The Thesis Committee for Youguang Chen
certifies that this is the approved version of the following thiesis:

A New Upscaling Method for Flow Simulation of Fractured Systems

APPROVED BY

SUPERVISING COMMITTEE:

Kamy Sepehrnoori, Supervisor

Wei Yu

A New Upscaling Method for Flow Simulation of Fractured Systems

by

Youguang Chen

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2018

Dedication

To my beloved parents Xiaoqian and Hongtao, for their love and support.

Acknowledgments

I would like to thank my advisor Prof. Kamy Sepehrnoori for his guidance and consistent support throughout my master study. This work would not have been possible without his guidance.

I would like to thank Dr. Yifei Xu for his helpful discussions throughout this work. I would also like to thank all the members in the reservoir simulation group directed by Prof. Kamy Sepehrnoori. Thanks to all the faculty and staff at PGE for providing the excellent environment for learning and conducting research in peteroleum engineering.

I appreciate the financial support from members of Reservoir Simulation Joint Industry Project (RSJIP) at the Center for Petroleum and Geosystems Engineering at The University of Texas at Austin.

Thanks to my parents for always giving me the best chance to be sucessful they could provide. Also many thanks to my friends, for all the happiness they share with me and their encouragements for me to pursue my dream in academia.

A New Upscaling Method for Flow Simulation of Fractured Systems

Youguang Chen, MSE
The University of Texas at Austin, 2018

Supervisor: Kamy Sepehrnoori

Fractured reservoirs have gained continuous attention in oil and gas industry since a huge amount of reserves are stored in such reservoirs. Fractures add complexity in reservoir models and thus have potentially large effects on the reservoir simulation results. Though a lot of fine scale fracture models for reservoir simulation have been developed to capture the fracture effects, they are generally complicated and time consuming for the cases with large number of fractures and problems (for example, some inverse problems and optimization problems) where lots of forward simulations are required.

Upscaling is a method to fasten the flow simulations by constructing reduced models in coarse scale to approximate the original fine scale models. It is important to construct coarse models in a proper way since the approximated models will generate errors as opposed to the fine scale models. Therefore, a new upscaling method is proposed in this work to capture the effects of fractures in fractured reservoir.

First, two hypothetical flow problems are presented to provide pressure solutions for calculation of parameters in coarse models. Unsteady state method, one of these two flow problems, is firstly introduced in this work to obtain reasonable pressure solutions for reservoirs without source term. Second, we developed two partitioning methods to associate coarse grids with fine grids. Since these two partitioning approaches are suitable for different types of fracture networks, we proposed a multi-level partitioning method that is a general approach and could capture fracture effects of different fracture patterns. Third, we developed an efficient time-stepping algorithm for the unsteady state problem to reduce the computational efforts of upscaling process.

The applicability of the new upsclaing methodology is verified from numerical tests of different types of reservoirs with different fracture patterns and well configurations. Errors of pressure solution, oil saturation, and production solutions are generally limited below 5% in coarse scale. Furthermore, speedup of simulation is significant in all of the presented numerical tests.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Flow simulation for fractured porous media is highly important since nearly 60% of hydrocarbon reserves are in fractured reservoirs. Besides oil production, fractures are also included in many other geological systems, such as heat extraction in geothermal reservoirs [22, 52] and contaminant transport in groundwater flow [9, 30]. The high contrast in permeability between fractures and matrix makes the fine scale simulation expensive and challenging. Difficulties of flow simulation in fractured systems also stem from different scales, conductivities (high conductivity as hydraulic fractures or low conductivity like natural fractures), patterns (sparsely distributed fractures or densely interconnected fractures), and complex geometries of fractures.

Though fine scale fracture modeling has been largely developed to capture flow behaviors associated with fractures, it requires too many extra fracture grids for geological formations with numerous number of natural fractures (see Figure 1.1). Different scales and complex geometries of fractures may also add the complexity of the modeling and computational efforts. Moreover, today's iterative modeling workflows require many simulations of the forward

problems, for example, in some inverse problems, uncertainty quantification problems and optimization problems. Fine scale simulation may not be applicable under such conditions, and approximating models are therefore needed to lessen the computational efforts.

Upscaling is a way to construct approximate models by coarsening the fine grids and generating new parameters for coarse grids to solve the time-consuming issue in fine scale simulations. But conventional flow-based upscaling approaches have some limitations for the application in fractured systems. Firstly, hypothetical flow problems proposed for this upscaling method are all steady state problems, which may not suitable for the no-source systems (such as the reservoir with no-flow boundaries and only production wells). Besides, although aggregation methods for matrix grids have been studied, research on how to aggregate fine scale fracture grids to coarse scale is limited. Since fracture has relatively high permeability than its neighboring matrix, how to aggregate those fractures may have large effects on the final flow simulation results.

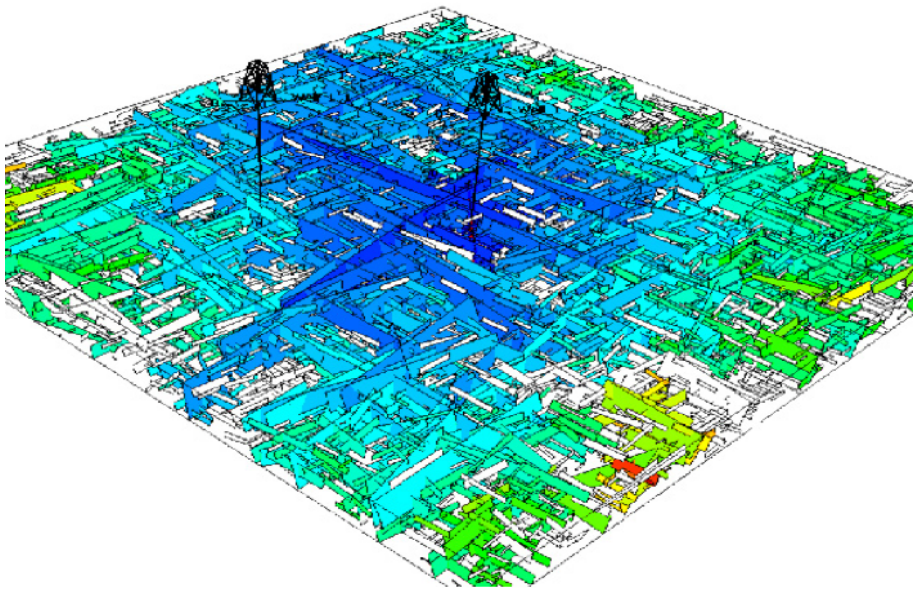


Figure 1.1: An example of reservoir with numerous natural fractures (source: https://www.golder.com/).

1.2 Objectives

Base on the problems described above, the objectives of this research are

- Form a non-intrusive upscaling framework for fracture reservoir systems to generate reduced order models for the coarse scale modeling, which could still generate relatively accurate results compared to the fine scale simulation.
- Apply the embedded discrete fracture model (EDFM) to form fine scale models to speedup the upscaling process.

- Establish new upscaling methods appropriate to different fracture patterns, different reservoir conditions and well configuration by extending flow-based upscaling approaches. Besides, using numerical tests in different scenarios to verify the applicability, accuracy and efficiency of the proposed upscaling method.

1.3 Brief description of chapters

In Chapter 2, a literature review of reservoir simulation methods for fractured systems and and some reduced models for such systems are presented. In Chapter 3, the governing equations and fines scale discretized system is described. Then the upscaling methodology for flow simulation in fractured reservoirs is introduced in Chapter 4. Numerical tests are presented in Chapter 5 to verify the proposed upscaling method. Finally, conclusions and recommendations for future work are presented in Chapter 6.

Chapter 2

Literature Review

2.1 Fine scale flow simulation models of fractured systems

Much efforts have been devoted to the development of fine scale flow simulation models associated with fractures in porous media. Although different methods vary between each other, most of them can be categorized into three models: dual-porosity /dual-permeability models (DP/DK model), discrete fracture models (DFM) and embedded discrete fracture model (EDFM).

2.1.1 Dual-porosity and dual-permeability models

Dual-porosity and dual-permeability model (DPDK), firstly proposed in [6] and then developed by [54], has been used for a few decades. Flow transfer function are defined between fractures and matrix in such models, and a lot of studies have been conducted to improve their applicabilities and efficiencies in [36, 48, 39, 47]. Due to various assumptions are always related to the transfer function, the dual-porosity and dual-permeability models , however, are not appropriate for cases with disconnected fractures in [36] or with strong gravity and viscous forces in [32].

2.1.2 Discrete fracture models

Due to the large error introduced in DPDK models, discrete Fracture Model (DFM) was developed, which typically deploys unstructured grids to explicitly represent fractures and matrix. DFM is among the most accurate methodologies to simulate flow in fractured systems since it describes properties of fractures directly. Due to high computational cost of DFM, its application in real field application is still limited. Furthermore, this approach is not suited for dynamic fracture problems, such as new fractures are generated in refracturing/infill drilling technologies of stimulated reservoirs in [53] and enhanced geothermal systems in [26], where updates of the fracture systems are frequently needed.

2.1.3 Embedded discrete fracture models

Embedded Discrete Fracture Model (EDFM), as a comprise, using structured grids to honor the accuracy of DFM while saving the computational costs. The 3D application of the model is firstly presented by Moinfar et al. in [42]. For a matrix grid and each fracture segment within the matrix grid, EDFM computes a transport index (transmissibility) between them by assuming that the pressure around the fracture is linearly distributed, and that the pressure gradient is the same in both sides of the fracture. The accuracy and efficiency were verified both by synthetic cases in [13, 55] and by field simulation cases in [20]. Recently, the EDFM method was further extended to modeling dynamic behaviors of fractures in [56] and impermeable flow barriers

in [1].

2.2 Reduced models for flow simulation in fractured systems

Though fine scale fracture modeling has been largely developed to capture flow behaviors associated with fractures, it requires many extra fracture grids for geological formations with numerous number of natural fractures. Different scales and complex geometries of fractures may also add the complexity of the modeling and computational efforts. Moreover, today's iterative modeling workflows require many runs of the forward problems, for example, in some inverse problems, uncertainty quantification problems and optimization problems. Fine scale simulation may not be applicable under such conditions, and approximating models are therefore needed to lessen the computational efforts.

2.2.1 Multiscale methods

One way to reduce the model complexity is to use multiscale simulation methods. During the last decade, different multiscale finite element (MSFE) or multiscale finite volume (MSFV) methods have been proposed for fractured systems. Hadi Hajibeygi et al. [26] proposed a MSFV method for fractured porous media, in which only one additional degree of freedom (DOF) was introduced for each interconnected fracture network and local functions were introduced to capture the fractures at the coarse scale. Later, this method

was extended to unstructured grids in [12]. Matei Tene et al. [2] proposed an algebraic multiscale method with embedded discrete fracture (F-AMS). This method introduced the basis functions based on coarsening ration and considered four different coupling strategies. Furthermore, a multiscale restriction smoothed basis (MsRSB) method was developed in the fractured media recently by Swej Shah et al. [50]. The method partitioned grids representing fractures into independent coarse grids and constructed the basis functions by restricted smoothing to get stable and robust performance. J.R. Natvig et al. [44] combined streamlines and multiscale simulation mimetic solver into a new simulation approach.

2.2.2 Upscaling methods

Upscaling is another way to construct approximate models by coarsening the fine grids and generating new parameters for coarse grids to solve the time-consuming issue in fine scale simulations. Coarse scale models for fractured systems are constructed from two different approaches: single continuum approach and dual-continuum approach. Single continuum approach merges fractures and matrix into one continuum and calculates equivalent permeability tensor for each assigned grid, for example, in [45, 10, 40, 46, 3, 37, 17, 23]. Oda tensor method [45], assigning only one DOF for a matrix grid and all fractures in it, assumes that all fractures are connected, the imposed pressure for each matrix grid linearly decreases and fractures exist sufficiently. As a result, Oda's method is limited to well-connected fracture networks due to not con-

sidering different fracture scales and connectivity. Pozdniakov and Tsang [46] applied self-consistent approach to obtain effective permeability of the fractured system by assuming that fractures interact with matrix medium but not directly with other fractures, which limited the method to cases with spatially distributed fractures but not highly connected fractures. Single continuum approach is limited to some certain fracture systems due to the assumptions where it is based.

Another way to coarsen fracture-related models is the dual-continuum approach. Flow based upscaling for fractured systems is the mostly used coarsening method and has been developed for the last decade. This method solves a simple hypothetical flow problem (usually single phase flow) in fine scale, then partitions fine scale fracture grids into coarse grids, and finally applies the solution of the flow problem to calculate parameters of the coarse model. For the hypothetical flow problems, different flow problems were solved either locally or globally.

Choice of hypothetical flow problems is crucial in flow-based upscaling framework since it determines the fidelity and thus the effectiveness of the coarse model. During the last decade, different local or global single phase flow problems have been applied to coarsen fine scale fractured models. Karimi-Fard et al. [34] firstly applied pseudo-study-state flow problem (Poisson equation was solved) locally for each coarse region to calculate transmissibility between matrix and fractures through it, and then employed steady-state flow problem (Laplace equation was solved) for each neighboring coarse region pair

to account for flow between different coarse fractures blocks. Gong et al. [25] then extended this local method by adding a local flow problem to account for the gravitational forces in geological systems where gravity force dominates. These local flow problems were further improved in [27, 21, 4] to enhance the performances of the coarse fracture models. Since local method introduce error by assigning assumed local boundaries and has to be frequently solved for each coarse region and each neighboring coarse pair, the global flow problem was introduced. Karimi-Fard et al. [33, 31] solved steady state single phase flow problem globally by assuming flow boundaries to drive flow in each coordinate direction, and then combined these independent solutions such that the local pressure gradient aligns with the vector normal to interface between neighboring coarse grids. The coarse model performances depend on the boundary conditions used, according to [14]. Therefore, the result of the modeling may highly depend on the boundaries, sources and sinks conditions prescribed to the flow problem.

For the grid aggregation problem, different ways were conducted in different studies. In [34, 25, 27, 21], a multiple sub-region (MSR) method is developed, where

the entire model has been partitioned into several regions at first, then only one DOF of fracture is assigned to each coarse region, and the matrix continuum of this region is partitioned using the solution of flow problem solved before (the number of coarse matrix grids in each coarse region is given as input). This approach is limited to systems with highly and strongly connected

fractures. Subsequently, Gong [24] developed a combined method by using MSR to deal with large and well-connected fractures, and applying DFM to model distributed and disconnected fractures. Later, Fumagalli [4] improved the MSR partitioning approach by using a recursive breadth-first search and a splitting algorithm, which allow more than one DOF for fractures through each coarse region. Another aggregation method used in [31, 28] gathers fractures that are intersected for each coarse region.

Chapter 3

Formulations and Discretization Method for Flow Simulation of Fractured Systems

3.1 Governing equations

In this section, we summarize the governing equations and fine scale discretized system for the flow simulation considered in fractured porous media.

