These three observations are true for all compositional problems.



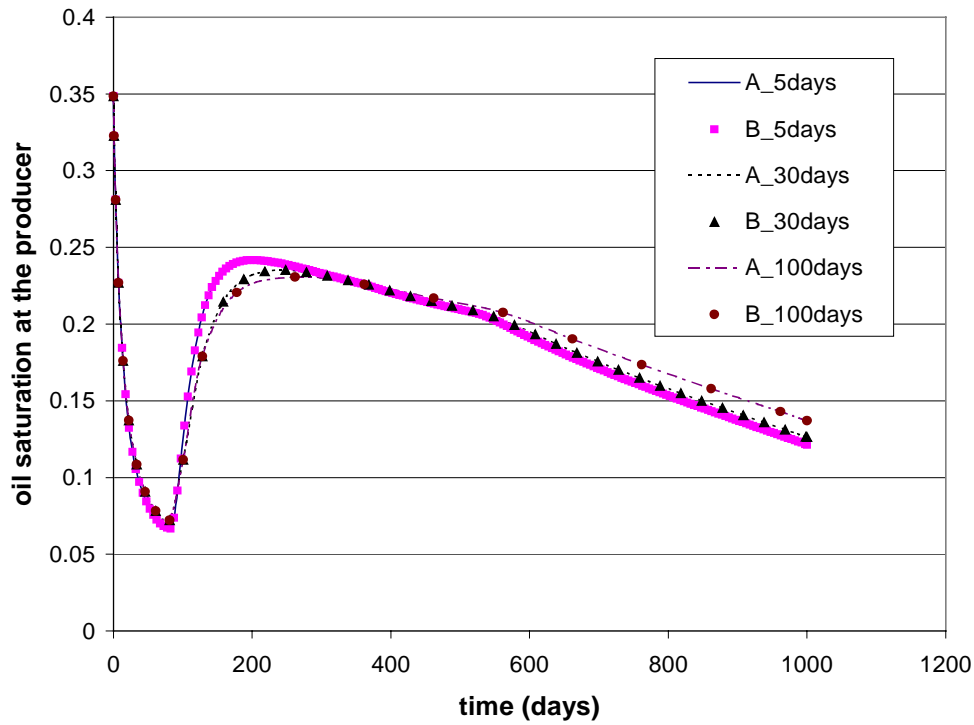
**Figure 5.18** Well block oil saturation comparisons for FIM models and Case 1 problem



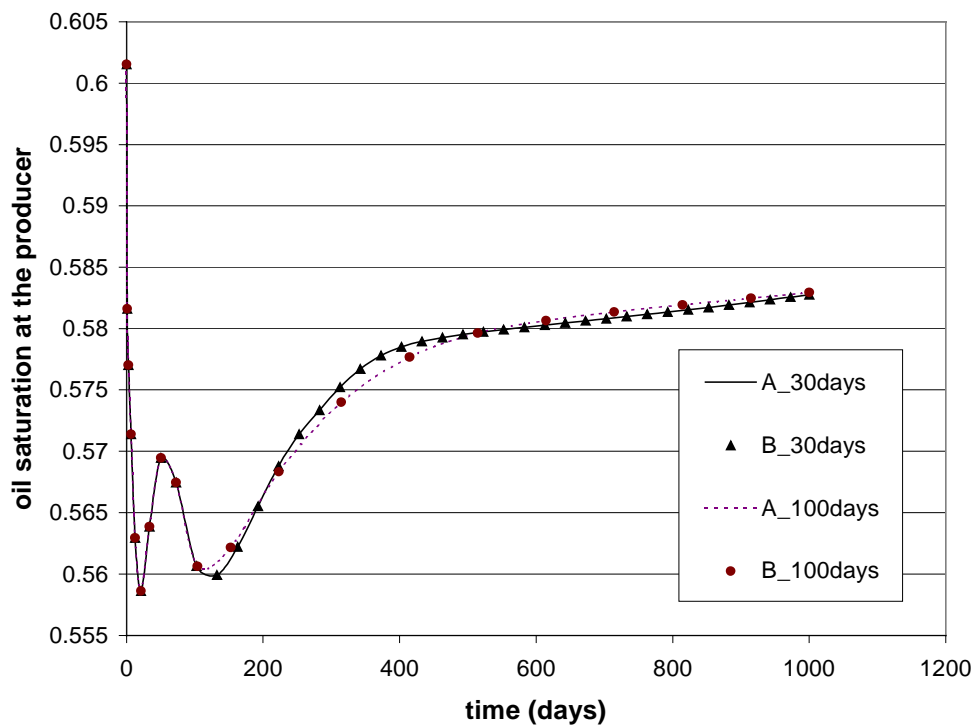
**Figure 5.19** Well block oil saturation comparisons for FIM models and Case 2 problem



**Figure 5.20** Well block oil saturation comparisons for FIM models and Case 3 problem



**Figure 5.21** Well block oil saturation comparisons for FIM models and Case 4 problem



**Figure 5.22** Well block oil saturation comparisons for FIM models and Case 5 problem

Iteration counts and timing results for problems discussed above are presented in Table 5.1 to Table 5.5. For all of the fully implicit runs, the CPR preconditioned GMRES method was used to solve the linear system. We can draw the following conclusions based on our analysis:

- For both FIM models, with the increase of timestep size, more Newton iterations are needed at each timestep and more linear solver iterations are needed to solve each linear system. This is because the system becomes less diagonally dominant as the timestep size increases.
- There is no significant difference in the number of Newton iterations between these two FIM models.
- Type B FIM model generally takes more linear solver iterations than Type A FIM model. As a result, Type B FIM model takes more time on the linear solver, and in some cases, the increase in CPU time can be as much as 30%. We also tried other solvers and preconditioners, such as ILU0 and BlitzPak, and we observe the same trend for all of them.
- Type B FIM model is also more costly in building the Jacobian matrix, since it needs to use the chain rule to calculate the derivatives. With the higher solver cost, the total cost is also substantially higher.

In summary, Type A FIM model costs less than Type B FIM model, at least for the problems tested here. The implementations of Type A models are more efficient than the implementations of Type B models in GPRS because of the variable switching, so we can only compare the relative cost of these two types of models. This is true for all of the other comparisons between models using these two types of variables.

	A_5days	B_5days	A_30days	B_30days	A_100days	B_100days
$N_{timesteps}$	202	202	39	39	18	18
$N_{Newton\ iters}$	420	420	98	98	56	56
$N_{solver\ iters}$	728	1135	229	363	143	221
$T_{solver}(sec)$	10.52	11.83	2.66	3.15	1.58	1.86
$T_{total}(sec)$	23.56	31.44	5.83	7.88	3.38	4.57

**Table 5.1** Iteration and timing comparisons for FIM models and Case 1 problem

	A_5days	B_5days	A_30days	B_30days	A_100days	B_100days
$N_{timesteps}$	202	202	39	39	18	18
$N_{Newton\ iters}$	428	429	109	107	63	63
$N_{solver\ iters}$	945	1538	300	493	187	298
$T_{solver}(sec)$	11.66	14.2	3.73	4.84	2.4	3.36
$T_{total}(sec)$	25.12	33.78	7.35	9.81	4.6	6.39

**Table 5.2** Iteration and timing comparisons for FIM models and Case 2 problem

	A_5days	B_5days	A_30days	B_30days	A_100days	B_100days
$N_{timesteps}$	202	202	38	38	17	17
$N_{Newton\ iters}$	413	413	99	99	58	58
$N_{solver\ iters}$	826	1647	248	483	153	272
$T_{solver}(sec)$	42.41	53.53	11.02	14.64	6.69	8.87
$T_{total}(sec)$	98.26	138.65	24.76	35.27	14.96	21.39

**Table 5.3** Iteration and timing comparisons for FIM models and Case 3 problem

	A_5days	B_5days	A_30days	B_30days	A_100days	B_100days
$N_{timesteps}$	202	22	41	41	21	21
$N_{Newton\ iters}$	417	417	110	110	64	65
$N_{solver\ iters}$	1012	1197	349	410	213	256
$T_{solver}(sec)$	77.9	80.16	22.23	22.96	13.04	14.03
$T_{total}(sec)$	128.44	174.81	36.26	48.65	21.76	29.64

**Table 5.4** Iteration and timing comparisons for FIM models and Case 4 problem

	A_5days	B_5days	A_30days	B_30days	A_100days	B_100days
$N_{timesteps}$	X	X	39	39	19	19
$N_{Newton\ iters}$	X	X	115	115	79	79
$N_{solver\ iters}$	X	X	483	638	379	540
$T_{solver}(sec)$	X	X	9050	18490	6782	13490
$T_{total}(sec)$	X	X	12680	24321	9292	17192

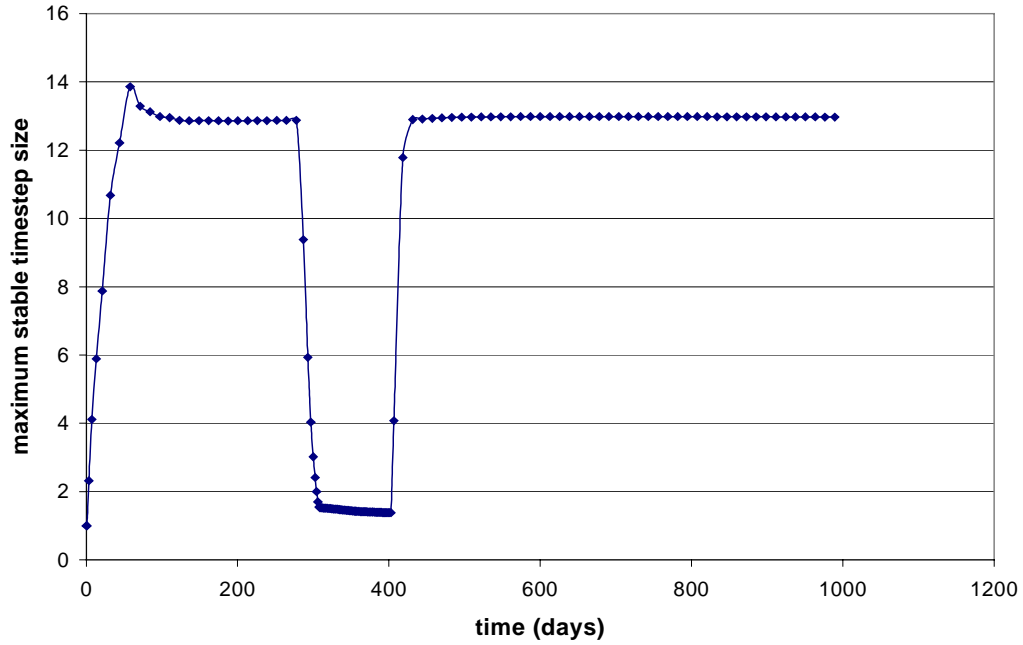
**Table 5.5** Iteration and timing comparisons for FIM models and Case 5 problem

### 5.2.2 IMPES models

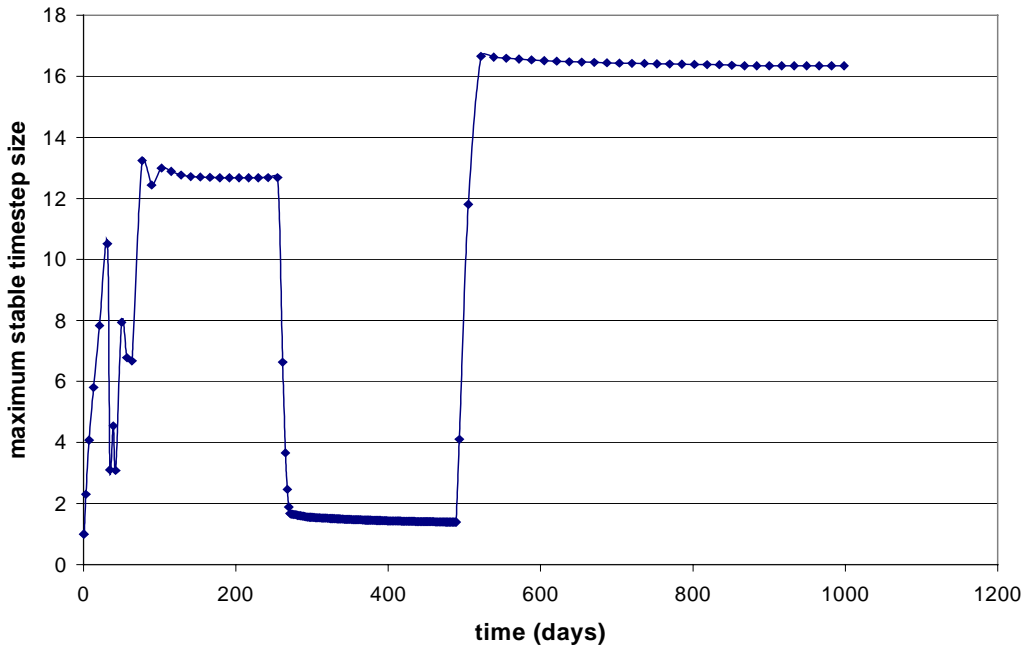
IMPES models with both Type A and Type B variables were used. Model performance is compared for different problems and different CFL numbers (larger CFL numbers lead to larger timestep sizes). Both CFL=1 and CFL=2 were used, CFL=1 is always stable, but CFL=2 may not be always stable. Figure 5.23 to Figure 5.26 show the stable timestep size (CFL=1) for each problem. Figure 5.27 to Figure 5.30 show the well block oil saturation for each problem, and the legend in each figure marks the model used, such as B\_CFL=2 stands for Type B IMPES model with a maximum CFL number of 2. Type A variable results are drawn in lines, and Type B variable results are drawn in dots. The fully implicit result (the dark solid line) for maximum timestep size of 30 days is included for comparison. From these results, we can draw the following conclusions:

- For all compositional problems tested here, the stable time step sizes of the IMPES model are small, ranging from 1.5 days to 16 days. This is because in compositional simulations, gas phase is always present, and gas flow is hard to handle for the IMPES model due to its high mobility.
- The stable timestep sizes of the IMPES model have sudden jumps. This is because the CFL limit of IMPES depends on the fractional flow derivative with respect to the saturation (saturation velocity), and at the saturation front, it is discontinuous, which causes the sudden jump.
- For the same timestep size, both IMPES models match exactly, this is because they are solving the same IMPES equations and at convergence they should have the same solution.
- Compared to the FIM model, the IMPES model has less numerical diffusion (shaper fronts), due to the explicit treatment of transmissibilities.

- For Case 1, 2 and 3, IMPES with CFL=2 is stable or nearly stable. But with the increase of number of components, such as in Case 4, this limit is reduced, and CFL=1 or 1.5 is a safe limit.