Three-phase black oil model is considered for both the fine and coarse scales simulations. Without loss of generality, we assume that $\Omega \subset \mathbb{R}^d$ (here we only discuss d=2) is the entire simulation domain. Let Ω^m denotes the matrix domain, and $\Omega^f = \Omega \backslash \Omega^m$ represents the fracture domain. The governing equations for black oil model can be expressed as

$$\phi \frac{\partial}{\partial t}\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o}\right) + \nabla \cdot \left(\frac{1}{B_g}\mathbf{u_g} + \frac{R_{so}}{B_o}\mathbf{u_o}\right) = f_g, \qquad x \in \Omega, \tag{3.1a}$$

$$\phi \frac{\partial}{\partial t}\left(\frac{S_o}{B_o}\right) + \nabla \cdot \left(\frac{1}{B_o}\mathbf{u_o}\right) = f_o, \qquad x \in \Omega, \tag{3.1b}$$

$$\phi \frac{\partial}{\partial t}\left(\frac{S_w}{B_w}\right) + \nabla \cdot \left(\frac{1}{B_w}\mathbf{u_w}\right) = f_w, \qquad x \in \Omega, \tag{3.1c}$$

where ρ_α, S_α, B_α and f_α are the density, saturation, formation factor and external source of each phase $\alpha \in \{g, o, w\}$, respectively. R_{so} is the solution gas oil ration, which represents the amount of gas dissolved in the oil at given

condition. Velocity of each phase $\mathbf{u}_\alpha$ could be expressed in Darcy's law:

$$\mathbf{u}_\alpha = -\frac{k_{r\alpha}}{\mu_\alpha}\mathbf{K}\,\nabla p_\alpha, \qquad \alpha \in \{g,\, o,\, w\}, \tag{3.2}$$

where $\mathbf{K}$ is the permeability tensor, μ_α, $k_{r\alpha}$ and p_α are the viscosity, relative permeability and pressure of phase α. In addition to equations 3.1 and 3.2, the saturations of phases are constrained by

$$\sum_\alpha S_\alpha = 1 \tag{3.3}$$

and the capillary pressure between different phases are defined by setting oil phase as the reference:

$$p_{c\alpha} = p_\alpha - p_o, \qquad \alpha \in \{g,\, w,\} \tag{3.4}$$

where p_{co} represents the capillary pressure of the gas phase, and p_{cw} represents the negative water phase capillary pressure.

The pressure equation is derived by combining the previous equations with the pressure-volume-temperature (PVT) properties. The formulation and detail derivation of pressure equation for black oil model can be found in [15].

3.2 Fine scale discretization

We split the matrix domain Ω^m in uniform Cartesian grids with dimension $N_x^m \times N_y$. As for fractures, EDFM is used to construct the fracture grids and their connections with the matrix grids. EDFM could deploy a structured grid to represent each fracture segment partitioned by matrix grids in the

physical domain and then append the new grid (denoted as "embedded grid") to the original matrix grids to form the computational domain. The volume of the fracture grid equals to the physical volume of the fracture by adjusting the porosity of the grid. The transition from upper layer to the middle layer in Figure 3.1 is an example showing this process.

In the context of this thesis, we append the embedded fracture grids in the x direction. Assuming that the matrix grids are distributed in the first N_x^m columns in x direction, and the last N_x^f columns of grids in x direction are fracture grids, we have $N_x^m + N_x^f$ columns of grids in total and we denote this number as N_x. We denote i as the 2D index point of each grid by

$$i := (i_1, i_2), \quad i_1, i_2 \in \mathbb{Z}.$$

Let $\mathcal{I}$, $\mathcal{I}^m$ and $\mathcal{I}^f$ be the index sets of all grid, matrix grids, and fracture grids respectively, given by

$$\mathcal{I} := \{i = (i_1, i_2) : 1 \leqslant i_1 \leqslant N_x,\ 1 \leqslant i_2 \leqslant N_y\},$$
$$\mathcal{I}^m := \{i = (i_1, i_2) : 1 \leqslant i_1 \leqslant N_x^m,\ 1 \leqslant i_2 \leqslant N_y\},$$
$$\mathcal{I}^f := \mathcal{I} \backslash \mathcal{I}^m.$$

We also denote the physical domain of grid i as ω_i and $\mathcal{N}_i$ as the index set of each grid's neighbors by

$$\mathcal{N}_i := \{j \in \mathcal{I} : \omega_i \cap \omega_j \neq \emptyset,\ j \neq i\}.$$

Here we apply Darcy's law to two grids that have connection and define the transmissibility between them as

$$T_{ij} = \frac{q_{ij,\alpha}}{\frac{k_{r\alpha}}{\mu_\alpha}(p^+_{ij,\alpha} - p^-_{ij,\alpha})}, \qquad \forall i,j \in \mathcal{I} \quad s.t. \quad i \in \mathcal{N}_j,\ j \in \mathcal{N}_i, \tag{3.5}$$

where $q_{ij,\alpha}$ is the volume flux of phase α between grid i and grid j, which is positive. $p^+_{ij,\alpha}$ is the larger pressure of $(p_{i,\alpha}, p_{j,\alpha})$, and $p^-_{ij,\alpha}$ is the lower pressure of $(p_{i,\alpha}, p_{j,\alpha})$. The transmissibility only depends on the permeability and geometry of the computational domain considered.

Besides transmissibility between matrix grids, transmissibility involved with fracture grids (also called "non-neighbor connections") are generally categorized into three types: T^I_{nnc} is the transmissibility between fracture and matrix it penetrates, T^{II}_{nnc} is the transmissibility between connected fracture segments in each individual fracture, and T^{III}_{nnc} is the transmissibility between intersecting fracture segments. The formula to calculate transmissibilities and the detail derivations of these formula can be found in [55].

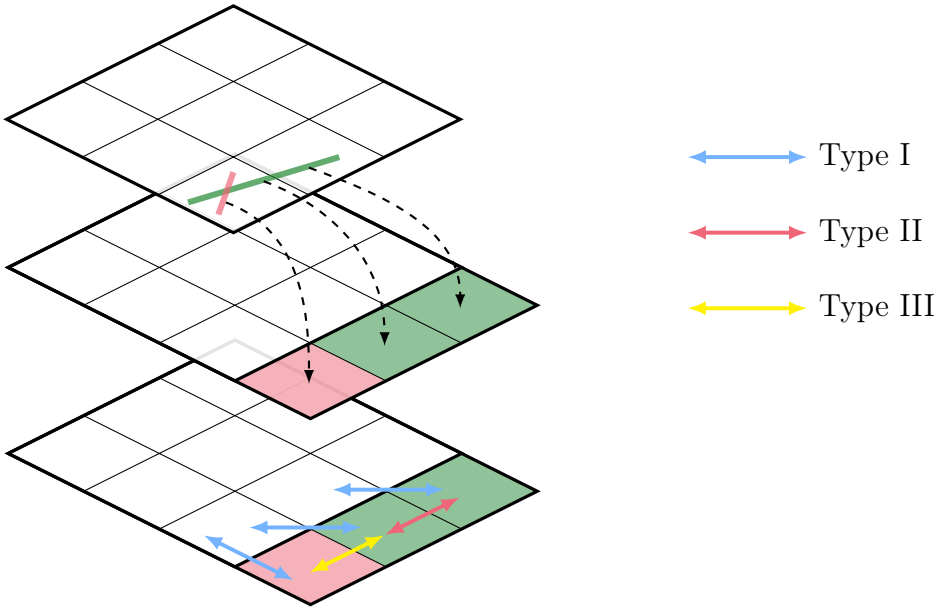


Figure 3.1: A sketch for illustration of embedded discrete fracture model. Upper layer: the physical domain considered here has 9 fine matrix grids and 2 fractures intersected with each other (green line and red line). Middle layer: the computational domain is constructed by embedding three fracture grids to the original matrix grids. The green grids correspond to the green fracture and the red one to the red fracture. Lower layer: extra connections considered to take the effects of fractures. The blue arrows represent connections between matrix and fracture grids, red arrow indicates connection between fracture segments in an individual fracture, and the yellow arrow represents the connection between different intersected fractures.

Chapter 4

Upscaling Methodology for Flow Simulation of Fractured Systems

In this section, the upscaling framework is presented by steps. Firstly, we explained different hypothetical flow problems that could be deployed to provide fine pressure solution. Then, we discussed different methods used to agglomerate fine scale fracture grids in order to form the coarse scale fracture girds. Parameters of the new system of coarse model are then discussed. At the end of this chapter, we presented the implication approach of the time-stepping method in unsteady state problem.

Two main procedures needed to be done are: constructing the coarse grids and calculating the model parameters for new coarse system. Coarsening grids is done by aggregating the original fine grids. How coarse is the new model constructed and how the fine grids are agglomerated determine the speed up and the accuracy of the simulation of coarse model. Besides, the method for generating coarse model parameters also controls how close are the simulation results as fine scale simulation.

4.1 Hypothetical fine scale flow problems

The fine scale pressure used for calculating coarse transimissibilities are the solution of hypothetical single-phase flow problems. Different flow problems, studied in the past few decades, would generate different coarse scale transmissibilites. The performance of transmissibilies depends how the boundary conditions and well conditions were applied. Generally, using real boundary and well configurations could generate better coarse simulation results. For the simulation problem with both source and sink terms, steady state solution exists when real boundary and well conditions are applied. But for the simulation cases with only sinks (for example, the case with no-flow boundaries and only production wells), the flowing of flow comes from the compressibility of flow and no steady-state solution exits under such conditions.

4.1.1 Steady state problem

By using real boundary and well conditions, a steady state solution exists when the reservoir model has both the sink and source. The steady state flow problem solved is

$$\nabla \cdot (\mathbf{K} \nabla p) = r, \tag{4.1}$$

where r is the source/sink term which stems from boundary and well conditions. In most of the computational domain, the steady state problem solves Laplacian equation. The discretized form of this problem is simply solved in a system of linear equations.

4.1.2 Unsteady state problem

For models with only sinks but no sources, the flowing of the fluid is caused by the compressibility of the fluid. Here we introduce a time-stepping method to involve the true boundary and well conditions and then to calculate the coarse transmissibilites. The flow problem considered under such condition is a slightly compressible single-phase fluid (for example, water) flow. The mass conservation equation and Darcy's law for such flow can be expressed as

$$\frac{\partial(\phi\rho)}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = r, \tag{4.2}$$

$$\mathbf{u} = -\frac{\mathbf{K}}{\mu}\nabla p, \tag{4.3}$$

where r is the source/sink term and is positive when fluid flows in. We denote the compressibility of rock and fluid as c_f and c_l. The density and porosity can be calculated as

$$\rho = \rho_r[1 + c_l(p - p_r)], \tag{4.4}$$

$$\phi = \phi_r[1 + c_f(p - p_r)], \tag{4.5}$$

where ρ_r and ϕ_r are the referenced density and porosity under referenced pressure p_r and referenced temperature T_r.

For the accumulation term, we have

$$\begin{aligned}\frac{\partial(\phi\rho)}{\partial t} &= \phi\frac{\partial(\rho)}{\partial t} + \rho\frac{\partial(\phi)}{\partial t} \\ &= \phi\frac{d\rho}{dp}\frac{\partial p}{\partial t} + \rho\frac{d\phi}{dp}\frac{\partial p}{\partial t} \\ &= [\rho_r\phi_r c_t + 2\rho_r\phi_r c_l c_f(p - p_r)]\frac{\partial p}{\partial t},\end{aligned} \tag{4.6}$$

where $c_t = c_f + c_l$ is the total compressibility. Since the second term in the above equation is too small compared to the first one (c_l and c_f usually have the order of 10^{-6}), we can approximate the accumulation term as:

$$\frac{\partial(\phi\rho)}{\partial t} = \rho_r \phi_r c_t \frac{\partial p}{\partial t}. \tag{4.7}$$

For the flux term, we have

$$\begin{aligned} \nabla \cdot (\rho \mathbf{u}) &= \nabla \cdot (\rho_r (1 + c_l (p - p_r)\mathbf{u})) \\ &= \rho_r \nabla \cdot ((1 + (p - p_r)\mathbf{u})). \end{aligned} \tag{4.8}$$

The pressure equation can be finally derived by substituting the equations 4.7, 4.8 and 4.3 into equation 4.1:

$$\rho_r \phi_r c_t \frac{\partial p}{\partial t} + \rho_r \nabla \cdot ((1 - \frac{(p - p_r)}{\mu} \mathbf{K} \nabla p)) = r. \tag{4.9}$$

Next, we use two-point flux approximation (TPFA) finite volume scheme to get discretization of the above equation in the fine scale. We assume we are dealing with the fine grid of index i at the $(n+1)th$ time step. For accumulation term, we have

$$\int_{\omega_i} \rho_r \phi_r c_t \frac{\partial p}{\partial t} dv = \frac{\rho_r V_i \phi_r c_t}{\triangle t} (p_i^{n+1} - p_i^n), \tag{4.10}$$

where $\triangle t$ is the time step size.

By using Divergence Theorem, the flux term can be converted to

$$\begin{aligned} \int_{\omega_i} \rho_r \nabla \cdot ((1 + (p - p_r)\mathbf{u}))dv &= \oint_{\partial\omega_i} \rho_r (1 + (p - p_r)\mathbf{u}) ds \\ &= -\sum_{j \in \mathcal{N}_i} \frac{\rho_r}{\mu} (1 + c_l \triangle p_{ij}^{+n})(p_j^{n+1} - p_i^{n+1}). \end{aligned} \tag{4.11}$$

Therefore, the discretized form of the equation can be derived as

$$\frac{\rho_r V_i \phi_r c_t}{\triangle t}(p_i^{n+1} - p_i^n) = \sum_{j \in \mathcal{N}_i} \frac{\rho_r}{\mu}(1 + c_l \triangle p_{ij}^{+n}) T_{ij}(p_j^{n+1} - p_i^{n+1}) + R_i. \quad (4.12)$$

A system of linear equations is solved to obtain the fine scale pressure solution at each time step. A realistic implementation problem associated with this time-stepping method is how many time steps should be solved to get the appropriate pressure solution to be used.

4.2 Coarse grids construction

In this section, we mainly discuss three agglomeration methods for fractures grids from fine to coarse scale. The construction of coarse matrix grids is out of discussion.

We note that no near-well upscaling is performed in our coarsening framework. Extra upscaling procedure is always required when blocks near wells are also expected to upscaled to coarse scale. The reason is that high pressure gradient and multiple phase flow effects both add complications in the near-well region. For example, the gas condensate may happen near wells and have large effects on the gas productivity. Under such conditions, using only the well index could not capture the phase behaviors. Many near-well upscaling methods have been studied for practical simulation purpose, one could refer to [19], [43] for some of these methods. In this paper, we do not upscale the blocks intersected by wells.

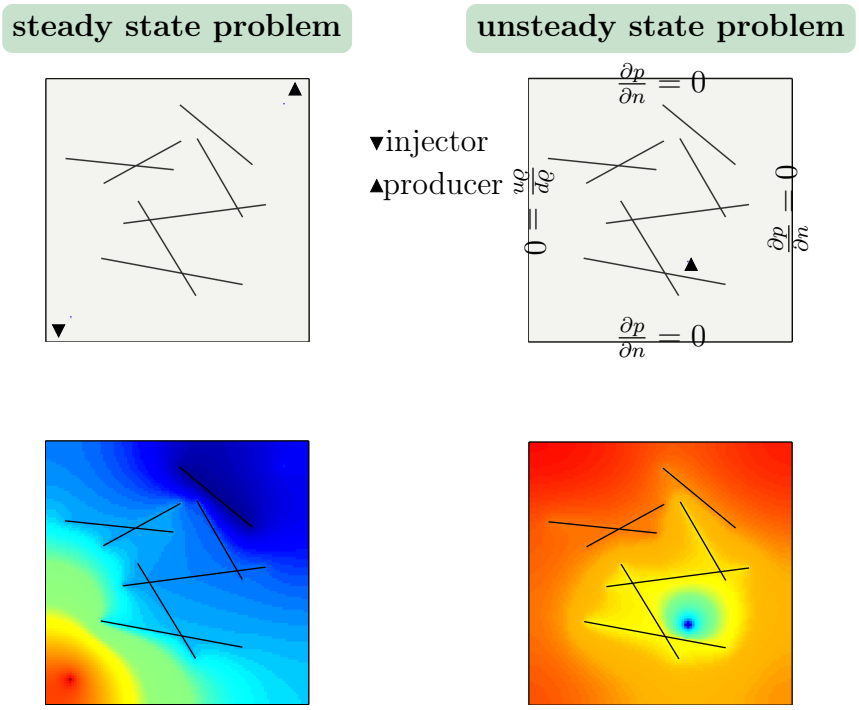


Figure 4.1: An illustration of flow problems used for calculating coarse scale transmissibilities. The steady state problem (upper left) is used for flow systems with both source and sink (in this case, both injector well and production well exist). The pressure profile of this problem (lower left) is the solution of the Laplacian equation. The unsteady state problem (upper right) is used for flow systems with only sinks (in this case, the flow boundaries are no-flow boundaries and only production well exists). The pressure distribution (lower right) is the solution by solving equations of slightly compressible single-phase flow. Note that, the pressure varies from high values (red) to low values (blue).