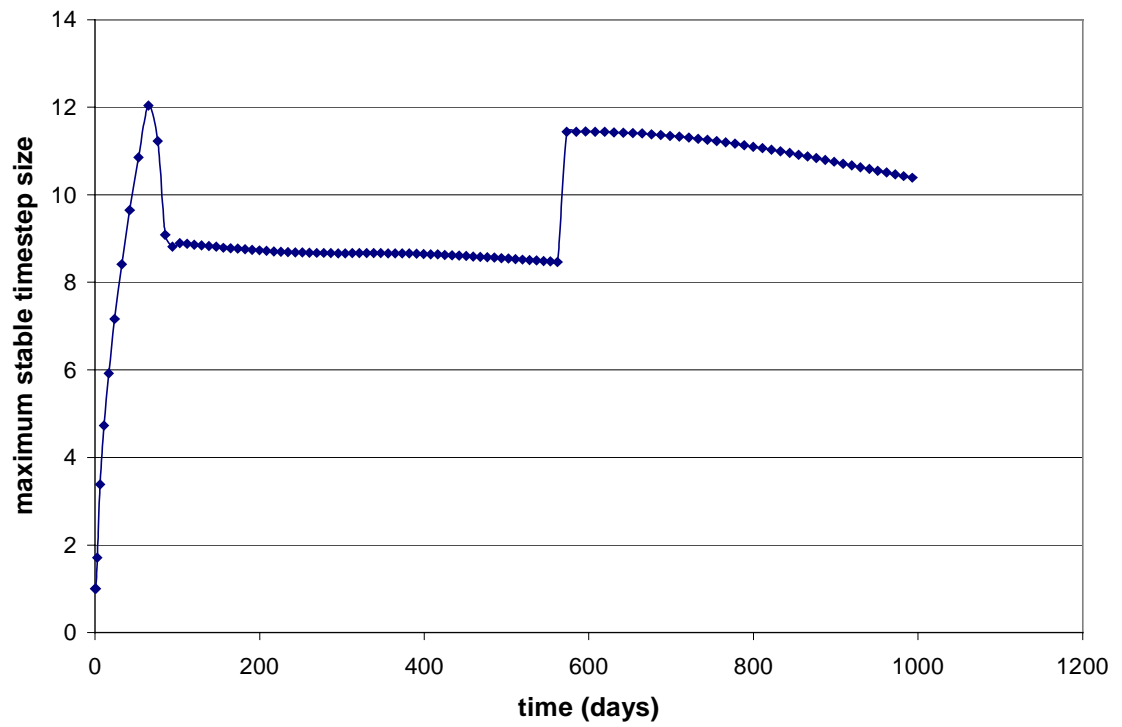


**Figure 5.23** Stable timestep size for the IMPES model and Case 1 problem

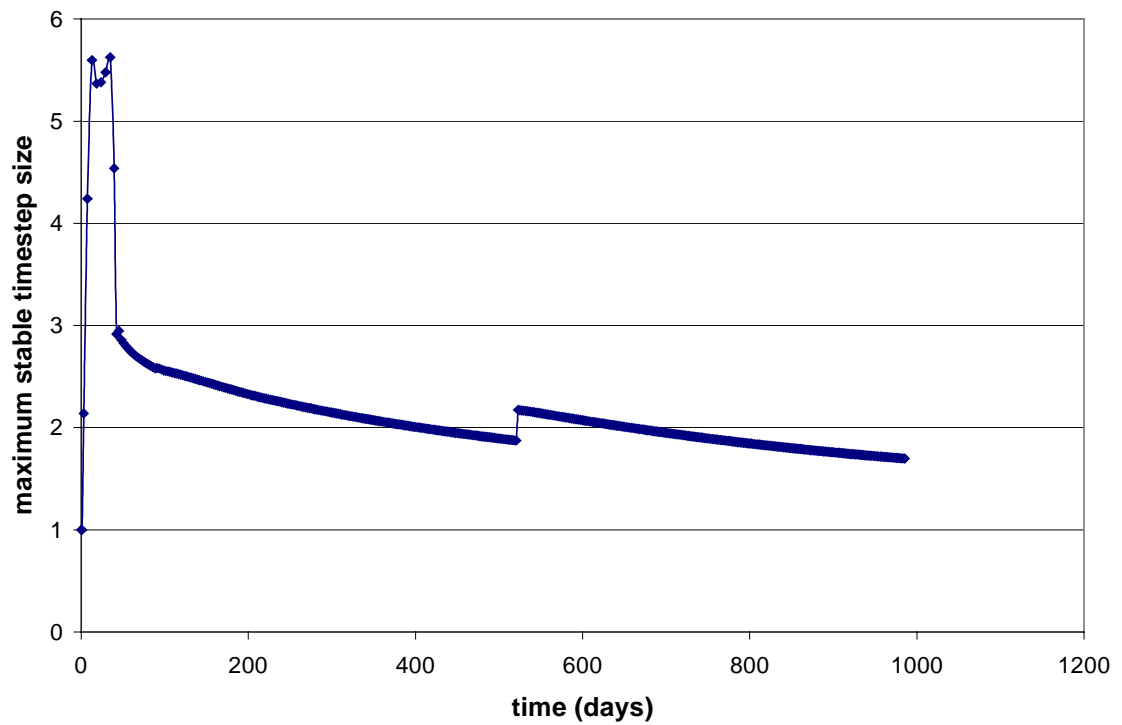


**Figure 5.24** Stable timestep size for the IMPES model and Case 2 problem

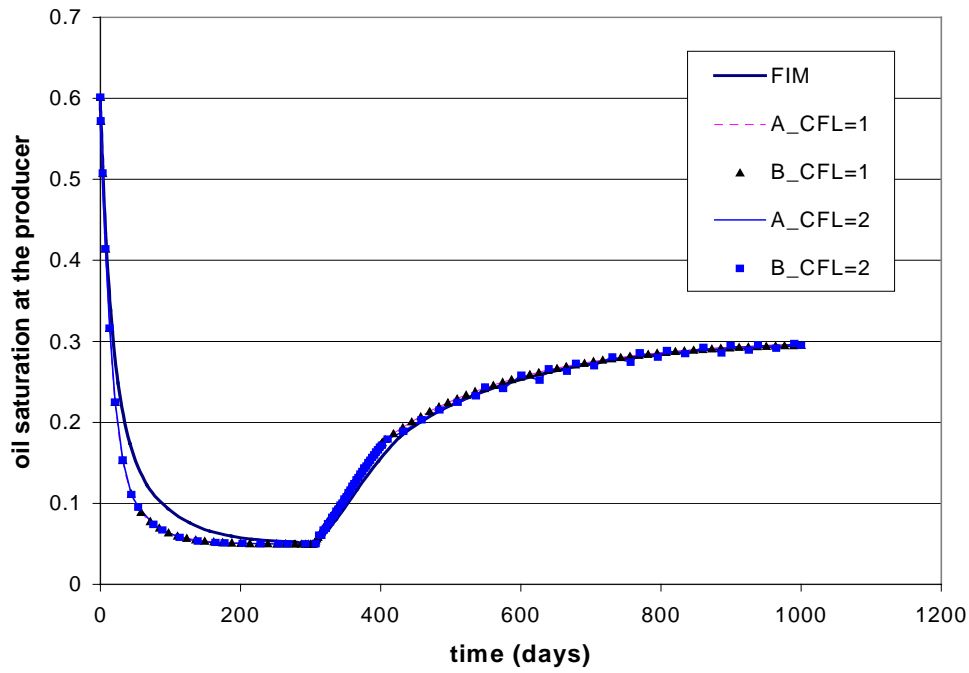




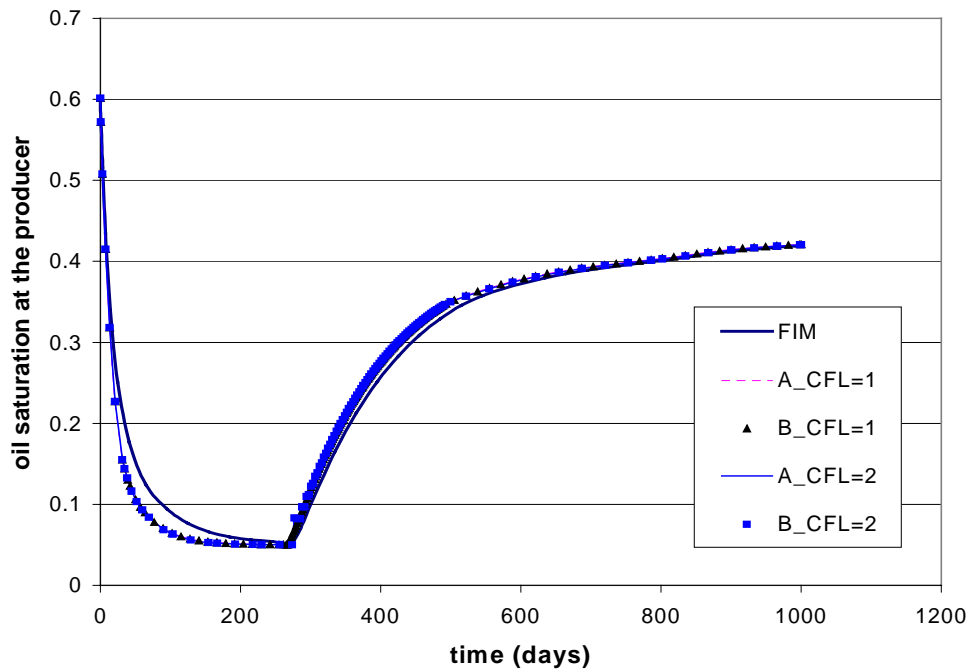
**Figure 5.25** Stable timestep size for the IMPES model and Case 3 problem



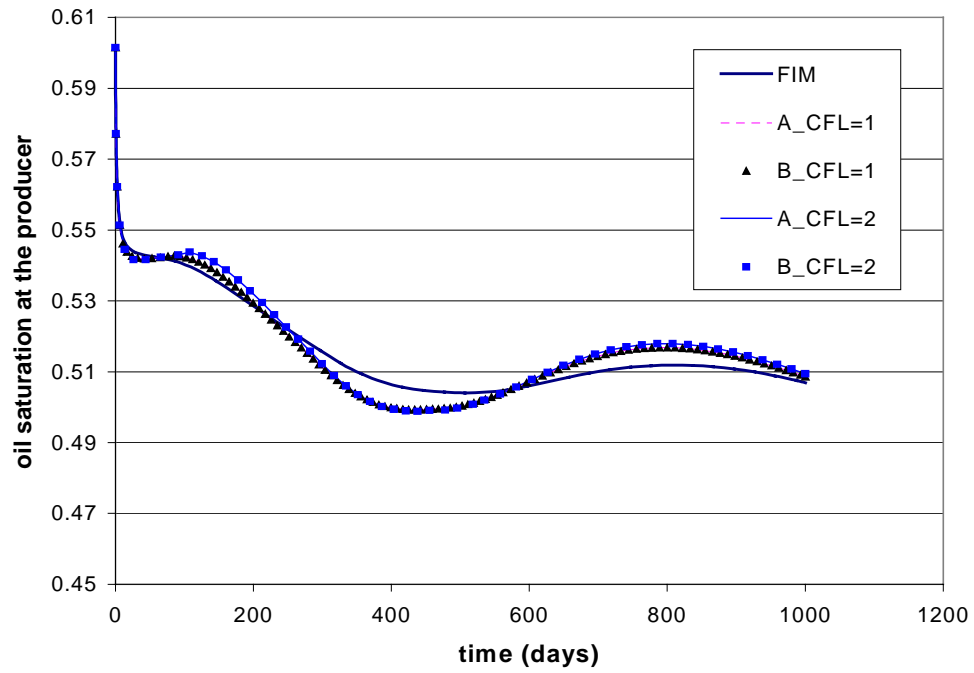
**Figure 5.26** Stable timestep size for the IMPES model and Case 4 problem



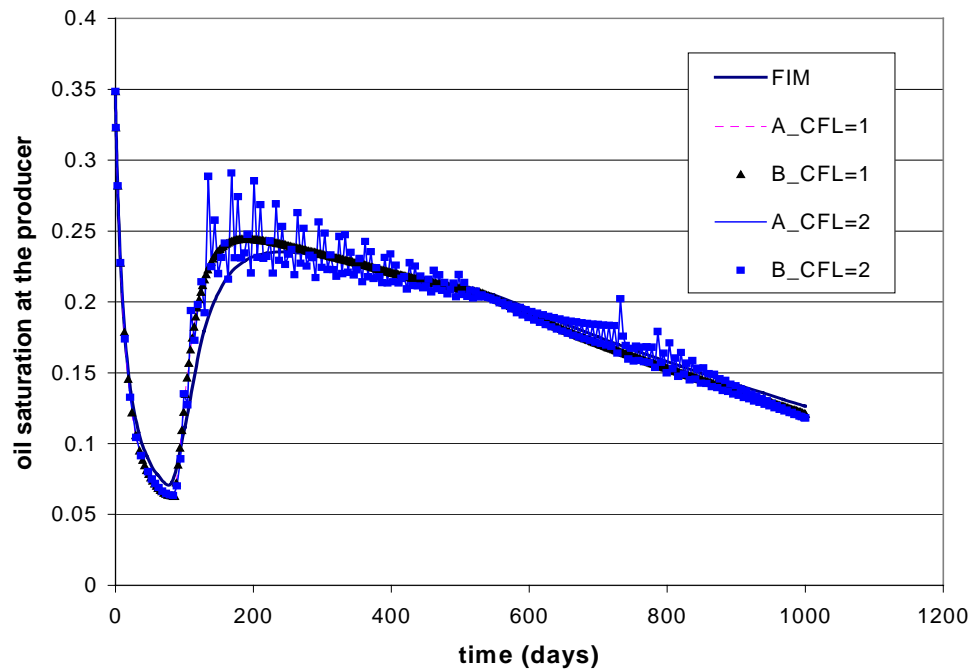
**Figure 5.27** Well block oil saturation comparisons for IMPES models and Case 1 problem



**Figure 5.28** Well block oil saturation comparisons for IMPES models and Case 2 problem



**Figure 5.29** Well block oil saturation comparisons for IMPES models and Case 3 problem



**Figure 5.30** Well block oil saturation comparisons for IMPES models and Case 4 problem

Iteration counts and timing results for problems discussed above are presented in Table 5.6 to Table 5.9. For all of the IMPES runs, the AMG preconditioned GMRES method was used to solve the linear system. We can draw the following conclusions based on our analysis:

- For both IMPES models, with the increase of CFL number (timestep size), more Newton iterations are needed at each timestep and more linear solver iterations are needed to solve each linear system. This is because the pressure system becomes less diagonally dominant as the timestep size increases.
- There is no significant difference in Newton iteration numbers between these two IMPES models.
- Number of linear solver iterations is also about the same for these two IMPES models. This is because both IMPES models generate the same pressure system, and the scalar difference between them is not enough to make any difference in linear solver performance.
- Type B models are built from Type A models by variable switching in GPRS. This increases the cost for Jacobian calculations in Type B models, as shown in the tables. But in practice, Type B IMPES model can be built directly from the mass balance equations. In this case Type B model will require fewer operations than Type A model to form the pressure system (Coats, 1999). Hence Type B IMPES model should be more efficient from the point of view of Jacobian matrix construction.

In summary, while our results don't show this, Type B IMPES model uses less time than Type A IMPES model, due to lower cost in Jacobian matrix construction.

	A_CFL=1	B_CFL=1	A_CFL=2	B_CFL=2
$N_{timesteps}$	146	146	80	80
$N_{Newton\_iters}$	301	301	174	174
$N_{solver\_iters}$	597	598	343	344
$T_{solver} (sec)$	0.89	0.89	0.83	0.83
$T_{total} (sec)$	8.98	13.66	7.82	11.94

**Table 5.6** Iteration and timing comparisons for IMPES models and Case 1 problem

	A_CFL=1	B_CFL=1	A_CFL=2	B_CFL=2
$N_{timesteps}$	214	214	114	114
$N_{Newton\_iters}$	442	440	249	246
$N_{solver\_iters}$	2995	2974	2008	1989
$T_{solver}$ (sec)	0.95	0.95	0.78	0.77
$T_{total}$ (sec)	14.9	20.53	10.99	15.99

**Table 5.7** Iteration and timing comparisons for IMPES models and Case 2 problem

	A_CFL=1	B_CFL=1	A_CFL=2	B_CFL=2
$N_{timesteps}$	107	107	54	54
$N_{Newton\_iters}$	218	218	112	112
$N_{solver\_iters}$	436	436	224	224
$T_{solver}$ (sec)	3.21	3.2	1.6	1.6
$T_{total}$ (sec)	29.49	44.53	15.65	15.65

**Table 5.8** Iteration and timing comparisons for IMPES models and Case 3 problem

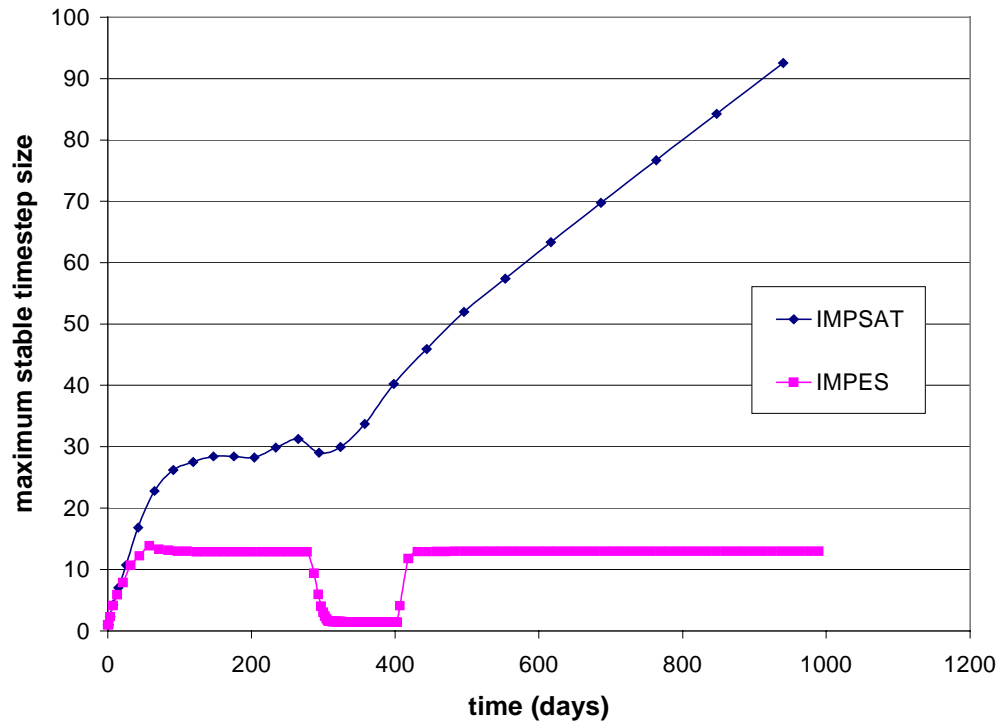
	A_CFL=1	B_CFL=1	A_CFL=2	B_CFL=2
$N_{timesteps}$	482	482	237	237
$N_{Newton\_iters}$	991	991	672	671
$N_{solver\_iters}$	1979	1980	1342	1341
$T_{solver}$ (sec)	3.62	3.82	2.17	2.55
$T_{total}$ (sec)	105.11	209.61	74.52	149.63

**Table 5.9** Iteration and timing comparisons for IMPES models and Case 4 problem

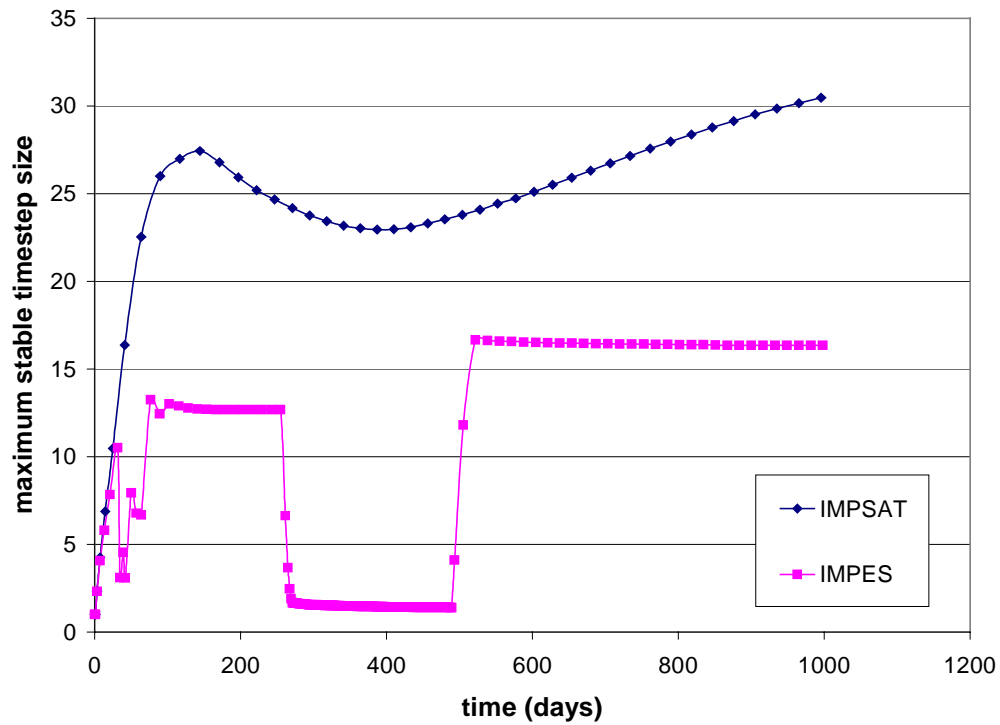
### 5.2.3 IMPSAT model

The performance of the IMPSAT (implicit pressure and implicit saturations) model is evaluated for different problems and different CFL numbers (larger CFL numbers lead to larger timestep sizes). CFL=1, 2 and 4 were used, CFL=1 is always stable, but CFL=2 and CFL=4 may not be always stable. Figure 5.31 to Figure 5.34 show the stable timestep size (CFL=1) for each problem, compared with the stable timestep size of the IMPES model. Figure 5.35 to Figure 5.38 show the well block oil saturation for each problem, and the legend in each figure marks the model used, such as CFL=2 stands for the IMPSAT model with a maximum CFL number of 2. The fully implicit result (the dark solid line) for maximum timestep size of 30 days is included for comparison. From these results, we can draw the following conclusions:

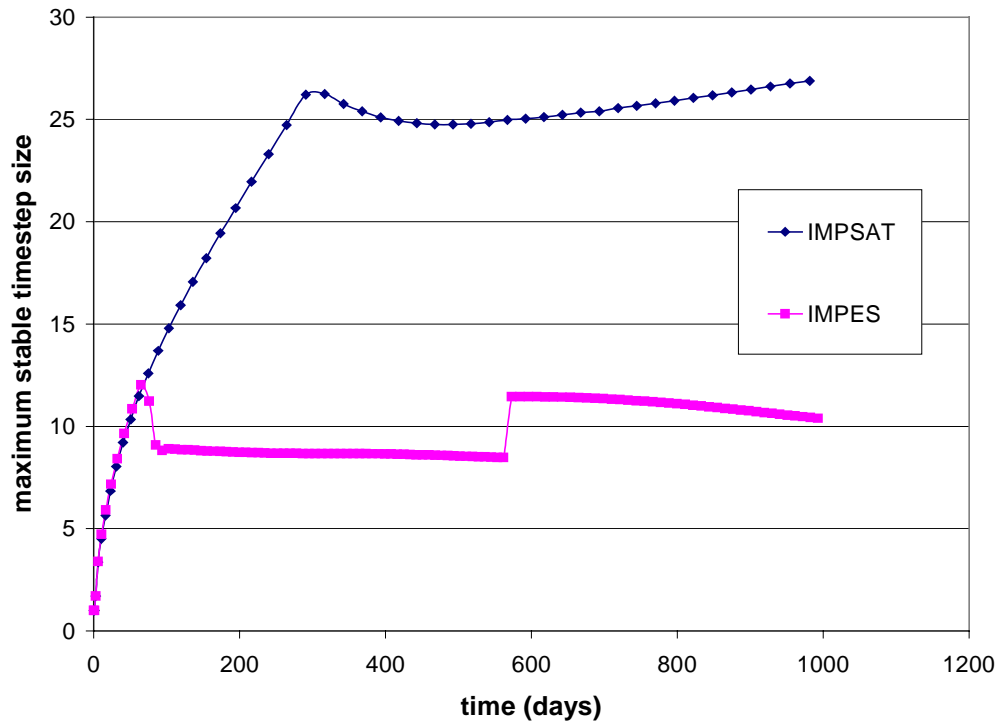
- The stable timestep sizes for the IMPSAT model are quite good, ranging from 6 days to 90 days, which is 2 to 7 times larger than that for the IMPES model. In the IMPSAT model, saturation is treated implicitly, so the existence of gas flow does not create any problem.
- The stable timestep sizes of IMPSAT always change smoothly, while for IMPES we observed sudden jumps. The CFL limit of the IMPSAT model depends on the component velocity.
- The results of the IMPSAT model agree well with the results of the FIM model, and the slight differences between the two are mostly due to the differences in timestep size. For some problems, such as Case 1 and 3, the timestep size of IMPSAT reached 100 days, while for FIM the maximum timestep size was limited to 30 days.
- In theory, the IMPSAT model should have less numerical diffusion than the FIM model, however for the problems tested here, we observe the same level of numerical diffusion for the IMPSAT and FIM models.
- For Case 1, 2 and 3, IMPSAT with CFL=4 is stable. But with the increase of number of components, such as in Case 4, this limit is also reduced, and CFL=2 or 2.5 is a safe limit.



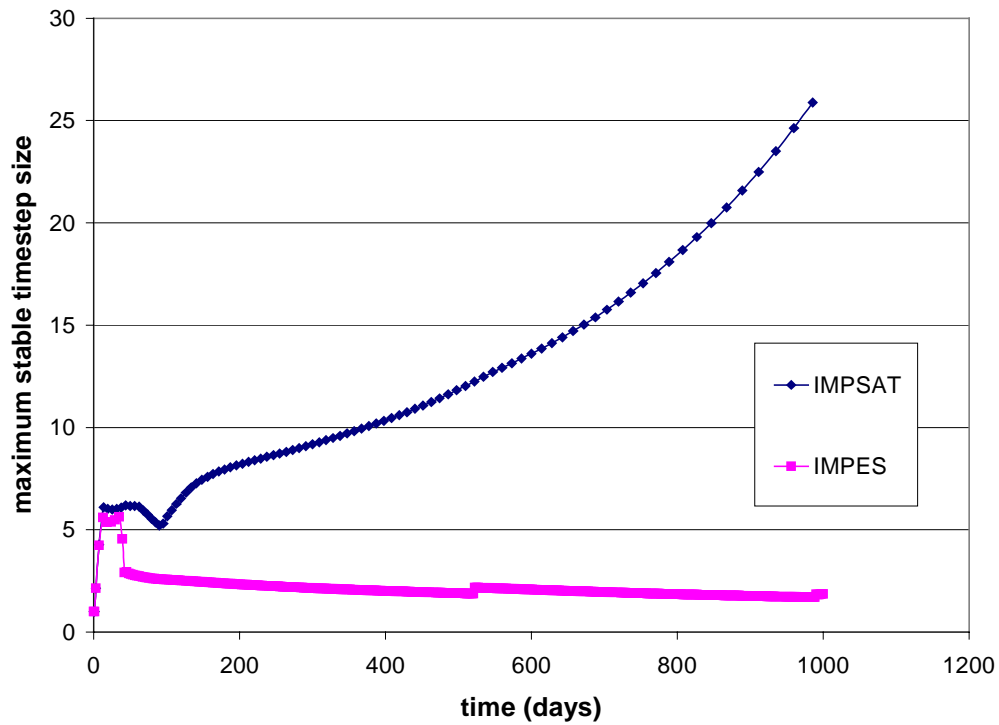
**Figure 5.31** Stable timestep size for the IMPSAT model and Case 1 problem



**Figure 5.32** Stable timestep size for the IMPSAT model and Case 2 problem

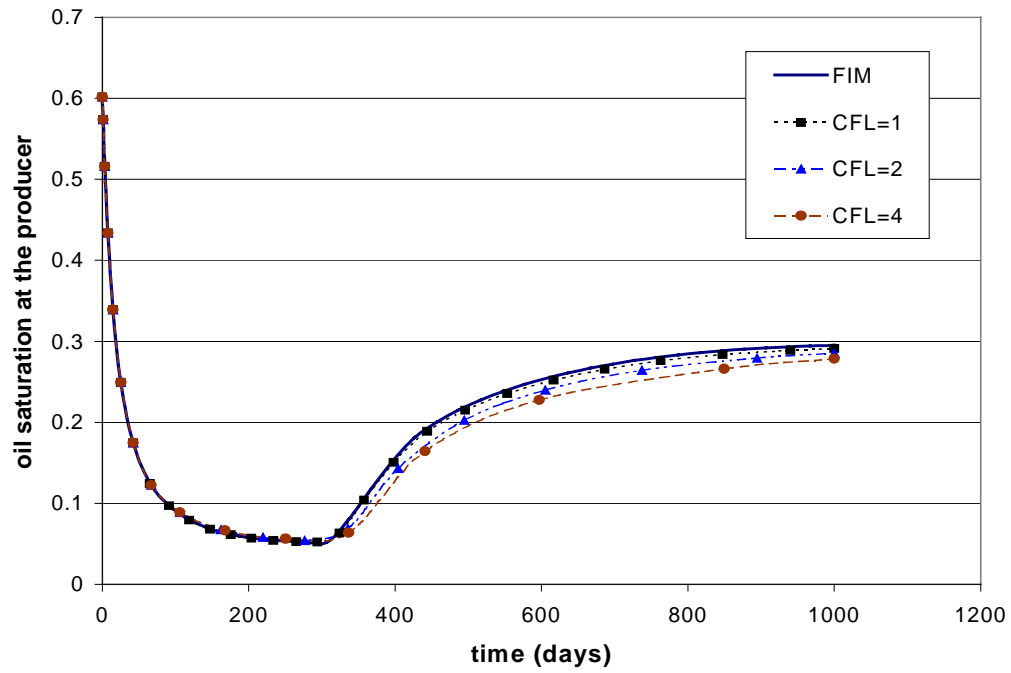


**Figure 5.33** Stable timestep size for the IMPSAT model and Case 3 problem

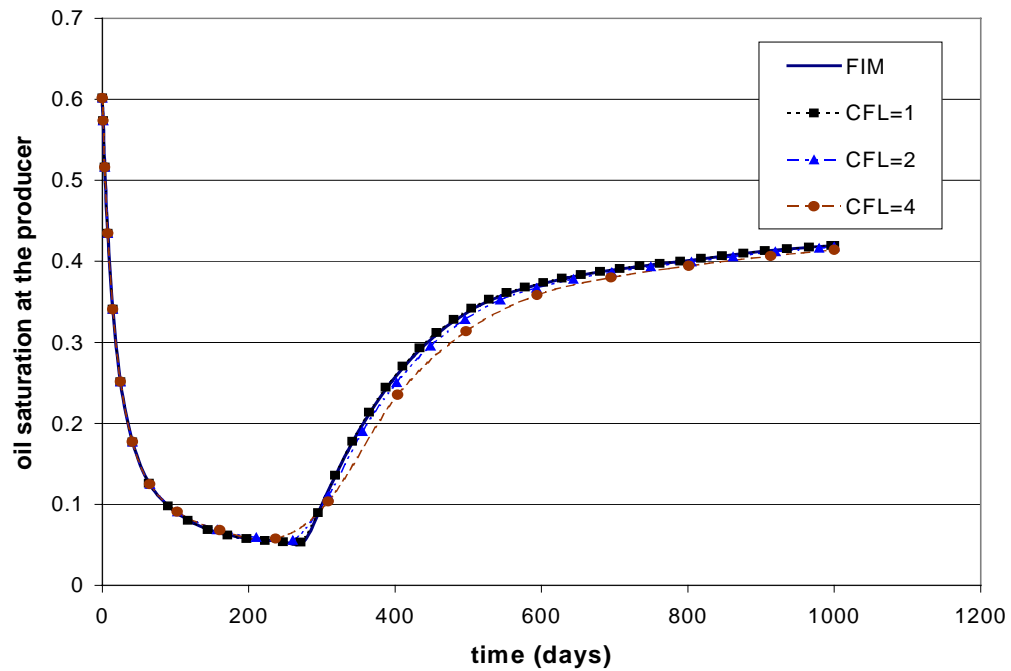


**Figure 5.34** Stable timestep size for the IMPSAT model and Case 4 problem

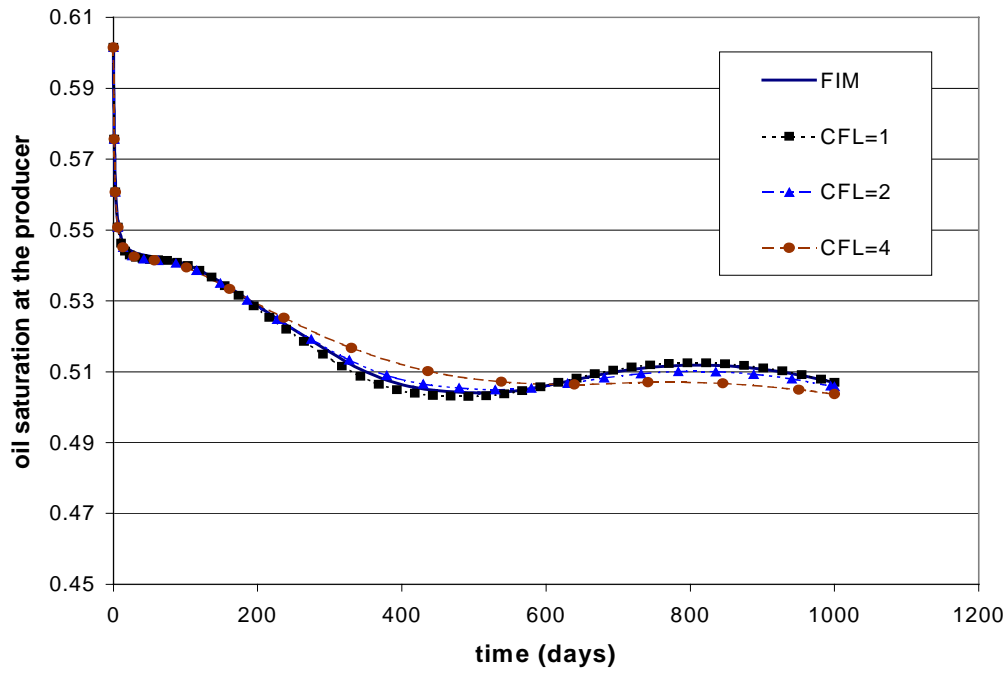




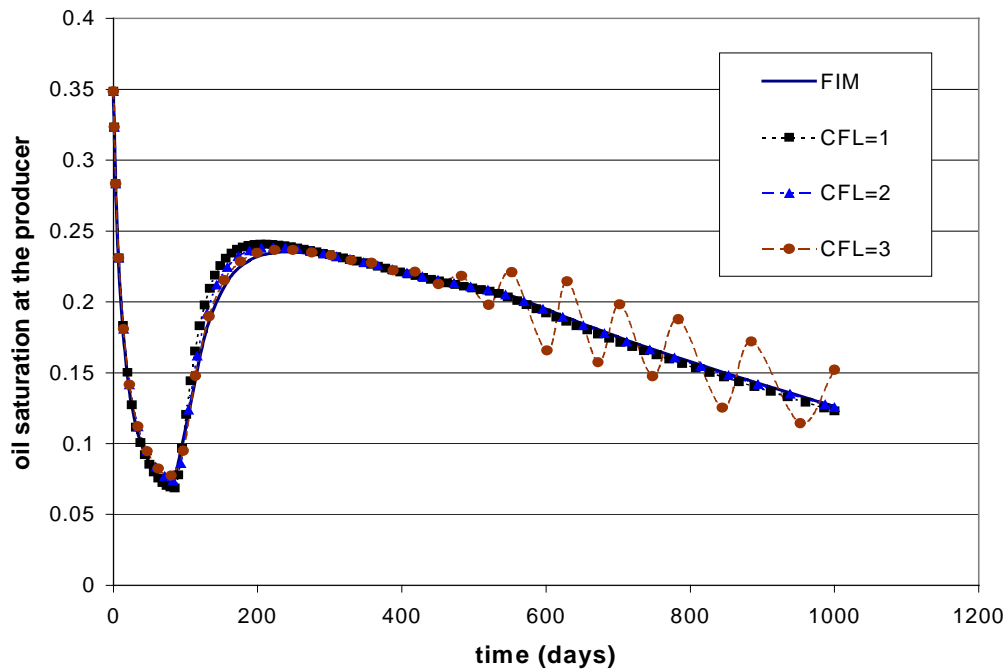
**Figure 5.35** Well block oil saturation comparisons for the IMPSAT model and Case 1 problem



**Figure 5.36** Well block oil saturation comparisons for the IMPSAT model and Case 2 problem



**Figure 5.37** Well block oil saturation comparisons for the IMPSAT model and Case 3 problem



**Figure 5.38** Well block oil saturation comparisons for the IMPSAT model and Case 4 problem

Iteration counts and timing results for problems discussed above are presented in Table 5.10 to Table 5.13. For all of the IMPSAT runs, the CPR preconditioned GMRES method was used to solve the linear system. We can draw the following conclusions based on our analysis:

- For the IMPSAT model, with the increase of CFL number (timestep size), more Newton iterations are needed at each timestep and more linear solver iterations are needed to solve each linear system. This is because the system becomes less diagonally dominant as the timestep size increases.
- Compared to the IMPES model, the IMPSAT model needs fewer (over 50%) number of timesteps and Newton iterations due to its improved stability. IMPSAT may cost more in the linear solver part than IMPES, because of its increased number of unknowns, but it can also save a lot in the Jacobian matrix construction and flash calculations. Both of these costs are directly proportional to the number of Newton iterations, and they are two of the most time consuming parts in compositional simulation. Finally, the total cost of IMPSAT is generally much lower than the total cost of IMPES, and most of time, IMPSAT can cut the running time by half compared to IMPES.
- For most problems, the improved stability of the IMPSAT model enables it to use the same number of timesteps and Newton iterations as the FIM model (with maximum timestep size of 100 days). Because IMPSAT solves for fewer unknowns in the linear solver part, it is a cheaper alternative to the FIM model. All comparisons here between the IMPSAT model and the FIM model are based on that the FIM model uses 100 days as the maximum timestep size, which is quite large for typical compositional simulations (The default maximum timestep size used in Eclipse 300 is only 50 days). Of course, the FIM model is unconditionally stable, and it can use any timestep size.

In summary, the IMPSAT model is much better than the IMPES model due to its improved stability, and it is also cheaper than the FIM. For most problems, the IMPSAT model can outperform both the IMPES and FIM models. But if the problem is very hard, such as high flow rate or small gridblock sizes, the stable timestep size of IMPSAT can also be quite small (less than 1 day), since it is proportional to cell volume divided by the

flow rate, as shown in Eqn. 4.7. In such cases, FIM models and AIM models are appropriate.