In this thesis, the structured coarse matrix grid size is generally determined by the averaged size of the matrix domain using the given number of the coarse matrix grids (for example, see the "problem setup" in Figure 4.2). For the near-well region, moderate size of grid is used as buffer to adjust the abrupt change from coarse matrix grid to fine matrix grid. For each coarse matrix grid I, we denote by $\widehat{\mathfrak{I}}_I$ the index set containing all the indexes of the fine matrix grids in I, by $\widehat{\mathcal{F}}_I$ the index set containing all the fine fractures grid

within I, we also define the domain of the coarse matrix grid I as:

$$\widehat{\omega}_I := \bigcup_{i \in \widehat{\mathcal{I}}_I} \omega_i,$$

and $\widehat{\mathcal{C}}_I$ as all the pairs of interconnected fine fractures (either intersected or in a same fracture) in I:

$$\widehat{\mathcal{C}}_I := \{(f_1,\, f_2) : f_1, f_2 \in \widehat{\mathcal{F}}_I,\, \omega_{f_1} \cap \omega_{f_2} \neq \emptyset\}.$$

We also introduce $\widehat{\mathcal{A}}^m$ as the index set for each interface between two neighboring matrix grids. For each interface $s \in \widehat{\mathcal{A}}^m$, we denote by $\widehat{\mathcal{IA}}_s$ and $\widehat{\mathcal{FA}}_s$ the index set for coarse matrix grids sharing the interface and index set for pairs of fine fracture grids through the interface:

$$\begin{aligned}
\widehat{\mathcal{IA}}_s &= (I_1,\, I_2), \qquad I_1,\, I_2 \in \widehat{\mathcal{I}}^m,\, \omega_{I_1} \cap \omega_{I_2} = \omega_s, \\
\widehat{\mathcal{FA}}_s &= \{(f_1,\, f_2) : f_1 \in \widehat{\mathcal{F}}_{I_1},\, f_2 \in \widehat{\mathcal{F}}_{I_2},\, \omega_{f_1} \cap \omega_{f_2} \neq \emptyset\},
\end{aligned}$$

where ω_s represents the domain of the interface s.

For the upscaling methods exploiting unstructured grids, grouping fine scale matrix grids can be achieved by different partitioning methods: graph partitioning technique [35], geological partitioning [31], and flow-based partitioning [31].

4.2.1 Greedy aggregation

A simple way to agglomerate fine fracture grids in a coarse matrix grid is to gather them all in one coarse fracture grid (see Figure 4.2). We denote

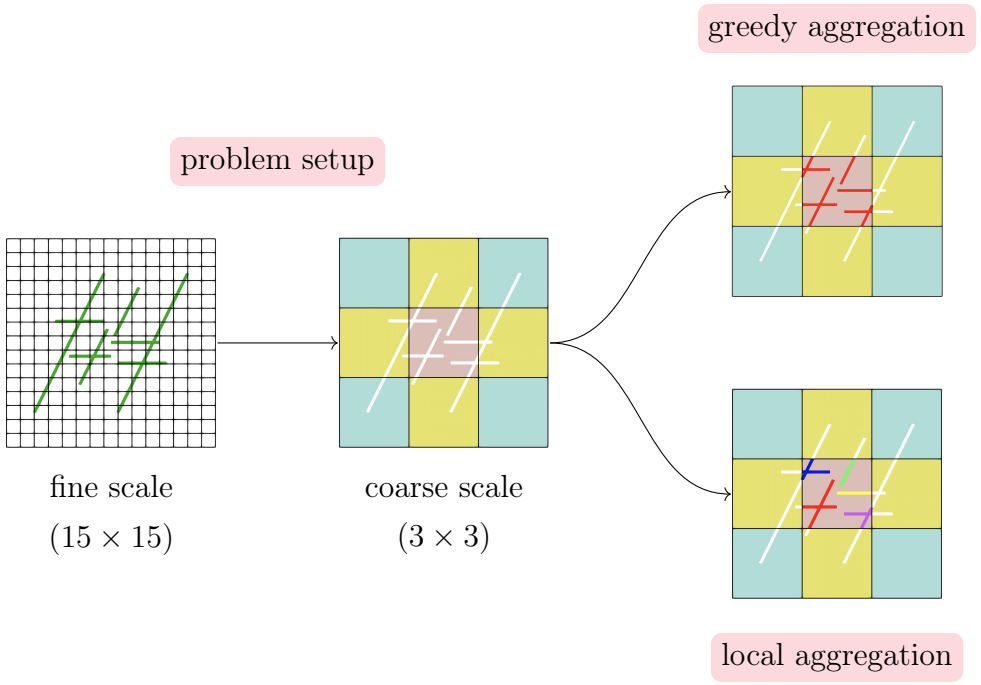


Figure 4.2: An illustration of greedy aggregation and local aggregation methods. The problem (middle) presented here has 3×3 coarse matrix grids constructed by 15×15 fine matrix grids with 8 fractures (left). Here we consider centered matrix grid (in red), the neighboring matrix grids of which are shaded in yellow. Greedy aggregation method (upper right) agglomerates all the fine fracture grids in the matrix grid into one coarse fracture grid (in red). Local aggregation method (lower right) only gathers fine fracture segments of each individual fracture (each color represents each coarse fracture grid).

the coarse fracture grid associated with the coarse matrix grid I as J and fine fracture grids included in this grid are

$$\widehat{I}_J = \bigcup_{i \in \widehat{\mathcal{I}}_I} (\mathcal{N}_i \cap \mathcal{I}^f). \tag{4.13}$$

In this scheme, at most one degree of freedom is assigned to fractures within each matrix grid. This method is simple and could generate limited fracture grids in coarse scale, which results in accelerating the simulation speed. We will show in the numerical tests section that this aggregation method

works well especially for systems with short natural fractures. But this greedy aggregation could impair the connectivity information of large fractures (for example, hydraulic fractures) which transverse several coarse blocks. The error generated at one local matrix grid may even propagate along the large fractures to other regions in the domain.

4.2.2 Local aggregation

Local aggregation method deals with fine scale fractures based on the local intersections of different fractures or local interconnections within an individual fracture. To be specific, for fractures intersected in a coarse matrix grid I, all the fine fracture segments should be aggregated to construct one coarse fracture agglomerate. For fractures without intersections with other fractures in I, coarse fracture grids are constructed individually by gathering only fine grids of each fracture in I. The lower right figure in Figure 4.2 illustrates an example of the local aggregation scheme. Based on fracture intersections, 5 clusters (in different colors) were formed in the matrix grid that is considered (in red). Algorithm algorithm 1 presented tow basic functions to combine fracture indices within a set. Furthuremore, a pseudo-code algorithm 2 is presented to show the local aggregation algorithm for each coarse matrix grid I.

The reason behind this method is that intersected or interconnected fractures have similar pressure values comparing with fractures without connections, since fractures can be considered as high speed flow conduits com-

Algorithm 1: GETINDEX $(f, \mathcal{I})$ and COMBINE$(f_1, f_2, \mathcal{I})$

```
Function GETINDEX(f, I):
    i ← 0                                          // if f ∉ I, i is set as 0
    if f ∈ I then
        i ← 1
        while f ∉ I[i] do
            i ← i + 1    // if f ∈ I, i is the index of set where f firstly
                          occurred
    return i
Function COMBINE(f1, f2, I):
    i1 = GETINDEX (f1, I), i2 = GETINDEX (f2, I)
    if i1 = 0 and i2 = 0 then                      // both f1 and f2 are not in I
        I ← I ∪ {f1, f2}
    else if i1 = 0 and i2 ≠ 0 then                 // f1 is not in I but f2 is in I
        I[i2] ← I[i2] ∪ {i1}
    else if i1 ≠ 0 and i2 = 0 then                 // f1 is in I but f2 is not in I
        I[i1] ← I[i1] ∪ {i2}
    else if i1 ≠ 0, i2 ≠ 0 and i1 ≠ i2 then        // f1 and f2 are in the different
                                                    subsets of I
        I[i1] ← I[i1] ∪ I[i2], I ← I\I[i2]
    return I
```

paring with background matrix. However, the limitation of this aggregation method is that the maximum number of coarse fracture grids constructed finally is not guaranteed. For regions with numerous fractures (for example, some natural fractured reservoirs), this method could generate a lot of fracture grids, the number of which may be many times the number of matrix grids.

4.2.3 Multi-level aggregation

Here we introduce a multi-level aggregation progress in order to preserve the fidelity of fractures with long lengths and lessen the number of coarse fracture grids in total. Three new attributes are defined as following:

Algorithm 2: LOCAL $(I, \widehat{\mathcal{F}}_I, \widehat{\mathcal{C}}_I)$

```
▷ construct sets based on fracture connections in coars matrix I
if F̂_I ≠ ∅ then
    if Ĉ_I ≠ ∅ then
        for (f_1, f_2) ∈ Ĉ_I do
            COMBINE(f_1, f_2, Q̂_I)
    for f ∈ F̂_I do
        if f ∉ Q̂_I then
            Q̂_I ← Q̂_I ∪ {f}
return Q̂_I
```

- $\tau_s \in \{0, 1\}$: neighboring connection attribute for each interface $s \in \widehat{\mathcal{A}}^m$.
- $\beta_I \in \mathbb{N}$: length attribute for each coarse matrix grid $I \in \widehat{\mathcal{I}}^m$.
- $\gamma_s \in [0, 1)$: volume attribute for each interface $s \in \widehat{\mathcal{A}}^m$.

We introduce the index set $\widehat{\mathcal{I}}_I$ that contains all the index of fine scale grids in coarse grid I, and index set $\widehat{\mathcal{N}}_I$ that contains all the index of coarse grids who share common interfaces with grid I,

$$\widehat{\mathcal{I}}_I := \{i \in \mathcal{I} : \omega_i \in \widehat{\omega}_I\},$$
$$\widehat{\mathcal{N}}_I := \{J \in \widehat{\mathcal{I}} : \widehat{\omega}_I \cap \widehat{\omega}_J \neq \emptyset, J \neq I\}.$$

Multi-level aggregation algorithm is illustrated in algorithm 3. The steps for this aggregation method are as follows:

1. Construct the coarse fracture grids based on the local aggregation method (the first "for" loop in algorithm 3).

2. For each interface between neighboring coarse matrix grids, if $\beta_I = 1$, we consider the neighboring fracture intersections. If two fractures are in the same coarse fracture grid in one matrix grid, these two fractures should in the same grid in another matrix grid (the second "for" loop in algorithm 3).

3. For each coarse matrix grid, all marked short fractures should be aggregated to one coarse fracture cell (the third "for" loop in algorithm 3).

4. For each interface between neighboring coarse matrix grids, neighboring coarse fracture grids should be aggregated if their volume is less than the volume attribute (the fourth "for" loop in algorithm 3).

A geometric illustration of multi-level partition is presented in Figure 4.3. The start point of the multi-level aggregation method is the coarsening scheme constructed by local aggregation method (as (a) in Figure 4.3). To simplify the illustration, we assume the neighboring connection attribute τ_s as the same value for all coarse interfaces $s \in \widehat{\mathcal{A}}^m$, and denote it as N. Similarly, we assume the length attribute $_I$ as the same value for all coarse matrix grids $I \in \widehat{\mathcal{T}}^m$, and denote it as L. If we consider three different inputs for the neighbor attribute N and fracture length attribute L (here we assume that all coarse matrix blocks share the same length attribute). (1) $N = 1, L = 4$: considering fracture intersection in the neighbor matrix blocks (in yellow), we can combine two fracture sets (yellow and purple in (a)) into one (purple in

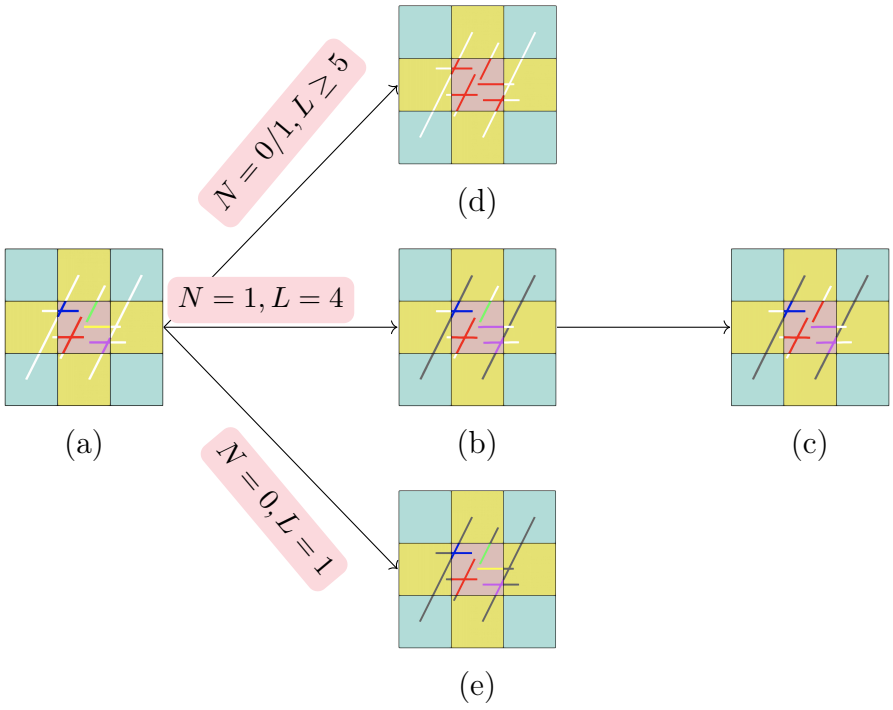


Figure 4.3: An illustration of multi-level aggregation method.

(b)). Since two long fractures (in gray) intersect the matrix block, we can combine the short fracture sets (green and red in (b)) into one set (red in (c)). The final number of coarse fracture sets associated with the considered matrix grid is reduced from 5 to 3. (2) $N = 0, L = 1$: without considering neighboring fracture intersections, and treating all fractures as long fractures make the coarse fracture sets remain the same (as exactly the results of local aggregation method). (3) $N = 0/1, L \geq 5$: whenever considering the neighboring intersections, the value of L decides that all fractures in this problem are seen as short fractures, and all the fracture sets in (a) could be aggregated into 1 (as in (d)). This is exactly the scheme of the greedy aggregation method.