	CFL=1	CFL=2	CFL=4
$N_{timesteps}$	27	18	15
$N_{Newton\_iters}$	76	56	50
$N_{solver\_iters}$	193	136	132
$T_{solver}(\text{sec})$	0.78	0.55	0.48
$T_{total}(\text{sec})$	3.04	2.25	2.05

**Table 5.10** Iteration and timing comparisons for the IMPSAT model and Case 1 problem

	CFL=1	CFL=2	CFL=4
$N_{timesteps}$	44	26	18
$N_{Newton\_iters}$	123	91	64
$N_{solver\_iters}$	323	255	186
$T_{solver}(\text{sec})$	1.88	1.51	1.2
$T_{total}(\text{sec})$	6.17	4.61	3.41

**Table 5.11** Iteration and timing comparisons for the IMPSAT model and Case 2 problem

	CFL=1	CFL=2	CFL=4
$N_{timesteps}$	50	28	17
$N_{Newton\_iters}$	129	81	58
$N_{solver\_iters}$	348	221	158
$T_{solver}(\text{sec})$	6.1	4.05	2.86
$T_{total}(\text{sec})$	22.53	14.4	10.5

**Table 5.12** Iteration and timing comparisons for the IMPSAT model and Case 3 problem

	CFL=1	CFL=2	CFL=3
$N_{timesteps}$	93	50	37
$N_{Newton\_iters}$	230	165	151
$N_{solver\_iters}$	546	405	424
$T_{solver}(\text{sec})$	3.44	3	3.42
$T_{total}(\text{sec})$	37.13	25.68	24.92

**Table 5.13** Iteration and timing comparisons for the IMPSAT model and Case 4 problem

#### 5.2.4 AIM models

The traditional AIM model and three new IMPSAT based AIM models were used here. Model performance is compared for different problems. For AIM models, there are two ways to decide the timestep size and assign implicit level for each gridblock, and they are discussed below:

- **Fix timestep size** For a given timestep size, assign the implicit level for each gridblock according to the stability criteria (CFL number at that gridblock). This method is straightforward, and the percentage of gridblocks for each formulation changes from timestep to timestep.
- **Fix percentage** For a fixed percentage of gridblocks for each formulation, first decide the maximum stable timestep size which can satisfy the given percentage, then assign the implicit level for each gridblock according to the stability criteria. This method is not straightforward, in order to decide the maximum stable timestep size for a given percentage, we need to order the gridblocks according their CFL number.

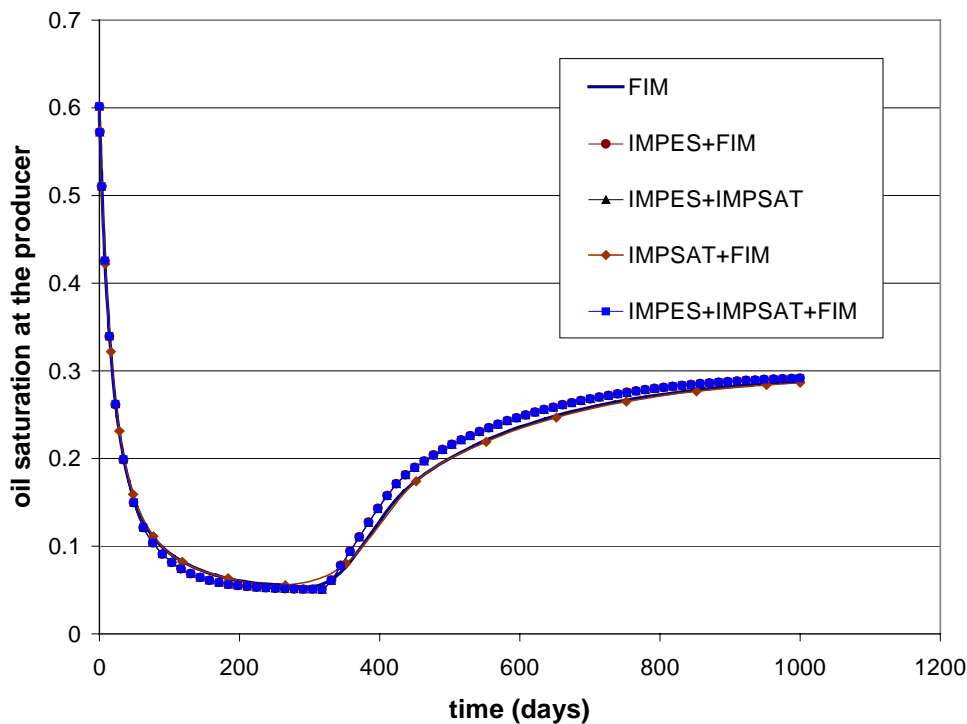
For AIM systems, the performance of linear solvers and preconditioners depends strongly on the percentage of gridblocks for each formulation. For example, CPR is more efficient for systems where most gridblocks are FIM, while AMG is more efficient for systems where most gridblocks are IMPES. Based on these considerations, the method of fixed percentage is used in GPRS. The percentage for each AIM model was assigned as following:

- **IMPES+FIM** 90% IMPES gridblocks and 10% FIM gridblocks
- **IMPES+IMPSAT** 90% IMPES gridblocks and 10% IMPSAT gridblocks

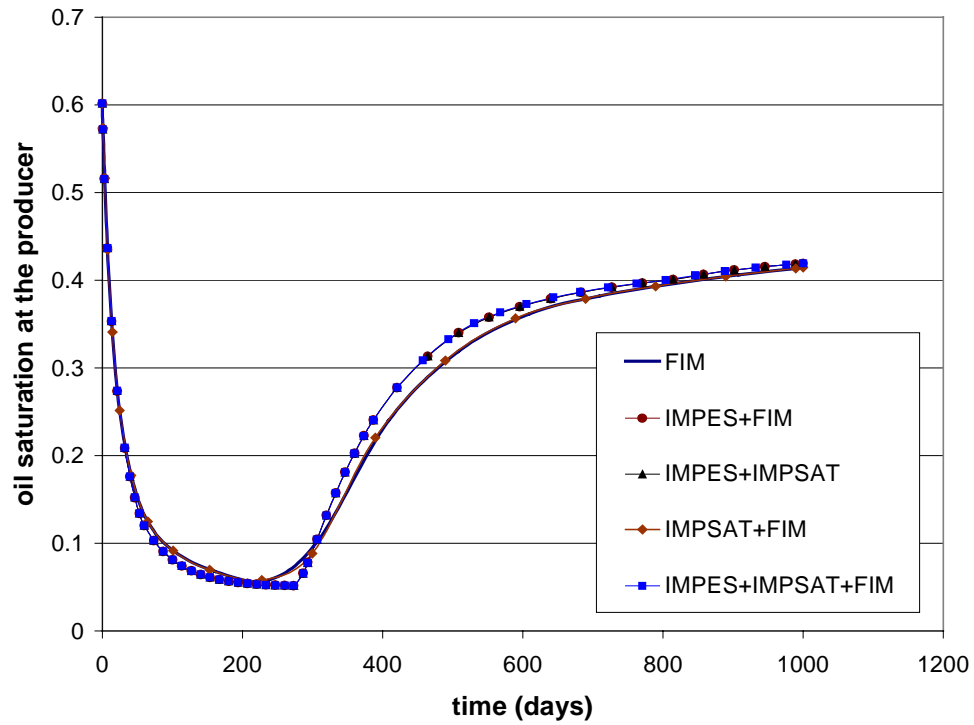
- **IMPSAT+FIM** 90% IMPSAT gridblocks and 10% FIM gridblocks
- **IMPES+IMPSAT+FIM** 90% IMPES gridblocks, 9% IMPSAT gridblocks and 1% FIM gridblocks

The well blocks were forced to be either FIM or IMPSAT. The maximum timestep size was set to be 100 days. Figure 5.39 to Figure 5.43 show the well block oil saturation for each problem, and the legend in each figure marks the model used. The fully implicit result (the dark solid line) for maximum timestep size of 100 days is included for comparison. From these results, we can draw the following conclusions:

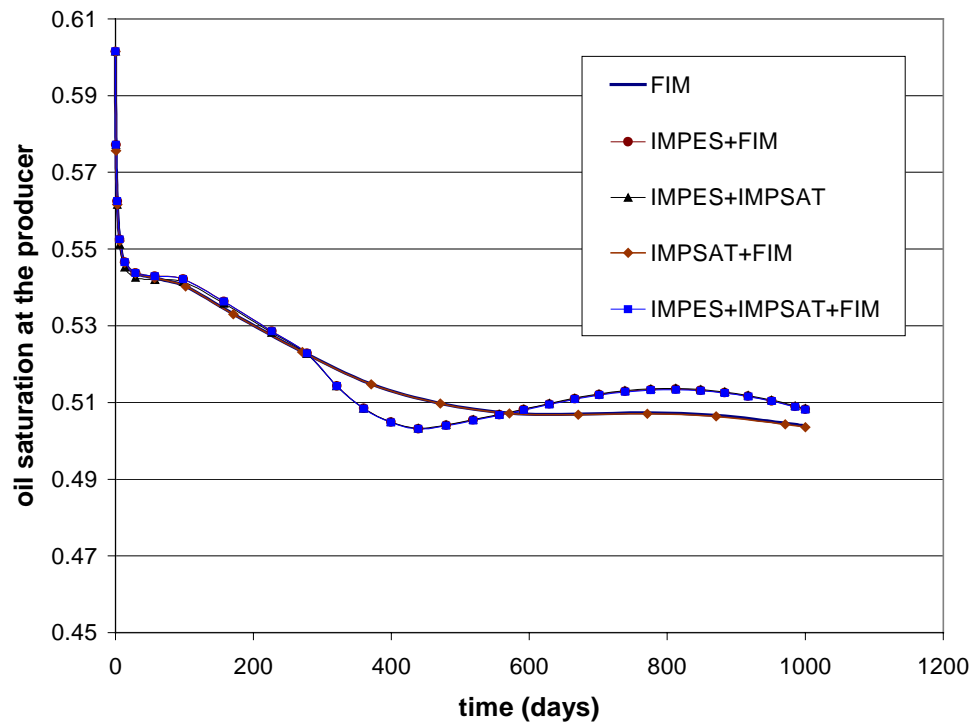
- The results of all AIM models agree well with the results of the FIM model.
- The results of AIM models with an IMPES formulation are on top of each other, while the results of the FIM and the IMPSAT+FIM models are on top of each other. This is mostly due to the difference in timestep sizes, IMPES timestep size can not reach 100 days, but without IMPES, it can reach 100 days. Due to this, we observe obvious differences in the results of different AIM models for Case 3 and 5.



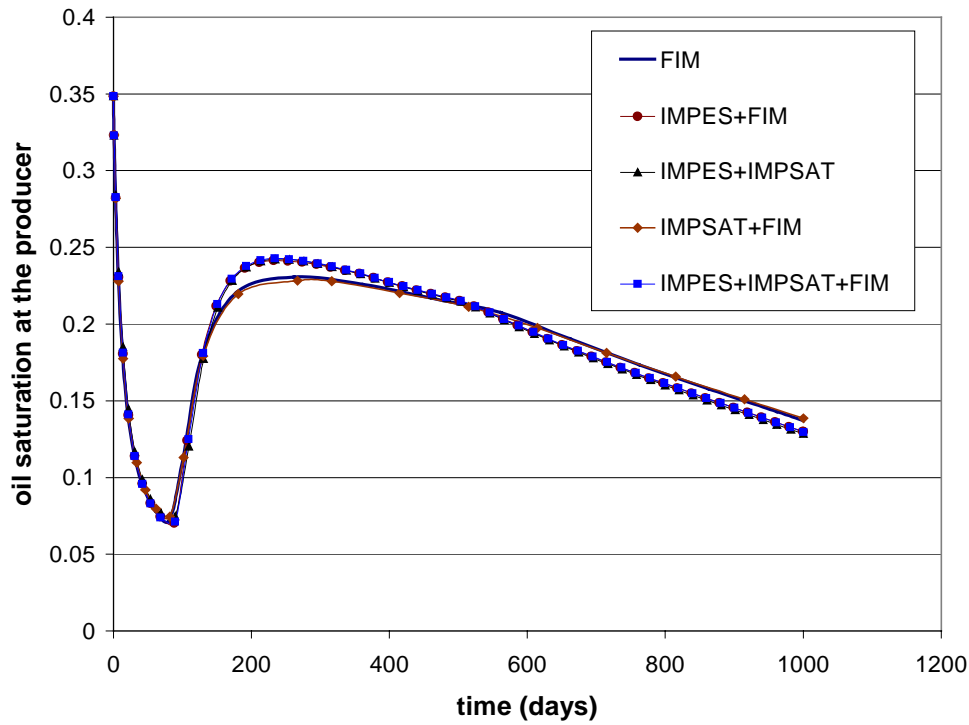
**Figure 5.39** Well block oil saturation comparisons for AIM models and Case 1 problem



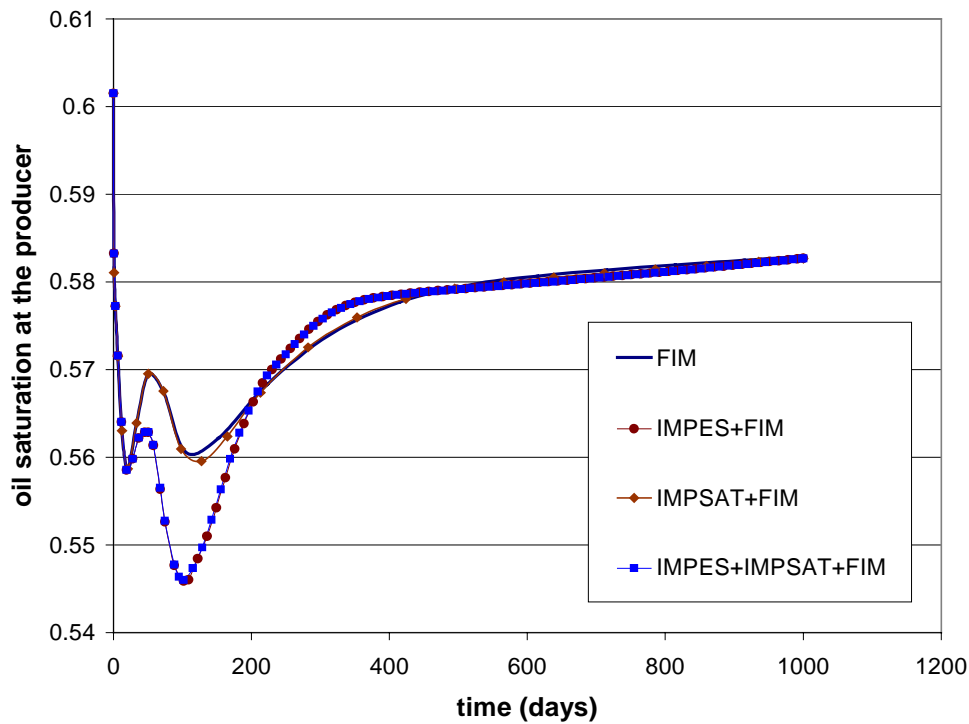
**Figure 5.40** Well block oil saturation comparisons for AIM models and Case 2 problem



**Figure 5.41** Well block oil saturation comparisons for AIM models and Case 3 problem



**Figure 5.42** Well block oil saturation comparisons for AIM models and Case 4 problem



**Figure 5.43** Well block oil saturation comparisons for AIM models and Case 5 problem



Iteration counts and timing results for problems discussed above are presented in Table 5.14 to Table 5.18. For all of the AIM runs, the ILU0 preconditioned GMRES method was used to solve the linear system. We can draw the following conclusions based on our analysis:

- For problems that are easy for IMPSAT, such as Case 1 and 3, the IMPES+IMPSAT model is faster than the traditional AIM model, since IMPSAT solves for fewer unknowns than FIM. On the other hand, the IMPES+IMPSAT model is less stable than the IMPSAT model, so any problem that can not be handled by the IMPSAT model, can also not be handled by the IMPES+IMPSAT model.
- For problems that are hard for IMPES, such as Case 1, 2, 4 and 5, the IMPSAT+FIM model needs far fewer timesteps and Newton iterations than the traditional AIM model, due to the improved stability of IMPSAT. Hence for hard problems, the IMPSAT+FIM model is much faster than the traditional AIM model.
- The ILU0 preconditioner used here is quite weak compared to the preconditioners (CPR, AMG) used for other models, so it won't make any sense to use the timing results here to compare the performance of AIM models with other models. But in theory, with a good preconditioner, AIM models should be much faster than FIM models, because they solve for fewer unknowns.

In summary, depending on the nature of the problem, one of the three new AIM models will outperform the traditional AIM model. The IMPES+IMPSAT+FIM model is the general AIM model, with proper tuning of the percentages according to the difficulties for the IMPES and IMPSAT models, it will always outperform the other three AIM models.

	IMPES+FIM	IMPES+IMPSAT	IMPSAT+FIM	IMPES+IMPSAT+FIM
$N_{timesteps}$	80	80	18	80
$N_{Newton\ iters}$	177	177	62	178
$N_{solver\ iters}$	1415	1392	701	1471
$T_{solver}(\text{sec})$	0.67	0.44	0.54	0.45
$T_{total}(\text{sec})$	5.53	5.27	2.45	5.32

**Table 5.14** Iteration and timing comparisons for AIM models and Case 1 problem

	IMPES+FIM	IMPES+IMPSAT	IMPSAT+FIM	IMPES+IMPSAT+FIM
$N_{timesteps}$	50	50	19	51
$N_{Newton\ iters}$	136	148	69	139
$N_{solver\ iters}$	1374	1419	820	1330
$T_{solver}(\text{sec})$	0.64	0.52	0.7	0.53
$T_{total}(\text{sec})$	5.83	6.25	2.97	5.88

**Table 5.15** Iteration and timing comparisons for AIM models and Case 2 problem

	IMPES+FIM	IMPES+IMPSAT	IMPSAT+FIM	IMPES+IMPSAT+FIM
$N_{timesteps}$	30	30	17	30
$N_{Newton\ iters}$	73	73	58	73
$N_{solver\ iters}$	770	813	657	775
$T_{solver}(\text{sec})$	1.3	0.83	2.27	1.1
$T_{total}(\text{sec})$	10.54	10	9.93	10.23

**Table 5.16** Iteration and timing comparisons for AIM models and Case 3 problem

	IMPES+FIM	IMPES+IMPSAT	IMPSAT+FIM	IMPES+IMPSAT+FIM
$N_{timesteps}$	55	54	21	54
$N_{Newton\ iters}$	154	158	84	147
$N_{solver\ iters}$	2355	2048	1943	2219
$T_{solver}(\text{sec})$	2.66	0.64	3.99	1.39
$T_{total}(\text{sec})$	21.66	22.19	16.54	20.98

**Table 5.17** Iteration and timing comparisons for AIM models and Case 4 problem

	IMPES+FIM	IMPES+IMPSAT	IMPSAT+FIM	IMPES+IMPSAT+FIM
$N_{timesteps}$	82	X	23	82
$N_{Newton\ iters}$	204	X	87	206
$N_{solver\ iters}$	25652	X	18612	25151
$T_{solver}(\text{sec})$	15744	X	19096	10563
$T_{total}(\text{sec})$	22026	X	20734	17135

**Table 5.18** Iteration and timing comparisons for AIM models and Case 5 problem



## Chapter 6

### Conclusions and Future work

#### 6.1 Conclusions

A new General Purpose Research Simulator (GPRS) has been developed. It incorporates various simulation models and techniques so that their advantages and disadvantages can be investigated in a consistent manner. The design of GPRS is such that it can be used by multiple researchers. The overall design and basic implementations of GPRS have been completed. Various models and techniques in GPRS have been tested, including a black-oil model, several compositional models, unstructured grid handling and multi-point flux approximation. It is now possible for other researchers to start working on extensions of individual modules. A basic introduction to the structure of GPRS is included in Appendix A.

A new General Formulation Approach has been developed and implemented in GPRS to facilitate the implementation of different models. This approach can generate almost any model, regardless of the selection of variables and the level of implicitness. All of the operations are performed on individual gridblocks, so different variables and different implicit levels can be used in different gridblocks. This is necessary for the building of AIM models. The approach presented is suitable for all kinds of simulations. In GPRS, it is only used to generate different compositional models.

An extensive study has been conducted for the new IMPSAT (implicit pressure, implicit saturations and explicit mole fractions) model, which is actually a special case of the General Formulation Approach. We have analyzed the characteristic of this IMPSAT model, derived its stability criterion and suggested an efficient way for its construction. We have also compared the performance of the IMPSAT model with the performance of IMPES and FIM models using various compositional problems. In addition, we have proposed three new IMPSAT based AIM models, and their performance has been compared with the performance of the traditional AIM model.

Different compositional models, varying in their selection of variables and implicit levels, have been evaluated using several compositional problems. These problems were

selected to be difficult in one specific area. From the results obtained, we can draw the following conclusions for different variable selections, for different implicit levels and for the new IMPSAT based AIM models:

### **Different Variable Selections**

- The same models (FIM or IMPES) with Type A and Type B variables have the same solutions, because they are solving the same equations.
- For the same models (FIM or IMPES) with Type A and Type B variables, there are no major differences in the Newton iteration numbers.
- For FIM models, Type B variables need more linear solver iterations (and hence computational cost) than Type A variables. Type B variables also increase the cost of Jacobian matrix construction, because of the need to use the chain rule to calculate derivatives. Hence Type A FIM model is more efficient than Type B FIM model.
- For IMPES models, there are no significant differences in linear solver iterations, since the pressure system is always the same. However, Type B variables are more efficient than Type A variables in Jacobian matrix calculation (Coats, 1999). Hence the total cost of Type B IMPES model is lower than the total cost of Type A IMPES model.

### **Different Implicit Levels**

- FIM models are most stable, and they can use large timestep sizes, but the cost of each Newton iteration is also high, especially for problems with large number of components.
- IMPES models are least stable, because only pressure is treated implicitly, but their cost is the lowest for each Newton iteration. For compositional problems, the stable timestep sizes of the IMPES model are small, due to the high gas mobility. Also it has sudden jumps because of the discontinuous phase velocities. For relatively easy problems, the IMPES model is faster than the FIM model, because it solves for fewer unknowns. However this advantage is lost for hard problems, where the timestep restriction makes IMPES even slower than the FIM model. The IMPES model has less numerical diffusion than the FIM model.

- The IMPSAT model is much more stable than the IMPES model, due to the implicit treatment of saturations, and it can use timestep sizes of up to 10 times larger than what is possible with IMPES. Also the stable timestep sizes of the IMPSAT model change more smoothly than the stable timestep sizes of the IMPES model. The cost of the IMPSAT model for each Newton iteration is fixed, independent of the number of components, just as is the case for the IMPES model. Compared to the IMPES model, the IMPSAT model needs far fewer timesteps and Newton iterations, and generally can cut the running time by half. For problems that are not very hard, the improved stability of the IMPSAT model enables it to use the same number of timesteps and Newton iterations as the FIM model. Because the number of unknowns to be solved is reduced, IMPSAT is a cheaper alternative to the FIM model. The IMPSAT model has the same level of numerical diffusion as the FIM model.

### **Different AIM Models**

- All AIM models are more efficient than the FIM model, since high flow rates are generally restricted to a small portion of the reservoir and the FIM model is only needed in the regions of high flow rates.
- For problems that are hard for IMPES, the IMPSAT+FIM model is much faster than the traditional IMPES+FIM model due to the improved stability of the IMPSAT model.
- For problems that are easy for IMPSAT, the IMPES+IMPSAT model is faster than the traditional IMPES+FIM model, due to the reduction in the number of unknowns that need to be solved implicitly.
- The IMPES+IMPSAT+FIM model is the most general AIM model, with properly tuned percentages for each formulation, it will always outperform the other three AIM models.

In summary, the key conclusions about the performance of compositional models are as following:

- Type A FIM model is better than Type B FIM model, both in the condition of the Jacobian matrix and the cost of building the Jacobian matrix.

- Type B IMPES model is more efficient than Type A IMPES model in the cost of building the pressure implicit Jacobian matrix.
- The IMPSAT model is generally over 50% faster than the IMPES model due to its improved stability.
- For problems that are not very hard for IMPSAT, the IMPSAT model is cheaper than the FIM model since it reduces the number of unknowns that have to be solved implicitly. This is especially true for problems with a large number of components.
- The IMPSAT based AIM models are more flexible, more stable and less expensive than the traditional AIM model. With properly tuned percentages for each formulation, the IMPES+IMPSAT+FIM model will always outperform the traditional AIM (IMPES+FIM) model.

## 6.2 Future Work

GPRS is a general-purpose research simulator. It requires further testing, improvements and enhancements. Work is especially needed in the following areas:

- Evaluation of IMPSAT++ models (including some mole fractions in the implicit set)
- Evaluation of quasi-IMPSAT model, where the mole fractions in the transmissibility part are updated iteration by iteration
- Improvements in preconditions for AIM models
- Better solver for unstructured grid
- Improvements in flash calculations
- Implementation of techniques for parallel computing
- Surface facility modeling
- Complex well modeling
- Automatic selection of the level of implicitness in AIM models
- Thermal modeling
- Fractured reservoir simulation
- Enhancements in the treatment of water to include gas solubility in water and water vapor in gas
- Other structured grid (cylindrical, curvilinear, corner-point)
- Coupled modeling of geomechanics
- Interface with other tools, and allow use of various units



## Nomenclature

$A$	Area of surface between two gridblocks, [ft <sup>2</sup> ]
$B_p$	Formation volume factor of phase $p$ (black-oil), [bbl/STB: oil&wat; ft <sup>3</sup> /SCF: gas]
$D$	Depth, [ft]
$F$	Total moles of hydrocarbon in one unit volume of fluid, [mole/ft <sup>3</sup> ]
$f_{c,p}$	Fugacity of component $c$ in phase $p$ , [psia]
$g$	Gravity force constant, [32.2 ft/sec <sup>2</sup> ]
$K_c$	Equilibrium ratio of component $c$
$k$	Absolute permeability, [md]
$k_{rp}$	Relative permeability of phase $p$
$l$	Hydrocarbon liquid mole fraction in hydrocarbon fluid
$M_c$	Total moles of hydrocarbon component $c$ , [lb-mole]
$M_w$	Total moles of water, [lb-mole]
$MW_c$	Molecular weight of component $c$ , [lbm/lb-mole]
$MW_p$	Molecular weight of phase $p$ , [lbm/lb-mole]
$n_c$	Number of components
$n_h$	Number of hydrocarbon components
$n_p$	Number of phases
$P_{c,p_1,p_2}$	Capillary pressure between phase $p_1$ and phase $p_2$ , [psia]
$p$	Block pressure, [psia]
$p^W$	Wellbore pressure of well $W$ , [psia]
$Q_c^W$	Mass flow rate of component $c$ at well $W$ , [lbm/day]
$q_p^W$	Volumetric flow rate of phase $p$ at well $W$ , [ft <sup>3</sup> /day]
$R_{c,p}$	Solubility of component $c$ in phase $p$ , [SCF/STB: $R_{g,o}$ ]
$r_c$	Total moles of hydrocarbon component $c$ in one unit volume of fluid, [lb-mole/ft <sup>3</sup> ]
$S_p$	Saturation of phase $p$
$T$	Transmissibility, [md·ft]
$V$	Volume, [ft <sup>3</sup> ]
$V_\phi$	Pore volume, [ft <sup>3</sup> ]
$V_p$	Volume of phase $p$ , [ft <sup>3</sup> ]
$V_T$	Total fluid volume, [ft <sup>3</sup> ]

$v_p$	Volumetric flow rate of phase $p$ , [ft <sup>3</sup> /day]
$W$	Total moles of water in one unit volume of fluid, [lb-mole/ft <sup>3</sup> ]
$WI^W$	Well Index of well $W$ , [md·ft]
$X_{c,p}$	Mole fraction of component $c$ in phase $p$
$x_c$	Mole fraction of component $c$ in the oil phase, same as $X_{c,o}$
$y_c$	Mole fraction of component $c$ in the gas phase, same as $X_{c,g}$
$z_c$	Overall mole fraction of component $c$
$Z$	Compressibility factor

### Acronyms

AIM	Adaptive implicit
AMG	Algebraic multi-grid
BHP	Bottom hole pressure
CPR	Constrained pressure residual
EOS	Equation of state
FIM	Fully implicit
GMRES	Generalized minimum residual
GPRS	General Purpose Research Simulator
ILU	Incomplete LU decomposition
IMPES	Implicit pressure and explicit saturations
IMPSAT	Implicit pressure and saturations
Type A	Type A variables (pressure, saturations and component mole fractions)
Type B	Type B variables (pressure, overall component densities)

### Greek

$\Delta$	Capital Delta (change)
$\delta$	Delta (change)
$\alpha, \beta$	Unit conversion constants
$\lambda_p$	Mobility of phase $p$ , [1/cp]
$\rho_p$	Density of phase $p$ , [lbm/ft <sup>3</sup> ]
$\rho_{\bar{c}}$	Density of component $c$ at standard condition, [lbm/ft <sup>3</sup> ]
$\mu_p$	Viscosity of phase $p$ , [cp]

$\Phi_p$  Potential of phase  $p$  , [psia]  
 $\phi$  Porosity

### **Subscripts**

$p$  Phase  
 $c$  Component  
 $w$  Water

### **Superscripts**

$n$  Time level  $n$   
 $n + 1$  Time level  $n + 1$   
 $v$  Iteration level  
 $W$  Well



## References

1. Aavatsmark, I., Barkve, T. and Mannseth, T.: "Control Volume Discretization Methods for 3D Quadrilateral Grids in Inhomogeneous, Anisotropic Reservoirs", paper SPE 38000, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
2. Abou-Kassem, J.H. and Aziz, K.: "Handling of Phase Change in Thermal Simulators", JPT, September 1985, pp1661-1663
3. Acs, G., Doleschall, S. and Farkas, E.: "General Purpose Compositional Model", SPEJ, August 1985, pp543-553
4. Agut, R., Edwards, G.E., Verma, S. and Aziz, K.: "Flexible Streamline-Potential Grids With Discretization on Highly Distorted Cells", proceedings of the 6<sup>th</sup> European Conference on the Mathematics of Oil Recovery, Peebles, Scotland, September 8-11, 1998
5. Almehaideb, R.A., Aziz, K. and Pedrosa, D.A.: "A Reservoir Wellbore Model for Multiphase Injection and Production", paper SPE 17941, proceedings of the SPE Middle East Oil Technical Conference and Exhibition, Manama, Bahrain, March 11-14, 1989
6. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., Mckenny, A. and Sorensen, D.: "LAPACK Users' Guide, Third Edition", Society for Industrial and Applied Mathematics, 1999
7. Aziz, K.: "Fundamentals of Reservoir Simulation", class notes for PE223 (Reservoir Simulation) class, summer 1996
8. Aziz, K. and Settari, A.: "Petroleum Reservoir Simulation", Applied Science Publishers, London, England, 1979
9. Aziz, K. and Wong, T.W.: "Considerations in the Development of Multipurpose Reservoir Simulation Models", proceedings of the 1<sup>st</sup> and 2<sup>nd</sup> International Forum on Reservoir Simulation, Alpbach, Austria, September 12-16, 1988 and September 4-8, 1989

10. Baker, L.E. and Luks, K.D.: "Critical Point and Saturation Pressure Calculation for Multi-Component Systems", SPEJ, February 1980, pp15-24
11. Ballin, P.R., Aziz, K., Journel, A.G. and Zuccolo, L.: "Quantifying the Impact of Geological Uncertainty on Reservoir Forecasts", paper SPE 25238, proceedings of the 12<sup>th</sup> SPE Symposium on Reservoir Simulation, New Orleans, CA, February 28-March 03, 1993
12. Behie, A. and Forsyth Jr., P.A.: "Practical Considerations for Incomplete Factorization Methods in Reservoir Simulation", paper SPE 12263, proceedings of the 7<sup>th</sup> SPE Symposium on Reservoir Simulation, San Francisco, CA, November 15-18, 1983
13. Branco, C.M. and Rodriguez, F.: "A Semi-Implicit Formulation for Compositional Reservoir Simulation", paper SPE 27053, SPE Advanced Technology Series, Vol. 4, No. 1, 1995
14. Byer, T.J.: "Preconditioned Newton Methods for Reservoirs With Surface Facilities", Ph.D. Dissertation, Stanford University, June 2000
15. Cao, H.: "GPRS Package (code, data file and user's manual) 2002", 2002
16. Castellini, A., Edwards, M.G. and Durlofsky, L.J.: "Flow Based Modules for Grid Generation in Two and Three Dimensions", proceedings of the 7<sup>th</sup> European Conference on the Mathematics of Oil Recovery, Baveno, Italy, September 5-8, 2000
17. Chen, W.H., Durlofsky, L.J., Engquist, B. and Osher, S.: "Minimization of Grid Orientation Effects Through Use of Higher-Order Finite Difference Methods," paper SPE 22887, proceedings of the 66<sup>th</sup> SPE Annual Technical Conference and Exhibition, Dallas, TX, October 6-9, 1991
18. Chien, M.C.H., Lee, S.T. and Chen, W.H.: "A New Fully Implicit Compositional Simulator", paper SPE 13385, proceedings of the 8<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, February 10-13, 1985
19. Coats, K.H.: "An Equation of State Compositional Model", SPEJ, October 1980, pp363-376

20. Coats, K.H.: "A Note on IMPES and Some IMPES Based Simulation Models", paper SPE 49774, proceedings of the 15<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 14-17, 1999
21. Coats, K.H.: "IMPES Stability: The Stable Step", paper SPE 69225, proceedings of the 16<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 11-14, 2001
22. Coats, K.H., Thomas, L.K. and Pierson, R.G.: "Compositional and Black-Oil Reservoir Simulation", paper SPE 29111, proceedings of the 13<sup>th</sup> SPE Symposium on Reservoir Simulation, San Antonio, TX, February 12-15, 1995
23. Coats, K.H., Thomas, L.K. and Pierson, R.G.: "Compositional and Black-Oil Reservoir Simulation", paper SPE 50990, SPE Reservoir Evaluation & Engineering, August 1998, pp372-279
24. Ciment, M. and Sweet, R.A.: "Mesh Refinement for Parabolic Equations", J. Comp. Phys., December 1973, pp513-525
25. Deo, M.D., Nutakki, R. and Orr Jr., F.M.: "Schmidt-Wenzel and Peng-Robinson Equation of State for CO<sub>2</sub>-Hydrocarbon Mixtures: Binary Interaction Parameters and Volume Translation Factors", paper SPE 18796, proceedings of the SPE California Regional Meeting, Bakersfield, CA, April 5-7, 1989
26. Ding, Y. and Lemonnier, P.: "Use of Corner Point Geometry in Reservoir Simulation", paper SPE 29933, proceedings of the International Meeting on Petroleum Engineering, Beijing, China, November 14-17, 1995
27. Dogru, A.H., Pita, J.A., Fung, L.S., Al-Zamel, N., Al-Shaalan, T. and Dreiman, W.T.: "Megacell Reservoir Simulation: the Quest for High Resolution in Full-Field Modeling", proceedings of the 6<sup>th</sup> International Forum on Reservoir Simulation, Salzburg, Austria, September 3-7, 2001
28. Dongarra, J.: "Performance of Various Computers Using Standard Linear Equations Software", March 19, 2002, <http://www.netlib.org/benchmark/performance.ps>
29. Dongarra, J., Lumsdaine, A., Pozo, R. and Remington, K.A.: "IML++ V.1.2, Iterative Methods Library Reference Guide", National Institute of Standards and Technology, April 1996