Algorithm 3: MULTILEVEL

input : coarse scale set: $\widehat{\mathcal{I}}^m, \widehat{\mathcal{A}}^m, \widehat{\mathcal{F}}_I, \widehat{\mathcal{FA}}_A, \widehat{\mathcal{L}}_m, \widehat{\mathcal{L}}_f$
output: coarse scale fracture sets.
▷ *construct sets based on local aggregation method*

for $I \in \widehat{\mathcal{I}}^m$ **do**
  LOCAL $(I, \widehat{\mathcal{F}}_I, \widehat{\mathcal{C}}_I)$

▷ *reduce sets based on neighboring fracture intersections*

for $s \in \widehat{\mathcal{A}}^m$ *s.t.* $\widehat{\mathcal{FA}}_s \neq \emptyset$, $\tau_s = 1$ **do**
  $(I, J) = \widehat{\mathcal{IA}}_s$
  for $(f_1, f_2), (t_1, t_2) \in \widehat{\mathcal{FA}}_s$ *s.t.* $(f_1 \neq t_1)$ **do**
    $i_1 = \text{GETINDEX}(f_1, \widehat{\Omega}_I)$, $i_2 = \text{GETINDEX}(t_1, \widehat{\Omega}_I)$, $j_1 = \text{GETINDEX}(t_1, \widehat{\Omega}_J)$, $j_2 = \text{GETINDEX}(t_2, \widehat{\Omega}_J)$
    if $(f_1, t_1) \in \widehat{\mathcal{C}}_I$ *and* $j_1 \neq j_2$ **then**
      COMBINE$(f_2, t_2, \widehat{\Omega}_J)$
    if $(t_1, t_2) \in \widehat{\mathcal{C}}_J$ *and* $i_1 \neq i_2$ **then**
      COMBINE$(f_1, t_1, \widehat{\Omega}_I)$

▷ *reduce sets based on fracture length*

for $I \in \widehat{\mathcal{I}^m}$ *s.t.* $\widehat{\mathcal{F}}_I \neq \emptyset$, $\beta_I > 1$ **do**
  $flag \leftarrow 0$
  for $q \in \widehat{\Omega}_I$ **do**
    if $\forall f_i \in q$ *s.t.* $\widehat{\mathcal{L}}_{f_i} < \beta_I$ **then**
      if $flag = 0$ **then**
        $f_{short} = q[1]$, $flag \leftarrow 1$
      else
        COMBINE$(f_{short}, q[1], \widehat{\Omega}_I)$

▷ *reduce sets based on fracture set volume*

for $s \in \widehat{\mathcal{A}}^m$ *s.t.* $\widehat{\mathcal{FA}}_s \neq \emptyset$ **do**
  $(I, J) = \widehat{\mathcal{IA}}_s$
  for $(f_1, f_2) \in \widehat{\mathcal{FA}}_s$ **do**
    $i = \text{GETINDEX}(f_1, \widehat{\Omega}_I)$, $j = \text{GETINDEX}(f_2, \widehat{\Omega}_J)$
    if VOLUME$(\widehat{\Omega}_I[i]) < \gamma_s$ VOLUME$(\widehat{\Omega}_J[j])$ **then**
      $\widehat{\Omega}_J \leftarrow \widehat{\Omega}_J \cup \widehat{\Omega}_I[i]$, COMBINE $(f_1, f_2, \widehat{\Omega}_J)$, $\widehat{\Omega}_I \leftarrow \widehat{\Omega}_I \backslash \widehat{\Omega}_I[i]$
    if VOLUME$(\widehat{\Omega}_J[j]) < \gamma_s$ VOLUME$(\widehat{\Omega}_I[i])$ **then**
      $\widehat{\Omega}_I \leftarrow \widehat{\Omega}_I \cup \widehat{\Omega}_J[j]$, COMBINE $(f_1, f_2, \widehat{\Omega}_I)$, $\widehat{\Omega}_J \leftarrow \widehat{\Omega}_J \backslash \widehat{\Omega}_J[j]$

return $\widehat{\Omega}$

4.3 Coarse model parameters

The coarse grid volume is the summation of fine grids' volume it contains, given by

$$\widehat{V}_I := \sum_{i \in \widehat{I}_I} V_i.$$

The porosity and pressure of the coarse grid are defined on the basis of volume weighted average:

$$\widehat{\phi}_I := \frac{\sum_{i \in \widehat{I}_I} V_i \phi_i}{\widehat{V}_I},$$
$$\widehat{p}_I := \frac{\sum_{i \in \widehat{I}_I} V_i p_i \phi_i}{\widehat{V}_I \widehat{\phi}_I}.$$

For two adjacent coarse grids I and J who share an interface, we define $\widehat{\mathcal{D}}_{IJ}$ as the index set of fine grids in coarse grid I who share an interface with fine grid in J, and $\widehat{\mathcal{D}}_I$ as the index set of all fine grids who share interfaces with other coarse grids, given by

$$\widehat{\mathcal{D}}_{IJ} := \{i \in \widehat{I}_I : \omega_i \cap \widehat{\omega}_J \neq \emptyset\},$$
$$\widehat{\mathcal{D}}_I := \bigcup_{J \in \widehat{A}_I} \widehat{\mathcal{D}}_{IJ}.$$

And the coarse transmissibility across the interface between grid I and J is given by

$$\widehat{T}_{IJ} := \frac{\sum_{i \in \widehat{\mathcal{D}}_{IJ}} \sum_{j \in \widehat{I}_J \cap \mathcal{N}_i} T_{ij}(p_i - p_j)(1 + c_l \triangle p_{ij}^+)}{(\widehat{p}_I - \widehat{p}_J)(1 + c_l \triangle \widehat{p}_{IJ}^+)}, \tag{4.14}$$

where $\triangle p_{ij}^{+}$ and $\triangle \widehat{p}_{IJ}^{+}$ are the bigger pressure difference based on the referenced pressure of two adjacent grids, and can be expressed as

$$\triangle p_{ij}^{+} = p_{ij}^{+} - p_r, \quad p_{ij}^{+} = \max\{p_i, p_j\},$$
$$\triangle \widehat{p}_{IJ}^{+} = \widehat{p}_{IJ}^{+} - p_r, \quad \widehat{p}_{IJ}^{+} = \max\{\widehat{p}_I, \widehat{p}_J\}.$$

By defining coarse properties in such ways, the coarse scale models still preserve mass conservation in the single phase flow.

Theorem 4.3.1. *Mass conservation is satisfied in coarse scale of single phase flow at each time step by using the transmissibilities defined in equation 4.14.*

Proof. Let us consider a coarse grid with index I, we can add up equation 4.12 for all the fine grids in coarse grid I and derive

$$\sum_{i\in\widehat{\mathcal{I}}_I} \frac{\rho_r V_i \phi_r c_t}{\triangle t}(p_i^{n+1} - p_i^n) \tag{4.15}$$
$$= \sum_{i\in\widehat{\mathcal{I}}_I} \sum_{j\in\mathcal{N}_i} \frac{\rho_r}{\mu}(1 + c_l \triangle p_{ij}^{+n}) T_{ij}(p_j^{n+1} - p_i^{n+1}) + \sum_{i\in\widehat{\mathcal{I}}_I} r_i. \tag{4.16}$$

For the left hand side term in equation 4.15, by using the previous definition we can obtain

$$\sum_{i\in\widehat{\mathcal{I}}_I} \frac{\rho_r V_i \phi_r c_t}{\triangle t}(p_i^{n+1} - p_i^n) \tag{4.17}$$
$$= \frac{\rho_r \widehat{V}_I \widehat{\phi}_I c_t}{\triangle t}\left(\frac{\sum_{i\in\widehat{I}_I} V_i p_i^{n+1}\phi_i}{\widehat{V}_I\widehat{\phi}_I} - \frac{\sum_{i\in\widehat{I}_I} V_i p_i^{n}\phi_i}{\widehat{V}_I\widehat{\phi}_I}\right) \tag{4.18}$$
$$= \frac{\rho_r \widehat{V}_I \widehat{\phi}_I c_t}{\triangle t}(\widehat{p}_I^{n+1} - \widehat{p}_I^n). \tag{4.19}$$

Note that

$$\widehat{I}_i = \widehat{D}_I \cup (\widehat{\mathcal{I}}_I \backslash \widehat{\mathcal{D}}_I), \tag{4.20}$$

$$\forall i \in \widehat{\mathcal{I}}_I, \qquad \mathcal{N}_i = (\mathcal{N}_i \cap \widehat{\mathcal{I}}_I) \cup (\mathcal{N}_i \backslash \widehat{\mathcal{I}}_I), \quad \emptyset = (\mathcal{N}_i \cap \widehat{\mathcal{I}}_I) \cap (\mathcal{N}_i \backslash \widehat{\mathcal{I}}_I). \tag{4.21}$$

We introduce f_{ij} to denote the fine scale flux term at $(n+1)th$ time step,

$$f_{ij}^{n+1} := \frac{\rho_r}{\mu}(1 + c_l \triangle p_{ij}^{+n}) T_{ij}(p_j^{n+1} - p_i^{n+1}), \qquad \forall i \in \widehat{\mathcal{I}}_I, j \in \mathcal{N}_i$$

The first term in the right hand side of equation 4.15 can then be abbreviated as

$$\sum_{i \in \widehat{\mathcal{I}}_I} \sum_{j \in \mathcal{N}_i} \frac{\rho_r}{\mu}(1 + c_l \triangle p_{ij}^{+n}) T_{ij}(p_j^{n+1} - p_i^{n+1}) \tag{4.22}$$

$$= \sum_{i \in \widehat{\mathcal{I}}_I} \sum_{j \in \mathcal{N}_i} f_{ij}^{n+1} \tag{4.23}$$

$$= \sum_{i \in \widehat{\mathcal{D}}_I} \sum_{j \in \mathcal{N}_i \backslash \widehat{\mathcal{I}}_I} f_{ij}^{n+1} + \sum_{i \in \widehat{\mathcal{D}}_I} \sum_{j \in \mathcal{N}_i \cap \widehat{\mathcal{I}}_I} f_{ij}^{n+1} + \sum_{i \in \widehat{\mathcal{I}}_I \backslash \widehat{\mathcal{D}}_I} \sum_{j \in \mathcal{N}_i} f_{ij}^{n+1}. \tag{4.24}$$

We also note the following facts:

- $\mathcal{N}_i \backslash \widehat{\mathcal{I}}_I = \bigcup_{J \in \widehat{A}_I} (\mathcal{N}_i \cap \widehat{\mathcal{N}}_J), \qquad \forall i \in \widehat{\mathcal{I}}_I$
- $(\mathcal{N}_i \cap \widehat{\mathcal{N}}_{J_1}) \bigcap (\mathcal{N}_i \cap \widehat{\mathcal{N}}_{J_2}) \neq \emptyset, \qquad \forall J_1, J_2 \in \widehat{\mathcal{N}}_I, J_1 \neq J_2$
- $\widehat{\mathcal{D}}_I = \widehat{\mathcal{D}}_{IJ} \cup (\widehat{\mathcal{D}}_I \backslash \widehat{\mathcal{D}}_{IJ}), \qquad \forall J \in \widehat{\mathcal{N}}_I$
- $\mathcal{N}_i \cap \widehat{\mathcal{I}}_J = \emptyset, \qquad \forall J \in \widehat{\mathcal{N}}_I, \forall i \in \widehat{\mathcal{D}}_I \backslash \widehat{\mathcal{D}}_{IJ}$

Based on these facts, we can simplify the first term on the right hand side of equation 4.22:

$$\sum_{i\in\widehat{\mathcal{D}}_I}\sum_{j\in\mathcal{N}_i\backslash\widehat{\mathcal{I}}_I} f_{ij}^{n+1} = \sum_{i\in\widehat{\mathcal{D}}_I}\sum_{J\in\widehat{\mathcal{N}}_I}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} f_{ij}^{n+1} \tag{4.25}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I}\sum_{i\in\widehat{\mathcal{D}}_I}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} f_{ij}^{n+1} \tag{4.26}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I}\sum_{i\in\widehat{\mathcal{D}}_{IJ}}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} f_{ij}^{n+1} + \sum_{J\in\widehat{\mathcal{N}}_I}\sum_{i\in(\widehat{\mathcal{D}}_I\backslash\widehat{\mathcal{D}}_{IJ})}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} f_{ij}^{n+1}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I}\sum_{i\in\widehat{\mathcal{D}}_{IJ}}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} f_{ij}^{n+1}. \tag{4.27}$$

By using previous coarse scale parameter definitions, we can change the form of equation 4.25 in to coarse scale form:

$$\sum_{i\in\widehat{\mathcal{D}}_I}\sum_{j\in\mathcal{N}_i\backslash\widehat{\mathcal{I}}_I} f_{ij}^{n+1} \tag{4.28}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I}\sum_{i\in\widehat{\mathcal{D}}_{IJ}}\sum_{j\in(\mathcal{N}_i\cap\widehat{\mathcal{I}}_J)} \frac{\rho_r}{\mu}(1+c_l\triangle p_{ij}^{+n})T_{ij}(p_j^{n+1}-p_i^{n+1}) \tag{4.29}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I} \frac{\rho_r}{\mu}(1+c_l\triangle p_{IJ}^{+n})\widehat{T}_{IJ}(\widehat{p}_J^{n+1}-\widehat{p}_I^{n+1}). \tag{4.30}$$

Let us assume $(\widehat{\mathcal{I}}_I\backslash\widehat{\mathcal{D}}_I)\neq\emptyset$ at first. Recalling some definitions set previously, we notice that for $\forall i\in(\widehat{\mathcal{I}}_I\backslash\widehat{\mathcal{D}}_I)$, we have

- $\mathcal{N}_i\subseteq\widehat{\mathcal{I}}_I$ *and* $\mathcal{N}_i\cap\widehat{\mathcal{I}}_I=\mathcal{N}_i$
- $\forall j\in\mathcal{N}_i, \qquad i\in\mathcal{A}_j$
- $\forall j\in\mathcal{N}_i, \qquad p_{ij}^{+}=p_{ji}^{+}$ *and* $T_{ij}=T_{ji}$

For the last two terms on the right hand side of equation 4.22, we have

$$2\sum_{i\in\widehat{\mathcal{D}}_I}\sum_{j\in\mathcal{N}_i\cap\widehat{\mathcal{J}}_I} f_{ij}^{n+1} + 2\sum_{i\in\widehat{\mathcal{J}}_I\setminus\widehat{\mathcal{D}}_I}\sum_{j\in\mathcal{N}_i} f_{ij}^{n+1} \tag{4.31}$$

$$= 2\sum_{i\in\widehat{\mathcal{J}}_I}\sum_{j\in\mathcal{N}_i} f_{ij}^{n+1}$$

$$= \big[\sum_{i\in\widehat{\mathcal{J}}_I}\sum_{j\in\mathcal{N}_i}\frac{\rho_r}{\mu}(1+c_l\triangle p_{ij}^{+n})T_{ij}(p_j^{n+1}-p_i^{n+1})\big] + \tag{4.32}$$

$$\big[\sum_{j\in\widehat{\mathcal{J}}_I}\sum_{i\in\mathcal{A}_j}\frac{\rho_r}{\mu}(1+c_l\triangle p_{ji}^{+n})T_{ji}(p_i^{n+1}-p_j^{n+1})\big]$$

$$= \sum_{i\in\widehat{\mathcal{J}}_I}\sum_{j\in\mathcal{N}_i}\frac{\rho_r}{\mu}(1+c_l\triangle p_{ij}^{+n})T_{ij}[(p_j^{n+1}-p_i^{n+1})+(p_i^{n+1}-p_j^{n+1})] = 0. \tag{4.33}$$

For the condition when $(\widehat{\mathcal{J}}_I\setminus\widehat{\mathcal{D}}_I)=\emptyset$, since $\widehat{\mathcal{D}}_I\subseteq\widehat{\mathcal{J}}_I$, we have $\widehat{\mathcal{D}}_I=\widehat{\mathcal{J}}_I$. Thus, equation 4.31 still hold. Substituting equations 4.28 and 4.31 into equation 4.22, we obtain

$$\sum_{i\in\widehat{\mathcal{J}}_I}\sum_{j\in\mathcal{N}_i}\frac{\rho_r}{\mu}(1+c_l\triangle p_{ij}^{+n})T_{ij}(p_j^{n+1}-p_i^{n+1}) \tag{4.34}$$

$$= \sum_{J\in\widehat{\mathcal{N}}_I}\frac{\rho_r}{\mu}(1+c_l\triangle p_{IJ}^{+n})\widehat{T}_{IJ}(\widehat{p}_J^{n+1}-\widehat{p}_I^{n+1}). \tag{4.35}$$

For source and sink term, we have

$$\widehat{r}_I := \sum_{i\in\widehat{I}_I} r_i. \tag{4.36}$$

After substituting equations 4.17, 4.34 and 4.36 into equation 4.15, we obtain:

$$\frac{\rho_r\widehat{V}_I\widehat{\phi}_I c_t}{\triangle t}(\widehat{p}_I^{n+1}-\widehat{p}_I^{n}) = \sum_{J\in\widehat{\mathcal{N}}_I}\frac{\rho_r}{\mu}(1+c_l\triangle p_{IJ}^{+n})\widehat{T}_{IJ}(\widehat{p}_J^{n+1}-\widehat{p}_I^{n+1})+\widehat{r}_I. \tag{4.37}$$

Comparing with the fine scale discretized form of mass balance (equation 4.12), equation 4.37 is exactly the coarse scale discretized form of mass balance. □

4.4 Time step adjustment in unsteady state method

Since the unsteady state method proposed in previous sections contains time-steppings, the implementation issue of this method is discussed here. The linear solver used for solving the matrices in this research is conjugate gradient (CG), and the preconditioner of the matrices deploys incomplete LU (ilu0) factorization, both of which are implemented in PETSc linear algebra library ([5]). Here we propose a time-stepping algorithm 4 to adjust time steps and terminate iterations. Discussion of the effects of this algorithm is in Chapter 5.