30. Edwards, G.M.: “Split Elliptic Tensor Operators for Reservoir Simulation”, proceedings of the ICFD Conference on Numerical Methods for Fluid Dynamics, Oxford University, UK, March 31–April 03, 1998
31. Edwards, G.M., Agut, R. and Aziz, K.: “Quasi K-Orthogonal Streamline Grids: Gridding and Discretization”, paper SPE 49072, proceedings of the 73<sup>rd</sup> SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana, September 27-30, 1998
32. Farkas, E.: “Comparison of Linearization Techniques of Nonlinear Partial Differential Equations in Numerical Reservoir Simulation”, Ph.D. dissertation, Reservoir Engineering Department of the Montanuniversitat Leoben, Austria, 1997
33. Forsyth, P.A. and Sammon, P.H.: “Practical Considerations for Adaptive Implicit Methods in Reservoir Simulation”, J. Comp. Phys., Vol. 62, 1986, pp265-281
34. Fung, L.S., Hiebert, A.D. and Nghiem, L.X.: “Reservoir Simulation With a Control Volume Finite Element Method”, paper SPE 21224, proceedings of the 11<sup>th</sup> SPE Symposium on Reservoir Simulation, Anaheim, CA, February 17-20, 1991
35. Fussell, D.D. and Yanosik, J.L.: “An Iterative Sequence for Phase-Equilibria Calculations Incorporating the Redlich-Kwong Equation of State” SPEJ, June 1978, pp173-182
36. Fussell, L.T. and Fussell, D.D.: “An Iterative Technique for Compositional Reservoir Models” SPEJ, August 1979, pp211-220
37. Geoquest, Schlumberger: “Eclipse 100 Technical Description 2000A”, 2000, [http://www.sis.slb.com/content/software/simulation/eclipse\\_blackoil.asp](http://www.sis.slb.com/content/software/simulation/eclipse_blackoil.asp)
38. Geoquest, Schlumberger: “Eclipse 300 Technical Description 2000A”, 2000, [http://www.sis.slb.com/content/software/simulation/eclipse\\_compositional.asp](http://www.sis.slb.com/content/software/simulation/eclipse_compositional.asp)
39. Geoquest, Schlumberger: “FloGrid Reference Manual 2000A”, 2000, [http://www.sis.slb.com/content/software/simulation/eclipse\\_pre\\_post/flogrid.asp](http://www.sis.slb.com/content/software/simulation/eclipse_pre_post/flogrid.asp)
40. Gunasekera, D., Childs, P., Herring, J. and Cox, J.: “A Multi-Point Flux Discretization Scheme for General Polyhedral Grids”, paper SPE 48855, proceedings of the 6<sup>th</sup> SPE International Oil&Gas Conference and Exhibition, Beijing, China, November 2-6, 1998



41. Gunasekera, D., Cox, J. and Lindsey, P.: "The Generation and Application of K-Orthogonal Grid Systems", paper SPE 37998, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
42. Heinemann, Z.E. and Brand, C.W.: "Modeling Reservoir Geometry with Irregular Grids" paper SPE 18412, proceedings of the 10<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 6-8, 1989
43. Journel, A.G.: "Geostatistics for Reservoir Characterization", paper SPE 20750, proceedings of the 65<sup>th</sup> SPE Annual Technical Conference and Exhibition, New Orleans, CA, September 23-26, 1990
44. Kendall, R.P., Morrell, G.O., Peaceman, D.W., Silliman, W.J. and Watts, J.W.: "Development of a Multiple Application Reservoir Simulator for Use on a Vector Computer", paper SPE 11483, proceedings of the SPE Middle East Oil Technical Conference, Manama, Bahrain, March 14-17, 1983
45. Kenyon, D. and Behie, G.A.: "Third SPE Comparative Solution Project: Gas Cycling of Retrograde Condensate Reservoirs", paper SPE 12278, proceedings of the 7<sup>th</sup> SPE Symposium on Reservoir Simulation, San Francisco, CA, November 15-18, 1983
46. Killough, J.E.: "Ninth SPE Comparative Solution Project: A Reexamination of Black-Oil Simulation", paper SPE 29110, proceedings of the 13<sup>th</sup> SPE Symposium on Reservoir Simulation, San Antonio, TX, February 12-15, 1995
47. Killough, J.E. and Commander, D.: "Scalable Parallel Reservoir Simulation on a Windows NT-Bases Workstation Cluster", paper SPE 51883, proceedings of the 15<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 14-17, 1999
48. Kocerber, S.: "An Automatic, Unstructured Control Volume Generation System for Geologically Complex Reservoirs", paper SPE 38001, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
49. Lacroix, S., Vassilevski, Y. and Wheeler, M.: "Iterative Solvers of the Implicit Parallel Accurate Reservoir Simulator (IPARS), I: Single Processor Case", TICAM report 00-28, (2000), <http://www.ticam.utexas.edu/reports/2000/index.html>
50. Landmark Graphics Corporation: "BlitzPak User's Guide", 1998, <http://www.lgc.com>

51. Lee, J., Kasap, E. and Kelkar, M.G.: "Analytical Upscaling of Permeability for 3D Gridblocks", SPE paper 27875, proceedings of the SPE Western Regional Meeting, Long Beach, CA, March 23-25, 1994
52. Lee, S.H., Durlofsky, L.J., Lough, M.F. and Chen, W.H. "Finite Difference Simulation of Geologically Complex Reservoirs With Tensor Permeabilities", paper SPE 38002, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
53. Lim, K.T., Schiozer, D.J. and Aziz, K.: "A New Approach for Residual and Jacobian Array Construction in Reservoir Simulators", paper SPE 28248, proceedings of the SPE Petroleum Computer Conference, Dallas, TX, July 31-August 03, 1994
54. Nacul, E.C.: "Use of Domain Decomposition and Local Grid Refinement in Reservoir Simulation", Ph.D. Dissertation, Stanford University, March 1991
55. Nghiem, L.X., Aziz, K. and Li, Y.K.: "A Robust Iterative Method for Flash Calculations Using the Soave-Redlich-Kwong or the Peng-Robinson Equation of State", SPEJ, June 1983, pp521-530
56. Nolen, J.S.: "Treatment of Wells in Reservoir Simulation", proceedings of the 3<sup>rd</sup> International Forum on Reservoir Simulation, Baden, Austria, July 23-27, 1990
57. Odeh, A.S.: "Comparison of Solutions to a Three-Dimensional Black-Oil Simulation Problem", paper SPE 9723, JPT, January 1981, pp13-25
58. Palagi, C.L. and Aziz, K.: "Use of Voronoi Grid in Reservoir Simulation", SPE Advanced Technology Series 2, April 1994, pp69-77
59. Peaceman, D.W.: "Calculation of Transmissibilities of Gridblocks Defined by Arbitrary Corner Point Geometry", paper SPE 37306, 1996 (unpublished, but available from SPE e-library)
60. Peaceman, D.W.: "Interpretation of Well-Block Pressures in Numerical Reservoir Simulation", SPEJ, June 1978, pp183-194
61. Pedrosa, O.A. and Aziz, K.: "Use of Hybrid Grid in Reservoir Simulation", paper SPE 13507, proceedings of the 8<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, February 10-13, 1985.

62. Peng, D.Y. and Robinson, D.B.: "A New Two-Constant Equation of State", Ind. Eng. Chem. Fundam., Vol. 15, 1976, pp59-64
63. Ponting, D.K.: "Corner Point Geometry in Reservoir Simulation", proceedings of the 1<sup>st</sup> European Conference on Mathematics Of Oil Recovery, Cambridge, England, 1989, P.R.King(ed.), Clarendon Press, Oxford 1992, pp45-65
64. Pozo, R., Remington, K.A. and Lumsdaine, A.: "SparseLib++ v.1.5, Sparse Matrix Class Library Reference Guide", National Institute of Standards and Technology, April 1996
65. Prevost, M., private communication, 2002
66. Program Development Corporation (PDC): "GridPro", <http://www.gridpro.com>
67. Quandalle, P. and Savary, D.: "An Implicit in Pressure and Saturations Approach to Fully Compositional Simulation", paper SPE 18423, proceedings of the 10<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 6-8, 1989
68. Rachford, H. and Rice, J.: "Procedure for Use of Electronic Digital Computers in Calculating Flash Vaporization Hydrocarbon Equilibrium", JPT, 4, 1952, pp10-19
69. Ritzdorf, H.: "Readme to AMG Package", GMD Software, 1991
70. Russell, T.F.: "Stability Analysis and Switching Criteria for Adaptive Implicit Methods Based on the CFL Condition", paper SPE 18416, proceedings of the 10<sup>th</sup> SPE Symposium on Reservoir Simulation, Houston, TX, February 6-8, 1989
71. Saad, Y. and Schultz, M.: "GMRES: A Generalized Minimum Residual Algorithm for solving non-symmetric linear systems", SIAM J. Sci. Statist. Comput., Vol. 7, 1986, pp856-869
72. Sammon, P.H.: "A Nine-Point Differencing Scheme Based on High-Order Stream Tube Modeling", paper SPE 21223, proceedings of the 11<sup>th</sup> SPE Symposium on Reservoir Simulation, Anaheim, CA, February 17-20, 1991
73. Schiozer, D.J. and Aziz, K.: "Use of Domain Decomposition for Simultaneous Simulation of Reservoir and Surface Facilities", paper SPE 27876, proceedings of the SPE Western Regional Meeting, Long Beach, CA, March 23-25, 1994
74. Stueben, K.: "Algebraic Multigrid (AMG): Experiences and Comparisons", proceedings of the International Multigrid Conference, Copper Mountain, CO, April 6-8, 1983

75. Sukirman, Y.B. and Lewis, R.W.: "Three-Dimensional Fully Coupled Flow: Consolidation Modeling Using Finite Element Method", paper SPE 28755, proceedings of the SPE Asia Pacific Oil & Gas Conference, Melbourne, Australia, November 7-10, 1994
76. The GOCAD group: "GOCAD", <http://www.ensg.u-nancy.fr/GOCAD>
77. Thomas, G.W. and Thurnau, D.H.: "Reservoir Simulation Using an Adaptive Implicit Method", SPEJ, October 1983, pp759-768
78. Tran, T. and Journel, A.: "Automatic Generation of Corner-Point-Geometry Flow Simulation Grids from Detailed Geostatistical Descriptions", Stanford Center for Reservoir Forecasting , 1995
79. Vassilevski, Y.: "Two Marks on the Combinative Technique", private communication, 2001
80. Verma, S.: "Flexible Grids for Reservoir Simulation" Ph.D. Dissertation, Stanford University, June 1996
81. Verma, S. and Aziz, K.: "A Control Volume Scheme for Flexible Grids in Reservoir Simulation", paper SPE 37999, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
82. Verma, S. and Aziz, K.: "Two- and Three-Dimensional Flexible Grids for Reservoir Simulation", proceedings of the 5<sup>th</sup> European Conference on Mathematics of Oil Recovery, Leoben, Austria, September 3-6, 1996
83. Walas, S.M.: "Phase Equilibria in Chemical Engineering", Butterworth Publishers, Boston, MA, 1985
84. Watts, J.W.: "A Compositional Formulation of the Pressure and Saturation Equations" paper SPE 12244, SPE Reservoir Engineering, May 1986, pp243-252
85. Watts, J.W.: "Reservoir Simulation: Past, Present, and Future" paper SPE 38441, proceedings of the 14<sup>th</sup> SPE Symposium on Reservoir Simulation, Dallas, TX, June 8-11, 1997
86. Whitson, C.H. and Michelsen, M.L.: "The Negative Flash", proceedings of the International Conference on Fluid Properties and Phase Equilibria for Chemical Process Design, Banff, Canada, April 30-May 05, 1986

87. Wong, T.W., Firoozabadi, A. and Aziz, K.: "Relationship of the Volume-Balance Method of Compositional Simulation to the Newton-Raphson Method", paper SPE 18424, SPE Reservoir Engineering, August 1990, pp415-422
88. Yanosik, J.L. and McCracken, T.A.: "A Nine-Point Finite-Difference Reservoir Simulation for Realistic Prediction of Adverse Mobility Ratio Displacement", SPEJ, August 1979, pp253-262
89. Young, L.C.: "An Efficient Finite Element Method for Reservoir Simulation", paper SPE 7413, proceedings of the 53<sup>rd</sup> SPE Annual Technical Conference and Exhibition, Houston, TX, October 1-3, 1978
90. Young, L.C. and Stephenson, R.E.: "A Generalized Compositional Approach for Reservoir Simulation" SPEJ, October 1983, pp727-742



## **Appendix A Overview Of GPRS**

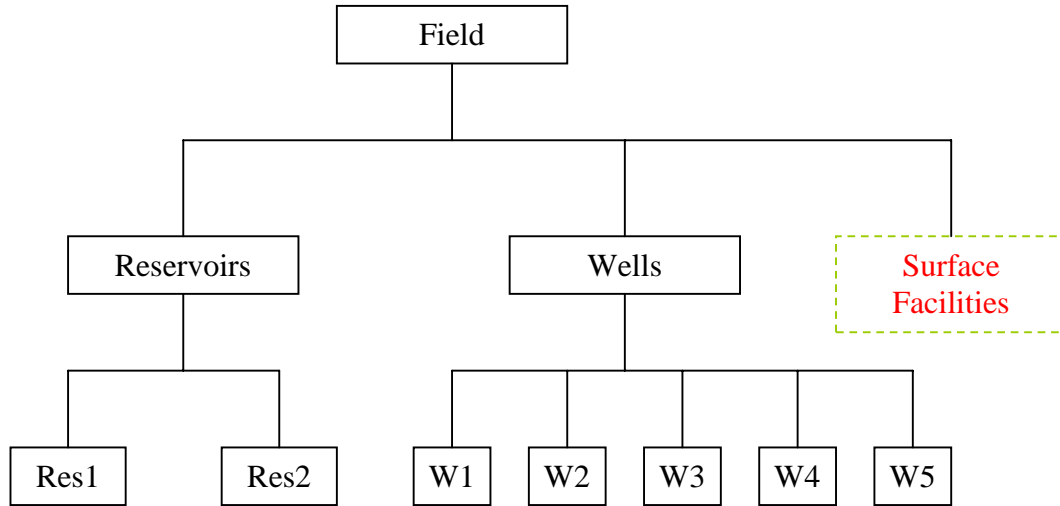
Modular object-oriented design is used for GPRS. All of the code is written in standard C++. While the design will help new users in getting started with GPRS, it will still require some effort to become fully familiar with this code. GPRS is a big and complex program, currently it includes about 120 files, which are separated into 11 sub-directories. In order to facilitate the use of GPRS, good documentation is essential (Cao, 2002).

To fully understand GPRS, users should first learn the basic structure of GPRS, which will be introduced in this part. After that, a good understanding of the basic algorithms and techniques used in GPRS (refer to Chapter 2 for details) is necessary. Finally GPRS has a lot of internal comments, which should be very helpful in understanding individual classes and methods.

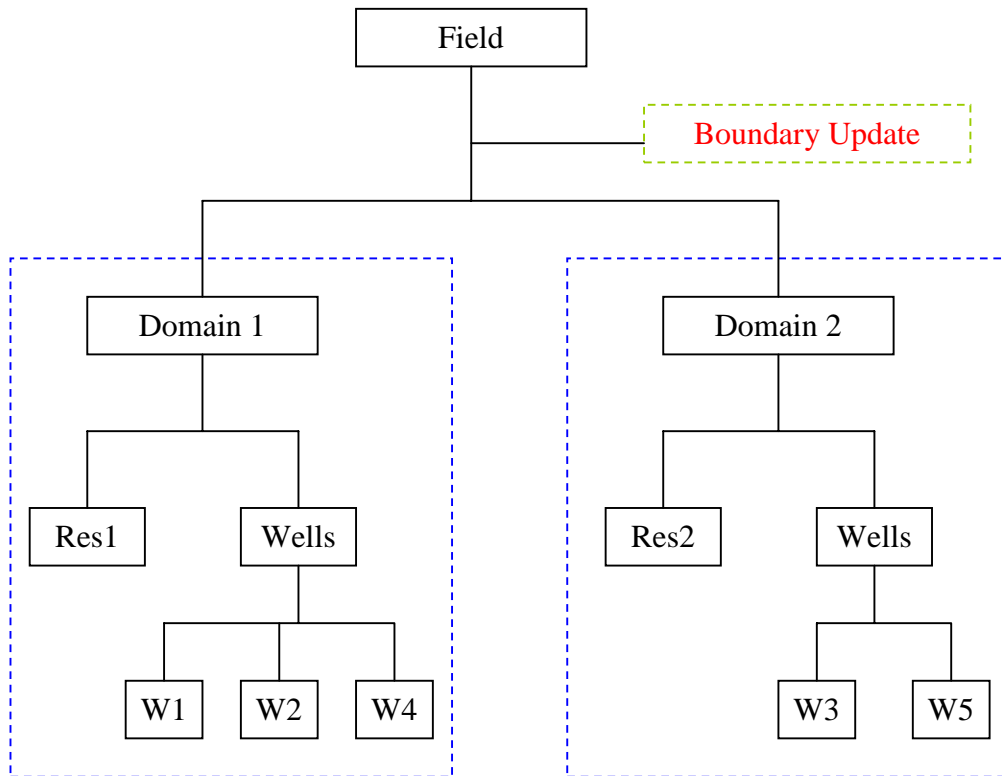
In this part, we will first explore the system models for GPRS level by level, from top down, after that we will discuss its flow sequence. We will also list the directories, files and libraries of GPRS, and explain their use. Finally, for end users, we will explain the structure of input files.

### **A.1 The System Model**

The system model shows the basic classes and their relations, and it is very helpful for the understanding of the structure of GPRS. Since GPRS is a complex code, it will be difficult to explain the system model using only one figure. To make it easier to understand, we will show the system models level by level, and explain each level in detail. Figure A.1 shows the physical structure of an oil field, which consists of several reservoirs, lots of wells, and some surface facilities. From a computational point of view, we can recast the system in Figure A.1 as the one shown in Figure A.2, which is actually the top level (field level) system model for GPRS. Currently, surface facility modeling is not included in GPRS, so it is left out in Figure A.2.