Some given parameters as criteria are defined as follows:

- c_l, $\triangle t \in \mathbb{R}$: compressibility and initial time step.
- $\mathbf{itr}_{\max} \in \mathbb{N}$: maximum iteration number.
- $\kappa_{\min}$, $\kappa_{\max} \in \mathbb{R}$: minimum and maximum condition number allowed.
- $\triangle p_{\mathrm{c}} \in \mathbb{R}$: minimum pressure change allowed.

Some calculated parameters at each time step are defined as follows:

- $\triangle t^n$: time step size at nth step.
- $Nt_{\max}$: $t^{n+1} < t^n \times Nt_{\max}$.
- $\mathbf{inf}_T^n$: number of infinite transmissibility.
- $\triangle p_{\min}^n$: $\min_n(\widehat{p} - \widehat{p}_{\mathrm{init}})$.

- κ^n: approximated matrix condition number at step n.

Algorithm 4: newTimeStepSize

Input : parameters at iteration step n ($\mathbf{inf}_T^n, \triangle t^n, \triangle p_{\min}^n, \kappa^n$).
Output: terminating iteration or adjusted $\triangle t^{n+1}$ for iteration step n+1.

while $n < \mathbf{itr}_{\max}$ **do**
  if $\mathbf{inf}_T^n = 0$ **and** $\triangle p_{\min}^n > \triangle p_c^n$ **and** $\kappa^n < \kappa_{\max}$ **then**
    break
  else if $\kappa^n > \kappa_{\max}$ **then**
    $\triangle t^n \leftarrow \triangle t^n \times \frac{\kappa^n}{\kappa_{\max}}$, $n \leftarrow n$
  else if $\kappa^n < \kappa_{\min}$ **then**
    $\triangle t^{n+1} \leftarrow \triangle t^n \times \min(Nt_{\max}, \frac{\kappa_{\max}+\kappa_{\min}}{2\kappa^n})$, $n \leftarrow n+1$
  else
    $\triangle t^{n+1} \leftarrow \triangle t^n$, $n \leftarrow n+1$
return $\triangle t^{n+1}$

Chapter 5

Numerical Tests for Flow Simulation of Fractured Systems

In this chapter, numerical experiments with different fracture patterns and well conditions are presented to verify the introduced upscaling method in Chapter 4.

For fracture patterns, test 1 and test 2 use short and randomly distributed fractures, test 3 deploys long and well-connected fractures, and test 4 is embedded with large hydraulic fractures and large number of short natural fractures either intersected or separated with hydraulic fractures. As for well configurations, test1 employs both injection and production wells to verify the efficiency of the general upscaling method, test 2 and test 4 only use vertical production wells to demonstrate the efficiency of the unsteady state flow problem and corresponding algorithm. Besides, a five-spot well pattern is presented in case 3 and a horizontal production well is deployed in test 4 to illustrate the advantages of the multi-level partitioning method.

All numerical tests performed in this section are simulated using a black-oil simulator with three phase flow. The PVT properties under the reference pressure are listed in Table 5.1. PVT properties of oil and gas as the

Table 5.1: Fluid properties under reference pressure pressure p_{ref}.

p_{ref} (kPa)	ρ_w (kg/m^3)	ρ_o (kg/m^3)	ρ_g (kg/m^3)	μ_w (10^{-3}Pa $\cdot$ s)	c_w (kPa^{-1})	c_o (kPa^{-1})
101.325	947.516	814.612	1.125	0.96	5.40×10^{-7}	1.45×10^{-6}

Table 5.2: Oil and gas PVT properties as functions of pressure (r_s is the gas-oil ration, e_g is the gas expansion factor, μ_o and μ_g are oil and gas viscosity).

p (kPa)	r_s (m^3/m^3)	e_g (m^3/m^3)	μ_o (10^{-3}Pa $\cdot$ s)	μ_g (10^{-3}Pa $\cdot$ s)
103.4	1.0010	0.7306	0.9030	0.0136
2289.1	13.9532	16.9861	0.7190	0.0139
4481.6	29.9993	34.0841	0.5970	0.0144
7770.4	57.0740	61.0608	0.4840	0.0154
12148.6	96.8836	98.3311	0.3940	0.0171
14341.1	117.9916	116.8294	0.3630	0.0181
27847.9	260.5732	212.9748	0.2550	0.0256
39162.2	392.0799	266.5061	0.2100	0.0319
51710.7	565.4125	312.2136	0.1780	0.0390
55158.1	616.1920	321.0613	0.1750	0.0412
58605.5	668.5710	328.3629	0.1700	0.0435

function of pressure are showen in Table 5.2. Relative permeability curves of different phases are showed in Figure 5.1.

The errors of coarse scale simulation results are evaluated based on certain fine scale simulation as reference solution. Here we introduce two parameters to represent pressure error of the coarse solution. For a given time $t \in (0, T]$, we denote $\widehat{p}_I^t$ as the pressure of the coarse grid I at time t, and $\widehat{\mathbf{p}}^t$ as the set consisted of pressure at time t of all the coarse grids. Similarly, $\overline{p}_I{}^t$

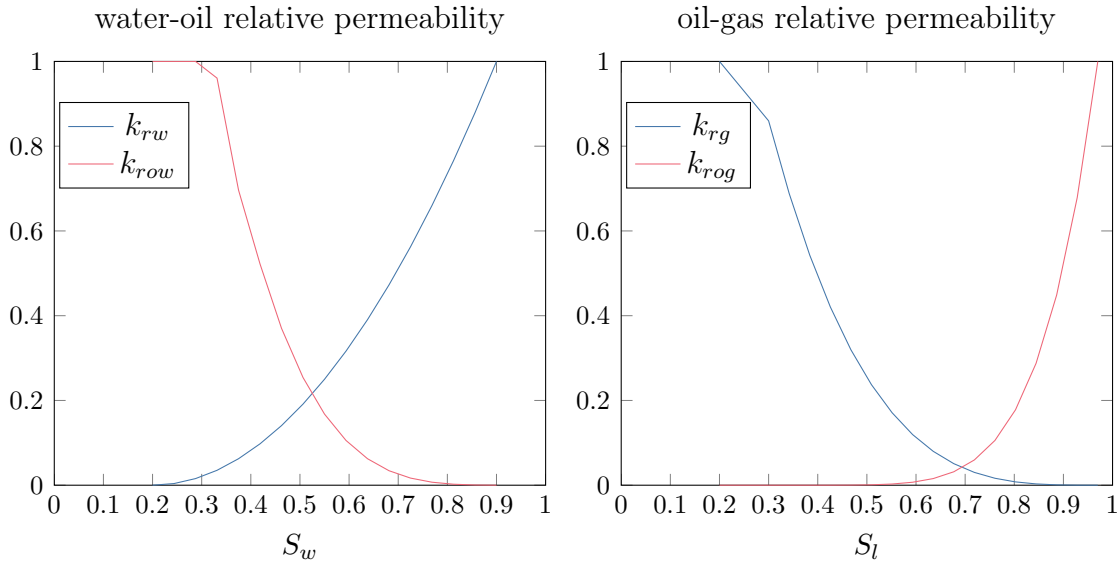


Figure 5.1: Relative permeability curves used for numerical tests.

is introduced to represent the fine scale pressure at time t which is averaged based on the volume of fine grids included in the coarse grid I, and $\overline{\mathbf{p}}^t$ is to used as the set consisted of averaged pressure at time t of all coarse grids.

As for errors, relative pressure error of coarse grid I at time t is defined as

$$\epsilon_{p,I}^t = \frac{|\widehat{p}_I^t - \overline{p}_I{}^t|}{\overline{p}_I{}^t} \qquad t \in (0, T], \tag{5.1}$$

and relative pressure error over the whole reservoir domain is defined as

$$\epsilon_p^t = \frac{\|\widehat{\mathbf{p}}^t - \overline{\mathbf{p}}^t\|_{\ell_2}}{\|\overline{\mathbf{p}}^t\|_{\ell_2}} \qquad t \in (0, T]. \tag{5.2}$$

5.1 Test 1: a heterogeneous reservoir with natrual fractures and without source term

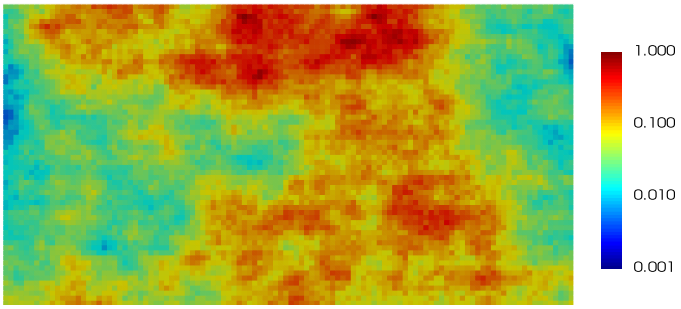
5.1.1 Setup

We consider a $1100 \times 600 \times 80\,\mathrm{ft}^3$ heterogeneous matrix domain in this case. 1000 randomly generated fractures with average length of 15 m exit in this reservoir model, which is shown in Figure 5.2. A production well with constant pressure of 1000 psi is located at the middle part of the reservoir. All the boundaries in this case are considered as no-flow boundaries. Parameters for fracture domain and matrix domain are listed in Table 5.3.

The ground truth solution is obtained from the case with fine grids of size $10 \times 10\,\mathrm{ft}^2$, and the number of which is 6600 for matrix and 11389 for fractures. The coarse simulation is deployed for cases with and without fractures, for grids of number as 7×10, 9×13 and 15×20 respectively.

The unsteady state method is used for the hypothetical single phase flow problem and greedy aggregation method is deployed to partition the coarse fracture grids.

(a) Fracture distribution
(green triangle represents the production well)



(b) Permeability distribution
(modified from 28th layer of SPE 10th problem)

Figure 5.2: 2D reservoir model of test 1.

5.1.2 Results

Oil production results are showed in Figure 5.3. From fine solution results in (a), heterogeneity and natural fractures both have effects on the simulation results. Same production profiles obtained using different time in (b) for the unsteady state problem indicate that the long time simulation time is unnecessary in this case.

Table 5.3: The reservoir model parameters for flow simulation of test 1.

Domain	Parameter	Value
Fracture	number	1000
	permeability (mD)	10000
	aperture (m)	0.03
	average length (m)	15
Matrix	porosity	0.05
	initial pressure (psi)	8000
	compressibility (psi^{-1})	1×10^{-6}
Well	pressure (psi)	1000

The accuracy of the reduced models is verified from production profile, pressure error and saturation error in different scales. Figure 5.3 (c) and (d) indicate that all the three different scales of coarse simulations can match well with the ground truth oil production and rate profiles. Indeed, the relative error of final production volume are 2.8E-2 for 1% grids, 1.7E-2 for 2% grids, and 3.0E-3 for 5% grids respectively. Figure 5.4 presents the relative pressure error after 1 year production. The highest relative pressure error is less than 5% in all scales, and pressure error is reduced as the number of grids increases. Besides, highest pressure error happens at the production region, which is reasonable because no extra upscaling manipulation is employed for the near-well regions. Absolute saturation error after 1 year production showed in Figure 5.5 also indicates the accuracy of the coarse models.

From fine solution results, we can observe that existence of fractures in this case enhances the oil an gas productions. Comparison results between

Table 5.4: Comparison of simulation errors and CPU time of Test 1 between cases with and without fractures for different scales at different simulation times. t_1 represents 1 year and t_2 represents 10 years.

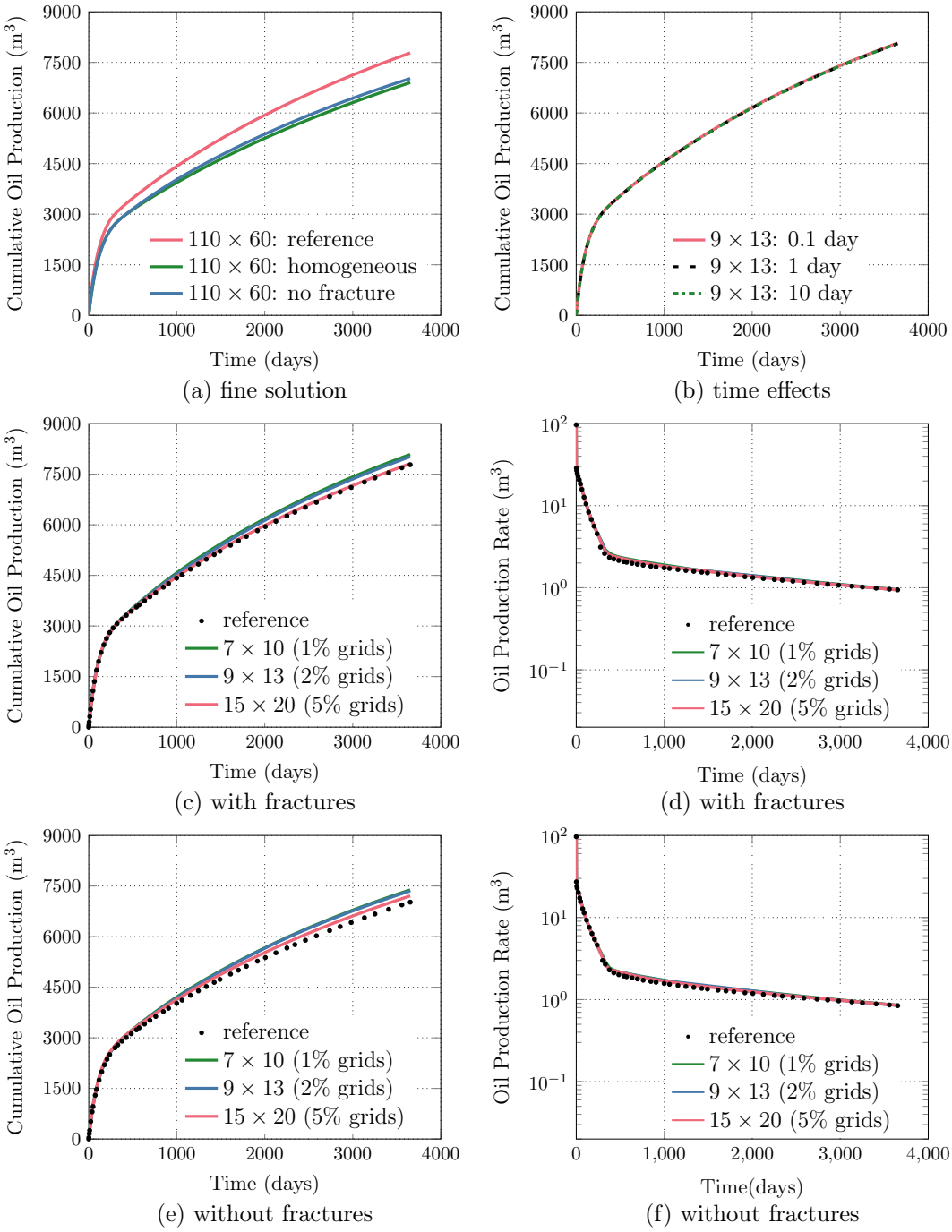
Scale	$\epsilon_{p,\ell_2}^{t_1}$	$\epsilon_{p,\ell_\infty}^{t_1}$	$\epsilon_{p,\ell_2}^{t_2}$	$\epsilon_{p,\ell_\infty}^{t_2}$	$\epsilon_{S_o,\ell_2}^{t_1}$	$\epsilon_{S_o,\ell_\infty}^{t_1}$	$\epsilon_{S_o,\ell_2}^{t_2}$	$\epsilon_{S_o,\ell_\infty}^{t_2}$	time (s)
				$\mathbf{N_f} = 1000$					
7×10	8.3E-3	2.5E-2	3.1E-3	3.8E-2	7.0E-3	1.3E-2	8.5E-2	1.4E-2	0.44
9×13	6.5E-3	2.8E-3	3.2E-3	2.7E-2	5.4E-3	1.3E-2	6.6E-3	1.4E-2	0.50
15×20	6.5E-3	1.3E-2	3.6E-3	2.4E-2	4.7E-3	9.9E-3	5.4E-3	9.7E-3	0.73
				$\mathbf{N_f} = 0$					
7×10	4.6E-3	7.5E-3	3.3E-2	3.7E-2	7.6E-3	1.3E-2	9.1E-3	1.3E-2	0.31
9×13	4.6E-3	7.4E-3	3.0E-2	3.5E-2	7.2E-3	1.2E-2	8.2E-3	1.2E-2	0.41
15×20	4.5E-3	6.5E-3	2.9E-2	3.4E-2	5.9E-3	1.2E-2	6.8E-3	1.0E-2	0.60

cases with and without fractures in Figure 5.3 and Table 5.4 show that the error of production, pressure and oil saturation distribution are nearly in the same magnitude, which indicate that fracture existence of natural fractures does not largely influence the upsclaing effects.

As for the efficiency, the results for different scales are showed in Table 5.5. The speedup of reduced model is significant since the coarsest model is 156 times faster than the fine scale simulation, which demonstrate the efficiency of the upscaling methodology introduced in Chapter 4.

Table 5.5: Convergence results of Test 1 (t_1 represents 1 year and t_2 represents 10 years. #m is the number of matrix grids, #f is the number of fracture grids, and #dof is the number of total grids.)

Scale	Accuracy					Efficiency				
	$\epsilon_{p,\ell_2}^{t_1}$	$\epsilon_{p,\ell_\infty}^{t_1}$	$\epsilon_{S_o,\ell_2}^{t_2}$	$\epsilon_{S_o,\ell_\infty}^{t_2}$	$\epsilon_{V_o}^{t_2}$	#m	#f	#dof	time	speedup
110×60	-	-	-	-	-	6600	4789	11389	68.64	-
7×10	4.5E-3	3.4E-2	1.0E-2	2.1E-2	2.8E-2	70	68	138	0.44	156
9×13	4.0E-3	1.7E-2	8.0E-3	1.9E-2	1.7E-2	117	115	232	0.50	137
15×20	3.7E-3	1.5E-2	5.3E-3	1.1E-2	3.0E-3	300	296	596	0.73	94

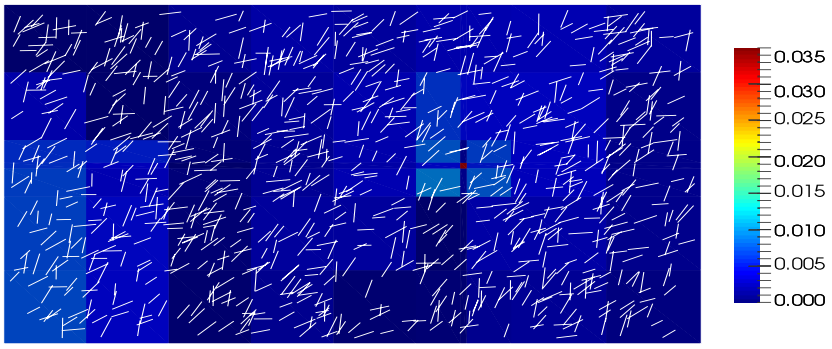


(a) fine solution (b) time effects

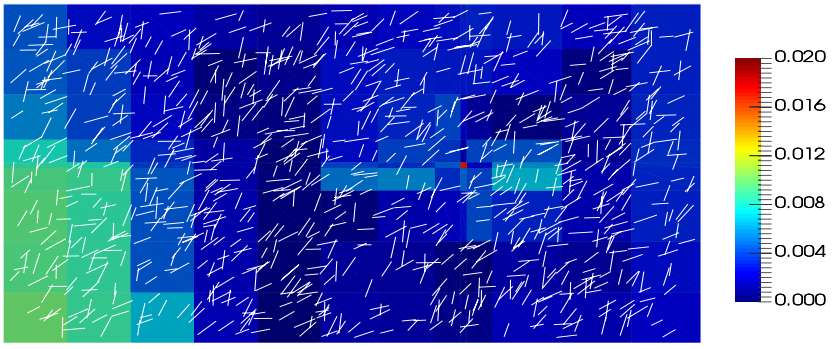
(c) with fractures (d) with fractures

(e) without fractures (f) without fractures

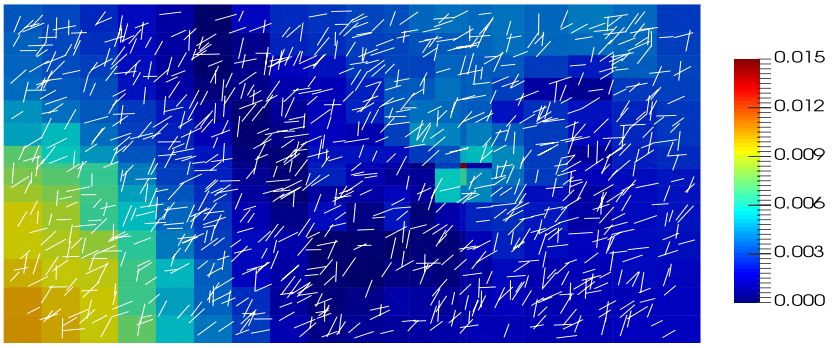
Figure 5.3: Oil production results of test 1: time effects in unsteady state problem is presented in (b), results of case with fractures is showed in (c) and (d), results of case without fractures is showed in (e) and (f).



(a) 7×10 (1% grids)

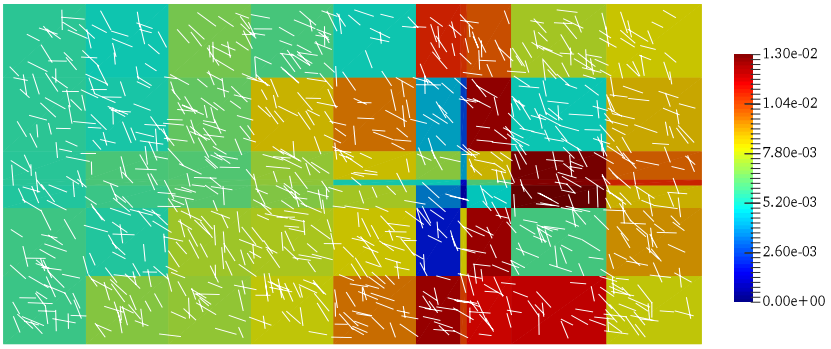


(b) 9×13 (2% grids)

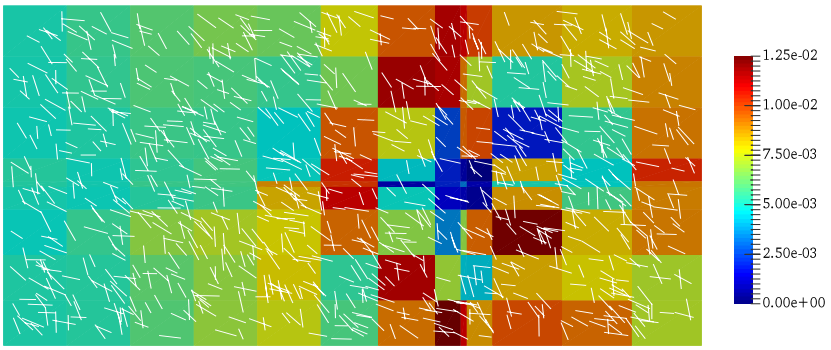


(c) 15×20 (5% grids)

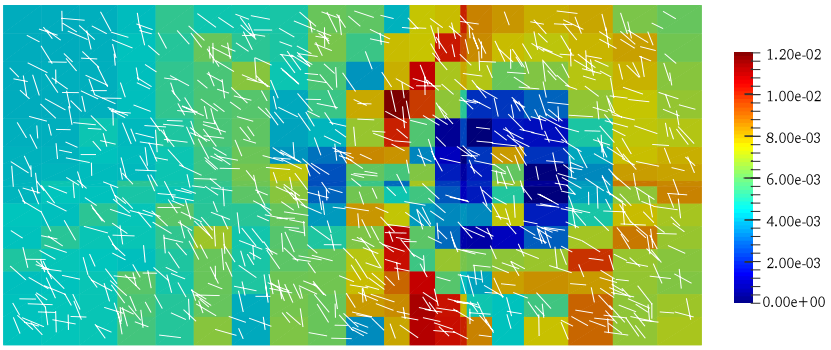
Figure 5.4: Relative pressure error of Test 1 after 1 year production.



(a) 7×10 (1% grids)



(b) 9×13 (2% grids)



(c) 15×20 (5% grids)

Figure 5.5: Absolute oil saturation error of Test 1 after 1 year production.

5.2 Test 2: a natural fractured reservoir with injection and production wells

5.2.1 Setup.

We consider a $1010 \times 1010 \times 80\,\mathrm{ft}^3$ homogeneous matrix domain in this case. 200 randomly generated fractures with average length of 100 ft embedded in this reservoir model, which is shown in Figure 5.6. An injection well injecting water under constant pressure of 8000 psi is located at the lower left corner of the reservoir model, and a production well with constant pressure of 1000 psi is located at the upper right corner of the reservoir. All the boundaries in this case are considered as no-flow boundaries. Parameters for fracture domain and matrix domain are listed in Table 5.6.

The ground truth solution is obtained from the case with fine grids of size $7.14 \times 7.14\,\mathrm{ft}^2$, and the number of which is 20164 for matrix and 3787 for fractures. The coarse simulation is deployed for cases with and without fractures, for grids of number as 12×12, 16×16 and 28×28 respectively. Correspondingly, the coarse scales are consisted of 1%, 2% and 5% grids of the fine scale solution.

Since both production and injection wells exist in this case, steady state problem could be used. Simulations are conducted either under steady or unsteady state methods, with greedy partitioning or local partitioning methods.

Figure 5.6: 2D reservoir model of case 2.

Domain	Parameter	Value
Fracture	number	200
	permeability (mD)	10000
	aperture (m)	0.03
	average length (m)	30.48
Matrix	porosity	0.05
	permeability (mD)	0.1
	initial pressure (psi)	5000
	compressibility (psi^{-1})	1×10^{-6}

Table 5.6: Parameters of fracture domain and matrix domain in case 2.

5.2.2 Results

Figure 5.7 shows the cumulative oil production of Test 2 of three coarse scales under four different upscaling methods, respectively. Figure 5.8 presents the convergence rate of the steady state and unsteady state problem of single time step under cases with and without fractures. Table 5.8 compare results of pressure error, saturation error and final recovery error between greedy and local partitioning of steady state method.

Greedy partitioning vs. local partitioning. From Figure 5.7, we can see that greedy partitioning shows better production profile than local method in both steady and unsteady state tests as opposed to the ground truth solution. Table 5.8 shows that the error are comparable between greedy and local partitioning, but greedy method has smaller error mostly. From Figure 5.9, the

greedy method clearly has better pressure distribution after 1 year production of oil and gas. Besides, local partitioning has larger number of embedded fracture grids, thus with more simulation time since each fracture segment will generate an extra fracture grid.

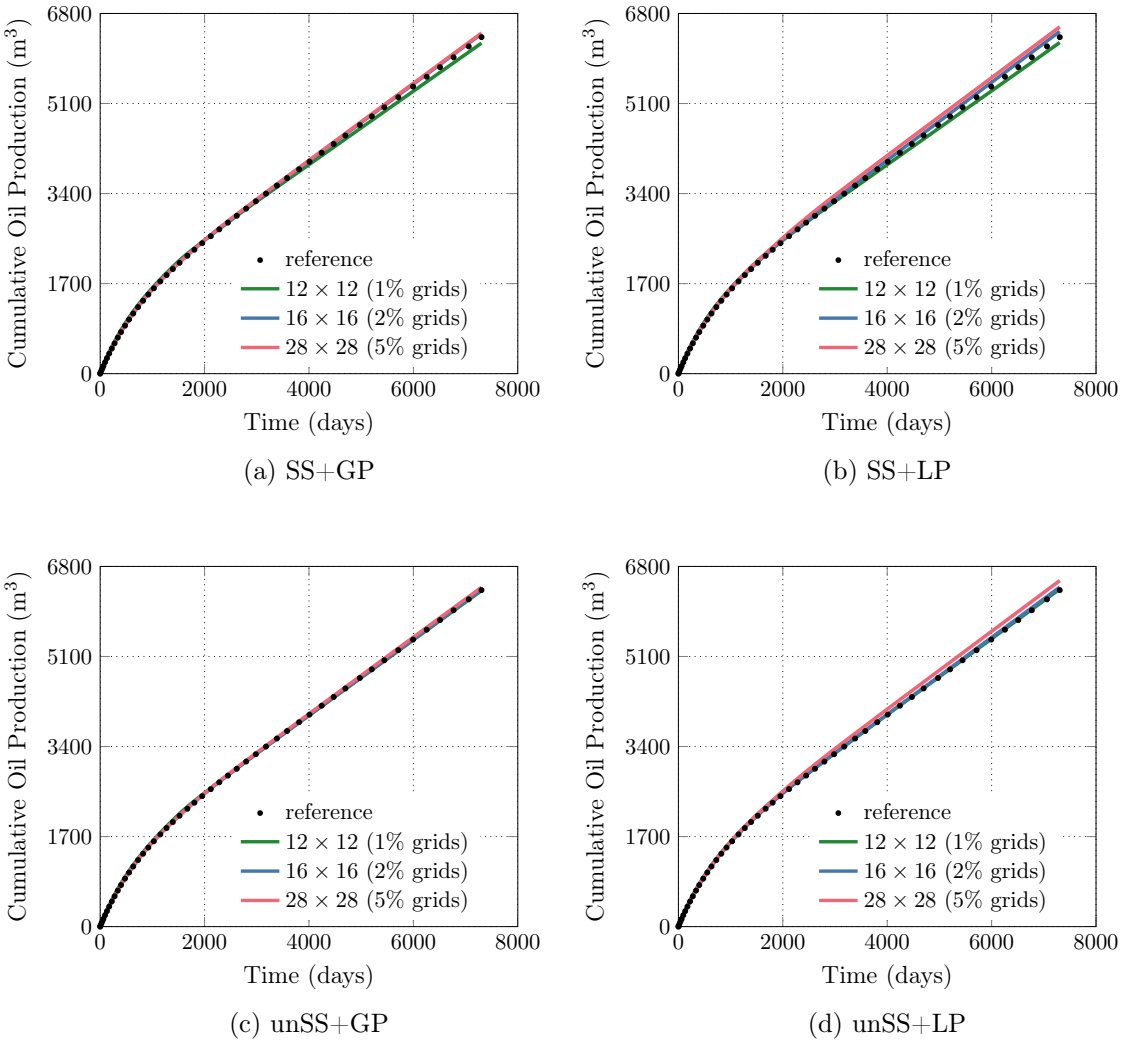
Generally, we can conclude that greedy partitioning works better under condition with numerous natural fractures separately distributed in clusters. We should note that the better performance of greedy partitioning is not always the case for all the fracture patterns, which will be observed in Test 3 and Test 4.