**Figure A.1** Structure of an oil field

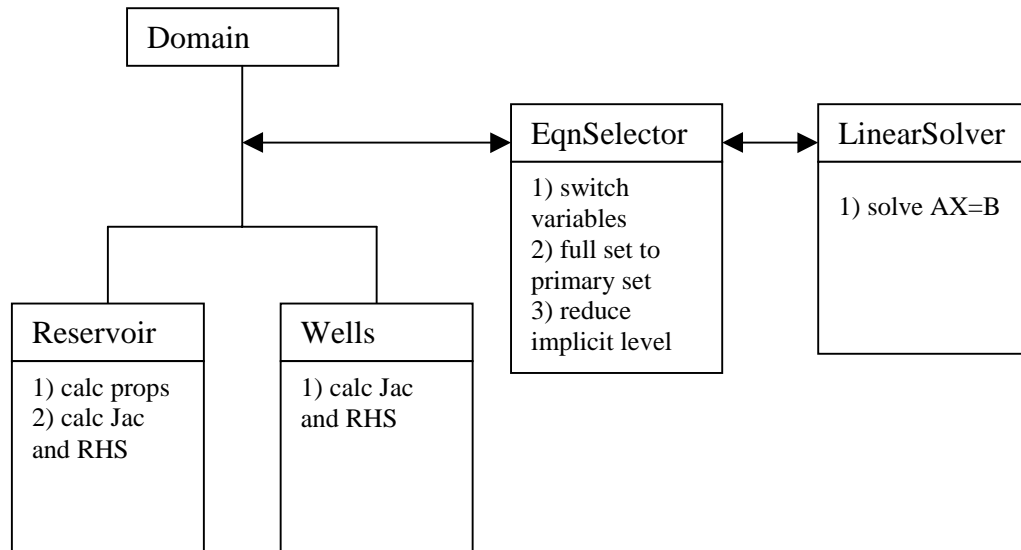


**Figure A.2** Field level system model



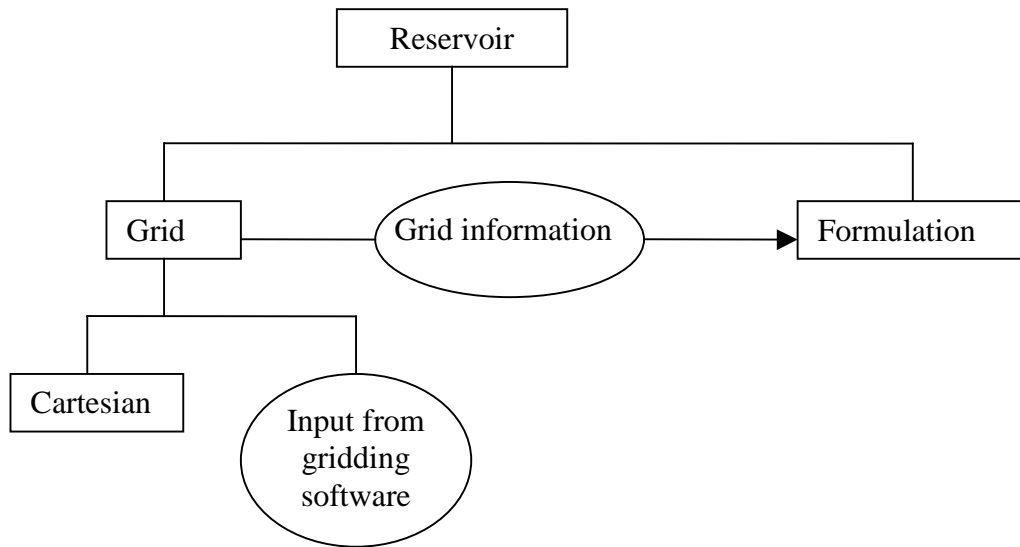
In Figure A.2, each oil field is split into several domains, and each domain includes one reservoir and all of the wells that are in that reservoir. Multiple domains are best handled by parallel computing methods. This can be done by first assigning each domain to a different CPU, then solving each of them separately, after that, the boundaries between domains need to be updated (passing data between different CPUs). This process has to be repeated until the whole field has converged. If there are more domains than CPUs, multiple domains can be assigned to the same CPU. Currently, GPRS is only used for sequential simulation. In order to upgrade it for parallel computation, some work is needed to implement the boundary update.

The domain level system model is shown in Figure A.3. As we know from Figure A.2, each domain includes a reservoir and several wells. Besides that, each domain also has an equation selector and a linear solver. Basically, in GPRS, the Jacobian matrices are calculated separately for the reservoir part and for the well part, the equation selector is used to recast the Jacobian with desired variables and implicit levels. After that the Jacobian matrices are passed to the linear solver and pieced together there for the linear solve. Finally, the solution goes back in the opposite direction.

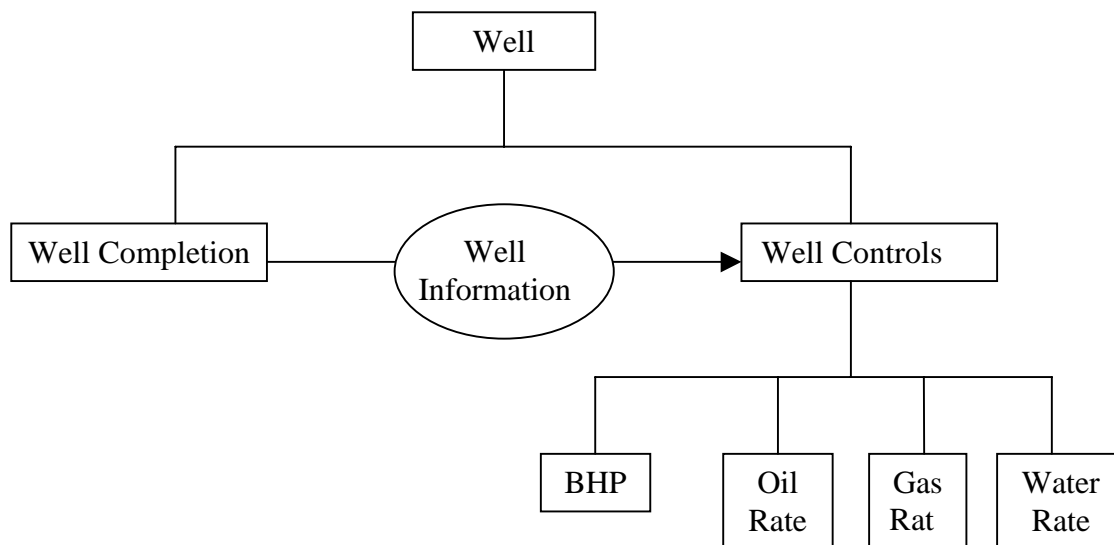


**Figure A.3** Domain level system model

As mentioned above, the Jacobian matrices are calculated separately for the reservoir part and for the well part. Figure A.4 and Figure A.5 show the system model for each of them. From Figure A.4, we can see that, each reservoir includes a grid and a formulation, grid part generates the grid information and passes it to the formulation part. The formulation part calculates the gridblock properties and builds the reservoir part of the Jacobian matrix and the RHS. In GPRS, the grid information is either internally generated (currently only for Cartesian grid), or read in from the output of a gridding software. The system model for the well part has a similar structure (as shown in Figure A.5). In this part, well information is generated from the well completion data and passed to the well control module, which calculates the well part of the Jacobian matrix and the RHS. Currently, four types of well controls are implemented, and they are BHP, oil flow rate, gas flow rate and water flow rate control.

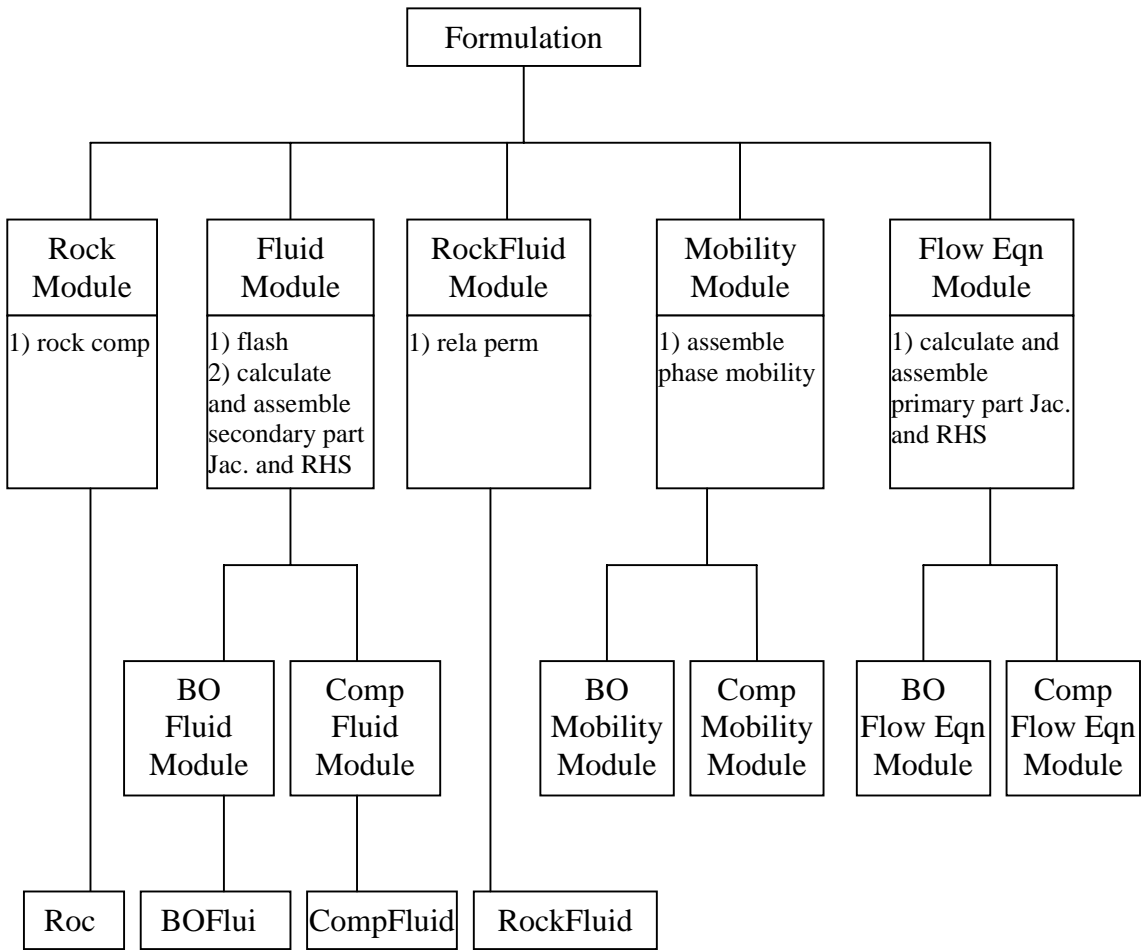


**Figure A.4** Reservoir level system model



**Figure A.5** Well level system model

Figure A.6 shows the last system model. The entire reservoir related calculations are carried out here. The calculations are organized using five mathematical modules: one for rock, one for fluid, one for rock/fluid, one for phase mobility and the last one for flow equations. Most of the gridblock properties are calculated in the first four modules, and the reservoir part of the Jacobian matrix and the RHS are calculated and assembled in the last module. If necessary, each module can have multiple subclasses for different models. For example the fluid part has a black-oil and a compositional module. If a module has connections to the physical world, such as the rock module, it will have a pointer to the corresponding physical object. All of the mathematical modules are organized in a multi-level inheritance structure, they all share the same public interface (methods), and dynamic binding is used to determine the actual objects and methods required during a run.



**Figure A.6** Formulation level system model

## A.2 Flow Sequence

Having shown the system models and classes used in GPRS, we will now review the flow sequence of GPRS. The essential steps in a simulator are given below (Aziz, 1996):

1. Read input data (problem definition)
2. Initialize (allocate data and set initial conditions)
3. Start time step calculations
  - Initialize with old timestep data
  - Start the Newton iteration
    - Calculate gridblock properties
    - Linearize (calculate and assemble Jacobian and RHS)
    - Solve the linear system
    - Perform Newton update
    - Check convergence, do another Newton iteration if necessary
4. Print and plot results at appropriate times
5. Increment time and go to Step 3 if ending conditions are not reached
6. End when run is complete

The most important step is Step 3, which accounts for over 95% of the simulation time.

The implementation of Step 3 in GPRS is as following:

```
calcInitProps();           // record old timestep values
while(true) {              // start Newton iterations
    calcProps();           // calculate gridblock properties
    linearize();           // calculate and assemble Jacobian and RHS

    convF = convergeF();    // check residual convergence
    mLinearSolver->solve()  // solve the linear system
    newtonUpdate();        // newton update
    convY = convergeDY();   // check variable change convergence
    if(convF && convY) break; // converged
}                          // end of Newton iterations
```

Within this step, four methods handle most of the calculations for the reservoir part. They are: *calcInitProp()*, *calcProp()*, *linearize()* and *newtonUpdate()*. They have similar structure as that of *linearize()*:

```
void Reservoir::linearize()
{
    // --- each mathematical module linearizes ---
    list<mthModelBase*>::iterator iter1 = mMthModels.begin();
    while (iter1 != mMthModels.end()) {
        (*iter1)->linearize(dt);
        iter1++;
    }

    // --- each well linearizes ---
    vector<Well*>::iterator iter2 = mWells.begin();
    while (iter2 != mWells.end()) {
        if((*iter2)->wellOpen()) (*iter2)->linearize();
        iter2++;
    }
}
```

Basically, most of the reservoir calculations are performed by these mathematical modules. Since all of these modules are inherited from the same base class, they share the same public interface, we can just loop through them, and let each of them do its work in turn. The only thing we need to take care of is to program the correct public methods (above four methods) for each module. Using this approach, each Newton iteration becomes simple and flexible. If at any time, a new module is needed, we only need to create it, add it to the list of modules, and implement the above four public methods. If any of the above four methods is not necessary for a module, then it will be empty for that module.

### A.3 Sub-directories and Files

Besides the main program (*Main.cpp*) which is located in the base directory, there are 11 sub-directories. Here we will introduce each of them and the files contained in them.

- **field**

<i>Field.h/cpp:</i>	field class
---------------------	-------------
  
- **fluid** Physical fluid

<i>Fluid.h/cpp:</i>	fluid base class
<i>Phase.h/cpp:</i>	phase base class
<i>Comp.h/cpp:</i>	component class
<i>BOFluid.h/cpp:</i>	black-oil fluid class, subclass of <i>Fluid</i>
<i>CompFluid.h/cpp:</i>	compositional fluid class, subclass of <i>Fluid</i>
<i>BOPhase.h/cpp:</i>	black-oil phase class, subclass of <i>Phase</i>
<i>HCPPhase.h/cpp:</i>	compositional hydrocarbon phase class, subclass of <i>Phase</i>
 <i>WaterPhase.h/cpp:</i>	 compositional water phase class, subclass of <i>Phase</i>
  
- **grid**

<i>Grid.h/cpp:</i>	Cartesian grid class
--------------------	----------------------
  
- **math** Formulation modules

<i>mathModelBase.h/cpp:</i>	base class for all modules
<i>mathRockCompModel.h/cpp:</i>	rock compressibility module, subclass of <i>mathModelBase</i>
 <i>mathFluidModelBase.h/cpp:</i>	 base fluid module, subclass of <i>mathModelBase</i>
<i>mathBOFluidModel.h/cpp:</i>	black-oil fluid module, subclass of <i>mathFluidModelbase</i>
 <i>mathCompFluidModel.h/cpp:</i>	 compositional fluid module, subclass of <i>mathFluidModelbase</i>
 <i>mathRockFluidModel.h/cpp:</i>	 rock fluid module, subclass of <i>mathModelBase</i>
<i>mathMobilityModel.h/cpp:</i>	phase mobility module, subclass of <i>mathModelBase</i>
<i>mathCompMobilityModel.h/cpp:</i>	compositional phase mobility module, subclass of <i>mathMobilityModel</i>

- methFlowEqnModel.h/cpp*: base flow equation module, subclass of *methModelbase*

*methBOFlowEqnModel.h/cpp*: black-oil flow equation module, subclass of *methFlowEqnModel*

*methCompFlowEqnModel.h/cpp*: compositional flow equation module, subclass of *methFlowEqnModel*
- **res** Reservoir

*Reservoir.h/cpp*: reservoir class

*EqnSelector.h/cpp*: equation selector base class

*EqnSelectorCoats.h/cpp*: variable Type A (Coats' model) equation selector class, subclass of *EqnSelector*

*EqnSelectorYoung.h/cpp*: variable Type B (Young and Stephenson's model) equation selector class, subclass of *EqnSelector*
- **rock** Physical rock

*Rock.h/cpp*: physical rock class
- **rockfluid** Physical rock fluid (relative permeability)

*RockFluid.h/cpp*: physical rock fluid base class

*RockFluid2P.h/cpp*: physical two-phase rock fluid class, subclass of *RockFluid*

*RockFluid3P.h/cpp*: physical three-phase rock fluid class, subclass of *RockFluid*
- **sim** Simulation constant definition