Steady state method vs. unsteady state method. From Figure 5.7, unsteady state method matches the ground truth solution more accurate than the steady state method, though they are comparable with each other and the difference between them is small. From 5.9, the pressure error difference of steady state method and unsteady state method is negligible. Therefore, we can conclude that the unsteady state method generate better production results, but not better pressure profiles.

But unsteady state method has a great advantage of efficiency. From Figure 5.8, unsteady state problem has much higher convergence rate than steady state method since steady state problem has residual of 1.E-6 after 300 iterations, while unsteady state problem could converge to 1.E-10 after only 11 iterations (note that these two problem has same number of dof, thus the iteration time is comparable with the iteration steps). This problem is

especially important for systems with fractures, since the iteration number of the case without fractures are very similar between these two problems. The reason for this phenomenon is that existence of fractures has large effects on the matrix structures, which increase the ill conditioning of the steady state case.

As for the efficiency, the upscaling effects is significant in this test. From Table 5.7, the coarse model of scale 12×12 (1% grids) could achieve a speedup of 19000, with only 0.7% cumulative production error after 20 years and 1.4% ℓ_∞ relative pressure error.



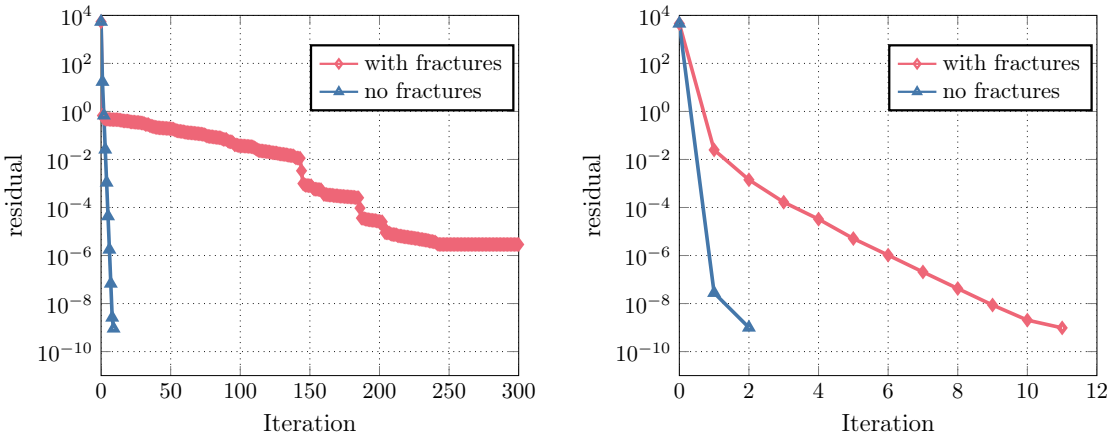
(a) SS+GP

(b) SS+LP

(c) unSS+GP

(d) unSS+LP

Figure 5.7: Cumulative Oil production results of reduced models in Test 2. "SS" represents steady state method, and "unSS" means unsteady state method. "GP" represents greedy partitioning and "LP" represents local partitioning.



(a) Steady state problem

(b) Unsteady state problem

Figure 5.8: Convergence history of steady state and unsteady state problems with or without fractures by using CG as linear solver and ilu0 as preconditioner (the unsteady state problem is solved in one time step with $\frac{\rho_L}{\Delta t} = 2.5\text{E-}6$).

Table 5.7: Convergence results of Test 2 using unsteady state problem and greedy partitioning.

Scale	Accuracy					Efficiency				
	$\epsilon^{t_1}_{p,\ell_2}$	$\epsilon^{t_1}_{p,\ell_\infty}$	$\epsilon^{t_2}_{S_o,\ell_2}$	$\epsilon^{t_2}_{S_o,\ell_\infty}$	$\epsilon^{t_2}_{V_o}$	#m	#f	#dof	time	speedup
142×142	-	-	-	-	-	20164	3787	23951	2505.59	-
12×12	3.0E-2	1.4E-1	2.8E-1	1.3E-1	6.8E-3	144	89	233	0.13	1.9E+4
16×16	2.5E-2	8.0E-2	2.8E-1	8.5E-2	5.6E-3	256	192	448	0.69	3.6E+3
26×26	2.0E-2	8.0E-2	4.2E-1	1.5E-1	8.2E-3	676	412	1088	1.86	1.3E+3

Table 5.8: Comparison of pressure and oil saturation errors between coarse models with greedy and local partitioning by using steady state problem.

Scale	Greedy Partitioning					Local Partitioning				
	$\epsilon^{t_1}_{p,\ell_2}$	$\epsilon^{t_1}_{p,\ell_\infty}$	$\epsilon^{t_2}_{S_o,\ell_2}$	$\epsilon^{t_2}_{S_o,\ell_\infty}$	$\epsilon^{t_2}_{V_o}$	$\epsilon^{t_1}_{p,\ell_2}$	$\epsilon^{t_1}_{p,\ell_\infty}$	$\epsilon^{t_2}_{S_o,\ell_2}$	$\epsilon^{t_2}_{S_o,\ell_\infty}$	$\epsilon^{t_2}_{V_o}$
12×12	2.0E-2	8.7E-2	2.6E-1	1.4E-1	3.4E-2	2.1E-2	8.9E-2	2.7E-1	1.4E-1	2.9E-2
16×16	2.3E-2	8.9E-2	3.0E-1	1.2E-1	4.0E-2	1.6E-2	6.4E-2	2.6E-1	1.1E-1	5.8E-2
26×26	1.7E-2	7.9E-2	3.9E-1	1.6E-1	1.6E-2	1.9E-2	8.0E-2	3.3E-1	9.7E-2	3.2E-2

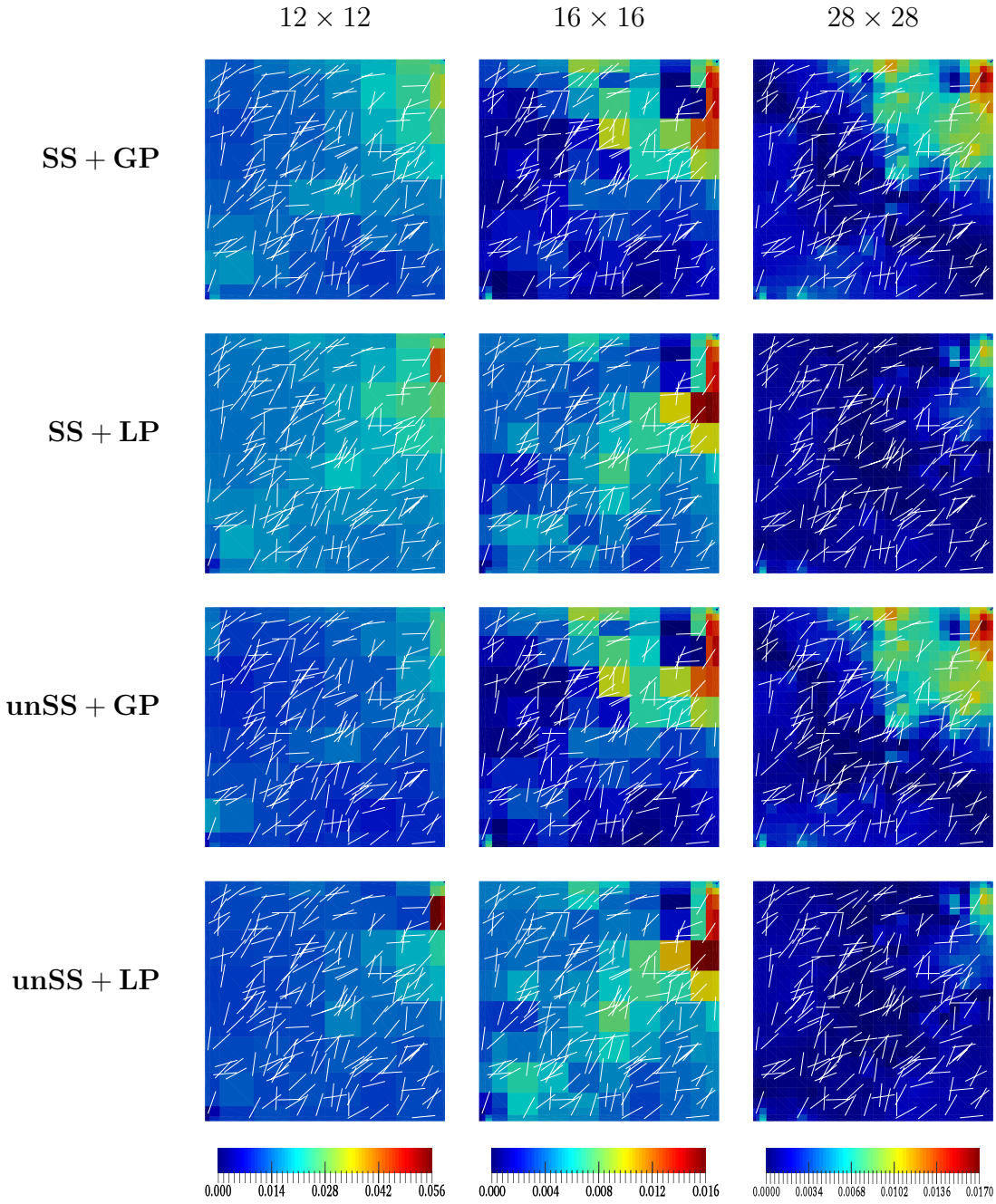


Figure 5.9: Relative pressure error of Test 2 by using unsteady state problem or steady state problem, and greedy partitioning method or local partitioning method after 1 year production.

5.3 Test 3: a five-spot case with large connected fractures

5.3.1 Setup

We consider a $1010 \times 1010 \times 80\,\mathrm{ft}^3$ homogeneous matrix domain in this case. 70 randomly generated large fractures embedded in this reservoir model, which is shown in Figure 5.10. An injection well injecting water under constant pressure of 8000 psi is located at the lower left corner of the reservoir model, and 4 production wells with constant pressure of 1000 psi is located at the four corners of this reservoir. All the boundaries in this case are considered as no-flow boundaries. Parameters for fracture domain and matrix domain are listed in Table 5.3.

The ground truth solution is obtained from the case with fine grids of size $10 \times 10\,\mathrm{ft}^2$, and the number of which is 10201 for matrix and 2585 for fractures. The coarse simulations are all conducted on the base of matrix grids with number as 21×21.

Since both production and injection wells exist in this case, steady state problem could be used. Coarse models are all generated from steady state method in this test. To study the partitioning approaches, we conduct simulation under greedy, local and multi-level partitioning methods. Figure 5.11 (a) shows the number of intersecting matrix cells of each fracture. The largest number of intersected cells is 20. (b) and (c) showed fractures which intersect no less than 7 and 10 matrix grids (denoted as "L=7" and "L=10", respectively).

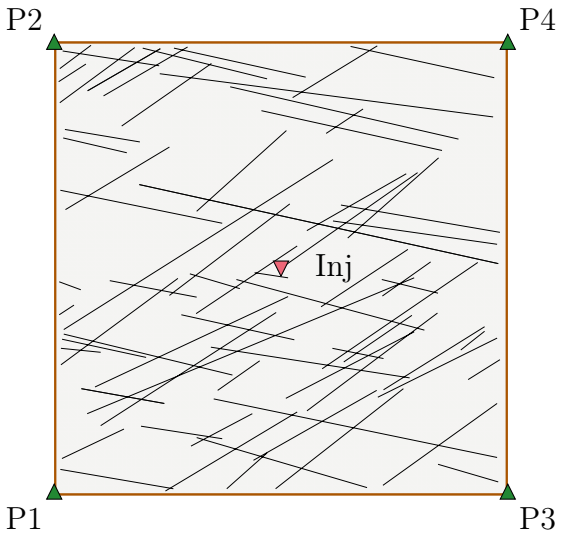
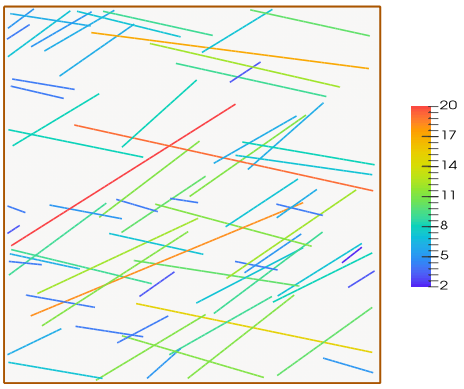


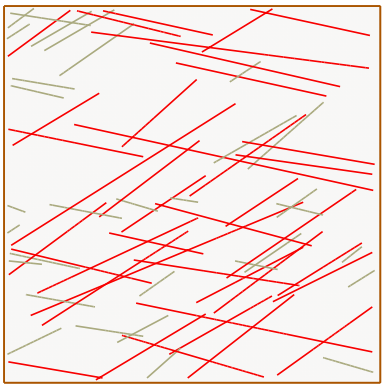
Figure 5.10: 2D reservoir model of Test 3.

Table 5.9: Parameters of fracture domain and matrix domain in Test 3.

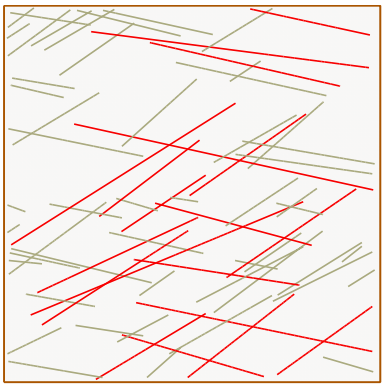
Domain	Parameter	Value
Fracture	number	70
	permeability (mD)	10000
Matrix	porosity	0.05
	permeability (mD)	0.1
	initial pressure (psi)	8000
	compressibility (psi^{-1})	1.0×10^{-6}



(a) Length of fractures



(b) $L = 7$



(c) $L = 10$

Figure 5.11: Pressure profile of the reference solution and relative pressure error of upscaling results by using unsteady state problem and greedy partitioning after 1 year production.