*Comlibs.h*: define common included files

*EnumDef.h*: define structures, Enumerates and constants used in GPRS



- **solver** Linear solvers

<i>LinearSolverBase.h/cpp:</i>	linear solver base class
<i>LapackFullSolver.h/cpp:</i>	full matrix solver from LAPACK, subclass of <i>LinearSolverBase</i>
<i>LapackBandSolver.h/cpp:</i>	band matrix solver from LAPACK, subclass of <i>LinearSolverBase</i>
<i>ImlGMRESSolver.h/cpp:</i>	GMRES solver from IML (Iterative Math Library), subclass of <i>LinearSolverBase</i>
<i>BlitzSolver.h/cpp:</i>	solver from BlitzPak, subclass of <i>LinearSolverBase</i>
<i>PreconditionerBase.h/cpp:</i>	preconditioner base class
<i>DiagPre.h/cpp:</i>	diagonal scaling preconditioner, subclass of <i>PreconditionerBase</i>
<i>BlockDiagPre.h/cpp:</i>	block diagonal scaling preconditioner, subclass of <i>PreconditionerBase</i>
<i>ILUPre.h/cpp:</i>	Incomplete LU decomposition (ILU0) preconditioner from SparseLib++, subclass of <i>PreconditionerBase</i>
<i>AMGPre.h/cpp:</i>	Algebraic Multi-Grid (AMG) preconditioner from GMD software, subclass of <i>PreconditionerBase</i>
<i>TrueIMPESPre.h/cpp:</i>	Constraint Pressure Residual (CPR) preconditioner, true IMPES is used to generate the pressure equation, subclass of <i>PreconditionerBase</i>
<i>QuasiIMPESPre.h/cpp:</i>	Constraint Pressure Residual (CPR) preconditioner, quasi IMPES is used to generate the pressure equation, subclass of <i>PreconditionerBase</i>

- **util** Utilities

<i>FileIO.h/cpp:</i>	file input and output class
<i>Table.h/cpp:</i>	table read in and look up class
<i>Strlib.h/cpp:</i>	string operation class

<i>F2Cwrapper.h/cpp:</i>	Fortran to C++ wrapper class for Fortran subroutines
<i>MyMatrix.h/cpp:</i>	matrix operations class, suitable for small matrix
<i>DataVector.h/cpp:</i>	vector of data base class
<i>IntDataVector.h/cpp:</i>	vector of integer data class, base class of <i>DataVector</i>
<i>DFDataVector.h/cpp:</i>	vector of double data class, base class of <i>DataVector</i>
<i>ConnDataVector.h/cpp:</i>	vector of connection data class, base class of <i>DataVector</i>
<i>DataPool.h/cpp:</i>	data pool class, store and provide access to all <i>DataVector</i> objects

- **well**

<i>Well.h/cpp:</i>	well class
<i>WellCompl.h.cpp:</i>	well completion class
<i>WellCont.h/cpp:</i>	well control base class
<i>WellBHPCont.h/cpp:</i>	well bottom hole pressure (BHP) control class, subclass of <i>WellCont</i>
<i>WellORateCont.h/cpp:</i>	well constant oil flow rate control class, subclass of <i>WellCont</i>
<i>WellGRateCont.h/cpp:</i>	well constant gas flow rate control class, subclass of <i>WellCont</i>
<i>WellWRateCont.h/cpp:</i>	well constant water flow rate control class, subclass of <i>WellCont</i>

#### A.4 Libraries

GPRS uses several public domain libraries, and most of them are used in the linear solver part. They are discussed below:

- **STL** (Standard Template Library) This library is a popular library that is treated by many as a standard C++ library. In GPRS, we use the “List” and “Vector” template classes from this library.
- **LAPACK** (Linear Algebra Package) (Anderson, 1999) This is a standard linear algebra package, which can be freely downloaded from [www.netlib.org](http://www.netlib.org). It has a Fortran version and a C version. Currently we are using the Fortran version, and wrappers are used to interface it with the main C++ code. For most SGI machines, LAPACK is already installed. For PC's, we use the MKL (Math Kernel Library) from INTEL, which includes LAPACK. In GPRS, we use the direct solvers (dgesv and dgbv) from LAPACK.
- **BlitzPak** (Landmark, 1998) This is a commercial solver package from Landmark Graphic Corporation. GPRS interfaces with it, and uses it as a structured grid solver.
- **AMG** This is an Algebraic Multi-Grid (AMG) solver from GMD software (Ritzdorf, 1991), which can be freely downloaded from <http://www.mgnet.org/mgnet-codes-gmd.html>. It is only used as a preconditioner for the pressure system in GPRS.
- **SparseLib++** (Pozo et al., 1996) This is a sparse matrices library (freely downloadable from <http://math.nist.gov/sparselib++/>) which includes some basic representations for sparse matrix, such as compressed row format, and some basic preconditioners, such as ILU0 (Incomplete LU decomposition without fill-in). In GPRS, we use it for sparse matrix representation and ILU0 preconditioner.
- **IML** (Iterative Math Library) (Dongarra et al., 1996) This is an iterative solver package (freely downloadable from <http://math.nist.gov/iml++/>), which includes GMRES, CG (Conjugate Gradient), etc. In GPRS, we only use its GMRES solver.

## A.5 Input Files

The main input file for GPRS is normally called “gprs.in”. Within it, three files are included (using “INCLUDE” keyword), one for the reservoir data, one for the wells data, and one for the control data. Contents of each of these files are describes below:

- Reservoir input file
  - grid data (grid geometry and properties)
  - fluid data, black-oil or compositional (phase and component properties)
  - phase component relation data (existence of each component in each phase)
  - rock fluid data (relative permeability and capillary pressure)
  - rock data (rock compressibility)
  - initial equilibrium data (WOC, GOC, initial pressure and overall composition, etc)
  
- Well input file
  - well specification (well name, well group name, which reservoir it belongs to, producer or injector)
  - well completion data (well block indexes, and WI for each of them)
  - well control data (all available well controls for this well)
  
- Control file
  - Timestep control (initial, minimum and maximum timestep size, and total simulation time)
  - Newton iteration number control (minimum, maximum and fixed number of Newton iterations for each timestep)
  - Timestep increase control (desired variable changes and increasing factor)
  - Newton iteration convergence control (convergence criteria for variable changes and residuals)
  - Formulation control (which type of variables to use, how many implicit levels, and what are they)
  - Linear solver control (which linear solver to use, which preconditioner to use, convergence tolerance and maximum number of iterations)

For information about the keywords and data format used by the input files, refer to the GPRS user's manual, which is provided as part of the GPRS package (Cao, 2002).

## Appendix B: Black-Oil Simulation Using Compositional Formulation

The black-oil model is a special case of the general compositional model, where flash calculations are reduced to explicit relations, and the primary equations can be directly expressed as functions of only primary variables. Hence a general compositional simulator should be able to perform black-oil simulation given the correct input, such as formation volume factors and gas solubility. Considering a three-phase (oil, gas and water) black-oil system, assuming gas component can exist in both the gas and the oil phases, and oil and water components can only exist in their own phases. The general compositional formulation can be written as

$$\frac{\partial}{\partial t} [V\phi \sum_p (S_p \rho_p x_{cp})] = \sum_l [T \sum_p (\frac{k_{rp}}{\mu_p} \rho_p x_{cp} \Delta\Phi_p)]_l - \sum_w [WI \sum_p (\frac{k_{rp}}{\mu_p} \rho_p x_{cp} (p_p - p^w))] \quad (B.1)$$

For the black-oil model, the phase properties and component mole fractions can be calculated by the relations shown below (Aziz, 1996):

- $\rho_p = \rho_p(p)$ , phase densities are only functions of pressure, and they are calculated by

$$\rho_w = \frac{\rho_{\bar{w}}}{B_w} \quad (B.2)$$

$$\rho_g = \frac{\rho_{\bar{g}}}{B_g} \quad (B.3)$$

$$\rho_o = \frac{\rho_{\bar{o}} + \rho_{\bar{g}} R_{\bar{g},o}}{B_o} \quad (B.4)$$

where,  $\rho_{\bar{w}}$ ,  $\rho_{\bar{g}}$  and  $\rho_{\bar{o}}$  are the component densities (phases at standard conditions), which are input constants.  $B_w$ ,  $B_g$  and  $B_o$  are the phase formation volume factors, which are functions of pressure and normally given as  $B$  vs.  $p$  tables.  $R_{\bar{g},o}$  is the gas component solubility ratio in the oil phase, which is also a sole function of pressure and given as a  $R$  vs.  $p$  table.

- $\mu_p = \mu_p(p)$ , phase viscosities are also only functions of pressure, normally a  $\mu_p$  vs.  $p$  table is given, and table lookup is used to get phase viscosities at specific pressure.
- $X_{cp} = X_{cp}(p)$ , component mole fractions are not independent variables, they are either constants or only functions of pressure, and can be calculated by the following relations:

$$X_{\bar{o},w} = X_{\bar{w},o} = X_{\bar{g},w} = X_{\bar{w},g} = X_{\bar{o},g} = 0 \quad (\text{B.5})$$

$$X_{\bar{w},w} = X_{\bar{g},g} = 1 \quad (\text{B.6})$$

$$X_{\bar{g},o} = \frac{\rho_{\bar{g}} R_{\bar{g},o}}{\rho_{\bar{o}} + \rho_{\bar{g}} R_{\bar{g},o}} \quad (\text{B.7})$$

$$X_{\bar{o},o} = \frac{\rho_{\bar{o}}}{\rho_{\bar{o}} + \rho_{\bar{g}} R_{\bar{g},o}} \quad (\text{B.8})$$

An even simpler method is to directly calculate  $\rho_p X_{c,p}$  from (Aziz, 1996)

$$\rho_p X_{c,p} = \rho_{\bar{c}} \frac{R_{c,p}}{B_p} \quad (\text{B.9})$$

## Appendix C Flash Calculation (Walas, 1985)

As mentioned in Section 2.6, the Cubic equation of state can be written as

$$Z^3 + sZ^2 + qZ + r = 0 \quad (\text{C.1})$$

$$\text{where, } \begin{cases} s = (u-1)B - 1 \\ q = A + (w-u)B^2 - uB \\ r = -AB - wB^2 - wB^3 \end{cases} \quad (\text{C.2})$$

Different Cubic equations of state have different  $u$ ,  $w$  values, and they also use different relations to calculate parameters  $a_i$  and  $b_i$  of each component:

- Redlich-Kwong,  $u=1, w=0, b_i = \frac{0.08664RT_{C,i}}{P_{C,i}}, a_i = \frac{0.42748R^2T_{C,i}^{2.5}}{P_{C,i}T^{0.5}}$
- Soave-Redlich-Kwong,  $u=1, w=0, b_i = \frac{0.08664RT_{C,i}}{P_{C,i}}, a_i = \frac{0.42748R^2T_{C,i}^2}{P_{C,i}}[1 + f_w(1 - (\frac{T}{T_{C,i}})^{0.5})^2]$   
where  $f_w = 0.48 + 1.574\omega_i - 0.176\omega_i^2$
- Peng-Robinson,  $u=2, w=-1, b_i = \frac{0.07780RT_{C,i}}{P_{C,i}}, a_i = \frac{0.45724R^2T_{C,i}^2}{P_{C,i}}[1 + f_w(1 - (\frac{T}{T_{C,i}})^{0.5})^2]$   
where  $f_w = 0.37464 + 1.54226\omega_i - 0.26992\omega_i^2$

where  $T_{C,i}$  and  $P_{C,i}$  are the critical temperature and critical pressure of each component,  $T$  is the reservoir temperature, and  $\omega_i$  is the acentric factor of each component.

For the Cubic equation of state, the component fugacities and the phase properties can be calculated from the following 5 steps (Walas, 1985):

1. Calculate phase parameters and their derivatives:

$$\begin{cases} a = \sum_{j=1}^{n_h} \sum_{i=1}^{n_h} X_i X_j (1 - k_{i,j}) \sqrt{a_i a_j} \\ b = \sum_{i=1}^{n_h} X_i b_i \end{cases}, \quad (\text{C.3})$$

$$\text{and} \quad \begin{cases} \frac{\partial a}{\partial X_i} = 2\sqrt{a_i} \sum_{j=1}^{n_h} X_j (1 - k_{i,j}) \sqrt{a_j} \\ \frac{\partial b}{\partial X_i} = b_i \end{cases} \quad (\text{C.4})$$

where  $k_{ij}$  is the binary interaction parameter.

$$\begin{cases} A = \frac{ap}{R^2 T^2} \\ B = \frac{bp}{RT} \end{cases} \quad (\text{C.5})$$

$$\text{and} \quad \begin{cases} \frac{\partial A}{\partial p} = \frac{A}{p} \\ \frac{\partial B}{\partial p} = \frac{B}{p} \end{cases}, \quad \begin{cases} \frac{\partial A}{\partial X_i} = \frac{A}{a} \frac{\partial a}{\partial X_i} \\ \frac{\partial B}{\partial X_i} = \frac{B}{b} \frac{\partial b}{\partial X_i} \end{cases} \quad (\text{C.6})$$

2. Calculate compressibility factor  $Z$  and its derivatives:

$$Z^3 + sZ^2 + qZ + r = 0 \quad (\text{C.7})$$

$$\text{and} \quad \begin{cases} \frac{\partial Z}{\partial p} = -\frac{\frac{\partial s}{\partial p} Z^2 + \frac{\partial q}{\partial p} Z + \frac{\partial r}{\partial p}}{3Z^2 + 2sZ + q} \\ \frac{\partial Z}{\partial X_i} = -\frac{\frac{\partial s}{\partial X_i} Z^2 + \frac{\partial q}{\partial X_i} Z + \frac{\partial r}{\partial X_i}}{3Z^2 + 2sZ + q} \end{cases} \quad (\text{C.8})$$

$$\text{where} \quad \begin{cases} s = (u - 1)B - 1 \\ q = A + (w - u)B^2 - uB \\ r = -AB - wB^2 - wB^3 \end{cases} \quad (\text{C.9})$$

3. If necessary, perform volume translation (e.g. Deo et al., 1989):

$$Z = Z - \frac{p}{RT} \sum_{i=1}^{n_h} X_i c_i \quad (\text{C.10})$$



$$\text{and} \quad \begin{cases} \frac{\partial Z}{\partial p} = \frac{\partial Z}{\partial p} - \frac{1}{RT} \sum_{i=1}^{n_h} X_i c_i \\ \frac{\partial Z}{\partial X_i} = \frac{\partial Z}{\partial X_i} - \frac{p}{RT} c_i \end{cases} \quad (\text{C.11})$$

where  $c_i = S_i b_i$

4. Calculate phase properties (phase densities and their derivatives):

$$\rho = \frac{P}{ZRT} \quad (\text{C.12})$$

$$\text{and} \quad \begin{cases} \frac{\partial \rho}{\partial p} = \frac{\rho}{p} - \frac{\rho}{Z} \frac{\partial Z}{\partial p} \\ \frac{\partial \rho}{\partial X_i} = -\frac{\rho}{Z} \frac{\partial Z}{\partial X_i} \end{cases} \quad (\text{C.13})$$

5. Calculate component fugacities and their derivatives:

$$f_i = p X_i e^{\Phi_i}, \quad (\text{C.14})$$

$$\begin{aligned} \Phi_i &= \frac{b_i}{b} (Z - 1) - \ln(Z - B) \\ &\quad - \frac{A}{B\sqrt{u^2 - 4w}} \left( \frac{b_i}{b} - \frac{1}{a} \frac{\partial a}{\partial X_i} \right) \ln \left[ \frac{2Z + B(u + \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right] \end{aligned} \quad (\text{C.15})$$

$$\frac{\partial f_i}{\partial p} = \frac{f_i}{p} + f_i \frac{\partial \Phi_i}{\partial p} \quad (\text{C.16})$$

$$\begin{aligned} \frac{\partial \Phi_i}{\partial p} &= \frac{b_i}{b} \frac{\partial Z}{\partial p} - \frac{\frac{\partial Z}{\partial p} - \frac{\partial B}{\partial p}}{Z - B} \\ &\quad + \left( \frac{b_i}{b} - \frac{1}{a} \frac{\partial a}{\partial X_i} \right) \frac{1}{\sqrt{u^2 - 4w}} \left[ \left( \frac{1}{B} \frac{\partial A}{\partial p} - \frac{A}{B^2} \frac{\partial B}{\partial p} \right) \ln \left( \frac{2Z + B(u + \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right) \right. \\ &\quad \left. + \frac{A}{B} \left( \frac{2 \frac{\partial Z}{\partial p} + \frac{\partial B}{\partial p} (u + \sqrt{u^2 - 4w})}{2Z + B(u + \sqrt{u^2 - 4w})} - \frac{2 \frac{\partial Z}{\partial p} + \frac{\partial B}{\partial p} (u - \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right) \right] \end{aligned} \quad (\text{C.17})$$

$$\frac{\partial f_i}{\partial X_j} = \frac{f_i}{X_i} \delta_{ij} + f_i \frac{\partial \Phi_i}{\partial X_j} \quad (\text{C.18})$$

$$\begin{aligned} \frac{\partial \Phi_i}{\partial X_j} = & \left[ \frac{b_i}{b} \frac{\partial Z}{\partial X_j} - \frac{b_i}{b^2} (Z-1) \frac{\partial b}{\partial X_j} \right] - \frac{\frac{\partial Z}{\partial X_j} - \frac{\partial B}{\partial X_j}}{Z-B} + \left( \frac{b_i}{b} - \frac{1}{a} \frac{\partial a}{\partial X_i} \right) \frac{1}{\sqrt{u^2 - 4w}} \\ & \left[ \left( \frac{1}{B} \frac{\partial A}{\partial X_j} - \frac{A}{B^2} \frac{\partial B}{\partial X_j} \right) \ln \left( \frac{2Z + B(u + \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right) \right. \\ & + \frac{A}{B} \left( \frac{2 \frac{\partial Z}{\partial X_j} + \frac{\partial B}{\partial X_j} (u + \sqrt{u^2 - 4w})}{2Z + B(u + \sqrt{u^2 - 4w})} - \frac{2 \frac{\partial Z}{\partial X_j} + \frac{\partial B}{\partial X_j} (u - \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right) \Big] \\ & + \frac{A}{B \sqrt{u^2 - 4w}} \ln \left( \frac{2Z + B(u + \sqrt{u^2 - 4w})}{2Z + B(u - \sqrt{u^2 - 4w})} \right) \left[ -\frac{b_i}{b^2} \frac{\partial b}{\partial X_j} + \frac{1}{a^2} \frac{\partial a}{\partial X_i} \frac{\partial a}{\partial X_j} \right. \\ & \quad \left. \left. - \frac{2}{a} (1 - k_{ij}) \sqrt{a_i a_j} \right] \right] \end{aligned} \quad (\text{C.19})$$