5.3.2 Results

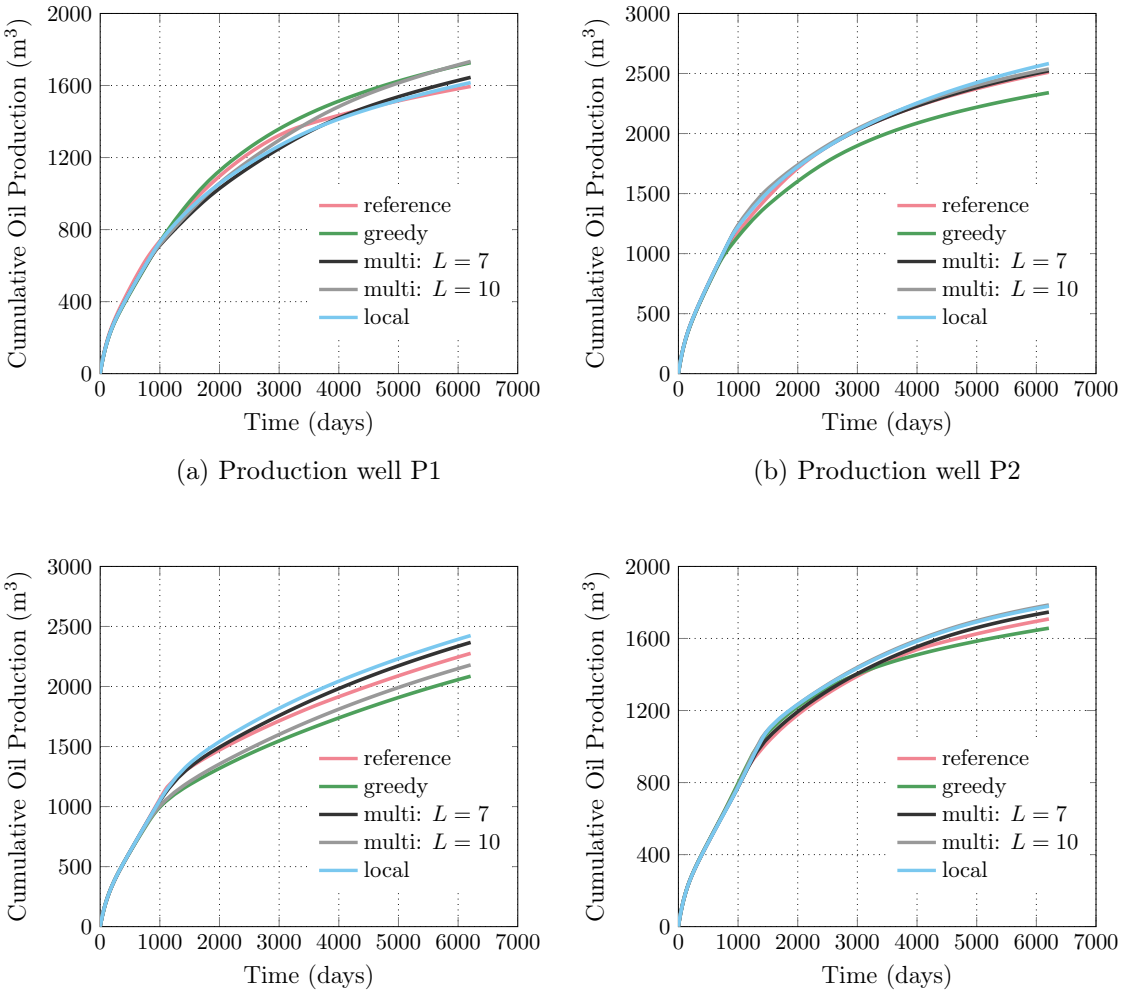
Figure 5.12 presents the oil production results of different partitioning methods in 4 production well respectively. Figure 5.13 displays fine pressure profile and relative pressure errors for different methods. Figure 5.14 shows the saturation contours of fine simulation results and coarse model simulation results.

Greedy partitioning vs. local partitioning. From Figure 5.12, local method works better than greedy method at production wells P1, P2 and P4. Produciton results of local partitioning method are acceptable at these three wells. From Figure 5.13, the pressure error of greedy method is relatively large. The reason why greedy partitioning fails at this test is that lots of fractures are very long and intersected with each other, from which small error could be propagated to the whole reservoir, and thus impair the production accuracy of all the four wells. For production well P3, all the simulation results are more or less deviated form the ground truth solution because there is one fracture oriented directly to the production well. Simulating such kind of fractures are not easy, thus extra upscaling manipulations should be added for the near well regions. But this is far from the objective of the study in this discussion. In conclusion, the greedy partitioning method is not enough for such systems with long fractures intersected with each other.

Local partitioning vs. multi-level partitioning. For multi-level partitioning method, we can observe that results of case with "$L = 10$" is similar to the local partitioning, and the results of the case with "$L = 7$" is more similar to the greedy one. This indicates that setting $L = 10$ is not enough in this test. We should note that setting the number of aggregation attribute "L" is dependent on the case, which is a disadvantage of this method. But practically, several setting could be implemented till the results do not change largely. From Figure 5.13 and Figure 5.14, the multi-level partitioning generate smallest error in pressure and most similar saturation contour shapes compared to the ground truth solution. The errors are logged in Table 5.10, from which we could conclude that multi-level partitioning with $L = 7$ has smallest error of pressure, and generates less fracture grids, and thus faster than the local method.

Table 5.10: Convergence results of Test 3: t_1 represents after 1 year production, t_2 represents after 5 years of production ("multi-1" represents case with "$L = 7$", "multi-2" represents case with "$L = 10$").

Method	Accuracy				Efficiency				
	$\epsilon^{t_1}_{p,\ell_2}$	$\epsilon^{t_1}_{p,\ell_\infty}$	$\epsilon^{t_2}_{S_o,\ell_2}$	$\epsilon^{t_2}_{S_o,\ell_\infty}$	#m	#f	#total	time	speedup
reference	-	-	-	-	10201	2585	12786	183.86	-
greedy	3.8E-2	1.3E-1	1.7E+0	3.4E-1	441	279	720	2.54	7.2E+1
multi-1	2.5E-2	7.4E-2	1.6E+0	3.6E-1	441	367	808	3.67	5.0E+1
multi-2	3.4E-2	1.3E-1	1.6E+0	3.8E-1	441	346	787	3.49	5.3E+1
local	2.9E-2	7.5E-2	1.6E+0	3.9E-1	441	434	875	4.1	4.5E+1



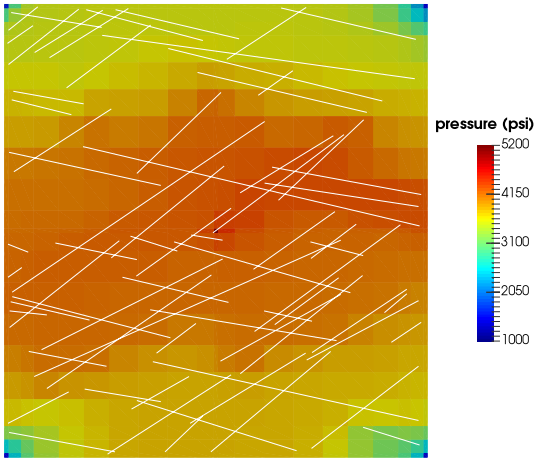
(a) Production well P1

(b) Production well P2

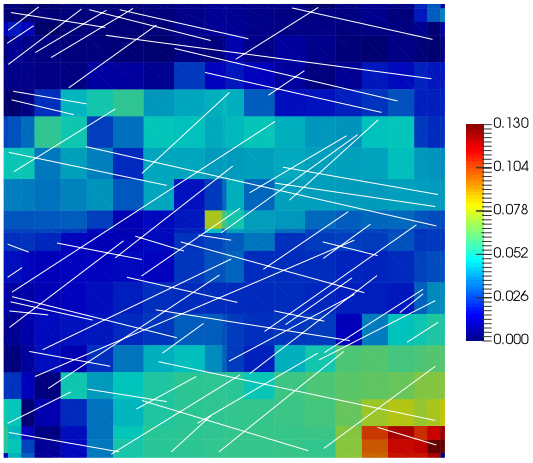
(c) Production well P3

(d) Production well P4

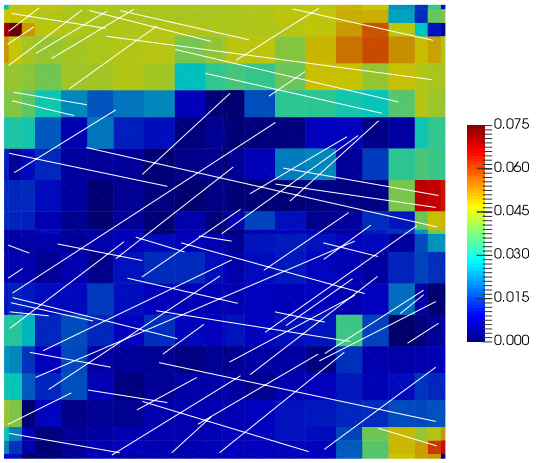
Figure 5.12: Cumulative Oil production results of Test 3.



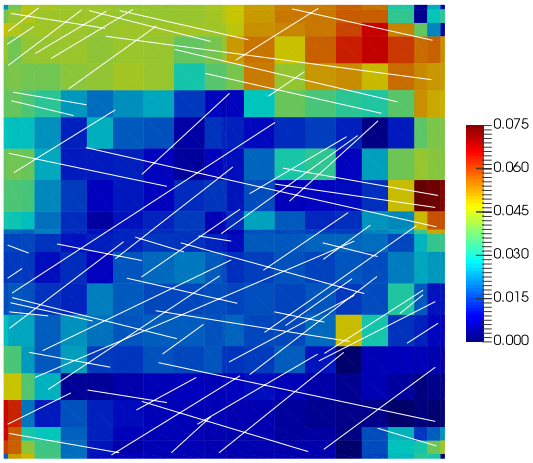
(a) Averaged fine result:



(b) Coarse: greedy partitioning

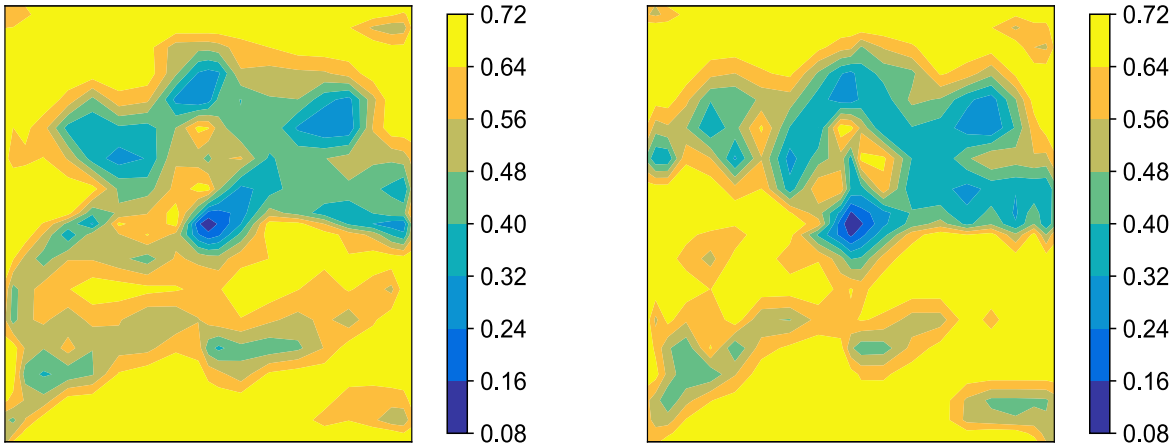


(c) Coarse: multi-level partitioning



(d) Coarse: local partitioning

Figure 5.13: Pressure profile of the reference solution (a) and relative pressure errors (b)-(d) of coarse models generated from different partitioning methods after 1 year production.



(a) Averaged fine result

(b) Coarse: greedy partitioning

(c) Coarse: multi-level partitioning

(d) Coarse: local partitioning

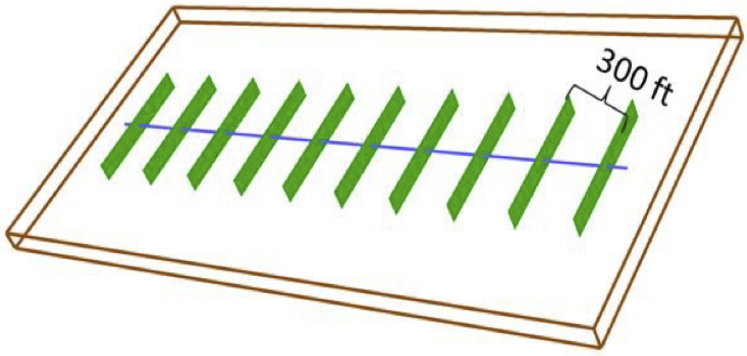
Figure 5.14: Saturation profile of the reference solution (a) and relative pressure errors (b)-(d) of coarse models generated from different partitioning methods after 5 years of production.

5.4 Test 4: a hydraulic fractured reservoir with complex fracture networks

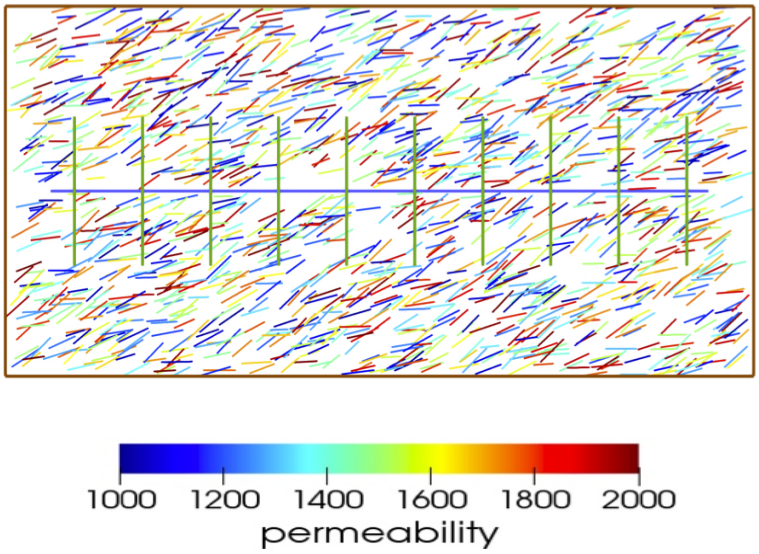
5.4.1 Setup

We consider a $3300 \times 1510 \times 80\,\mathrm{ft}^3$ homogeneous matrix domain in this test. 10 long hydraulic fractures go through the entire reservoir with 300 ft distance between each other. Besides, 1500 natural fractures are randomly generated with different permeability (see Figure 5.15. Note that some of these natural fractures are intersected with hydraulic fractures, which generate complex fracture networks over the entire reservoir domain. For matrix background, we deployed porosity of 0.05 and permeability of 0.001 mD to simulate the production of shale oil reservoirs. A horizontal production well through 10 hydraulic fractures is located at the middle of the reservoir. All the boundaries in this case are considered as no-flow boundaries. Parameters for fracture domain and matrix domain are listed in Table 5.11.

The ground truth solution is obtained from the case with fine grids of size $10 \times 10 \times 80\,\mathrm{ft}^3$, and the number of which is 49830 for matrix and 23935 for fractures. Simulations of coarse models are all conducted on the base of matrix grids with number as 30×13. Note that all multi-level coarse models have the aggregation attribute of $L = 61$, which indicate that only the 10 hydraulic fractures are treated as long fractures. Since only production and no flow boundaries exit in this test, unsteady state problem should be used to generate all the coarse models.



(a) 3D model of hydraulic fractures distribution in Test 4



(b) 2D model of natural fractures and their permeability distribution in Test 4

Figure 5.15: Reservoir models of Test 4.

Table 5.11: Parameters of fracture domain and matrix domain in Test 3.

Domain	Parameter	Value
Hydraulic Fractures	number	10
	permeability (mD)	10000
	aperture (ft)	0.1
	length (ft)	610
Natural Fractures	number	1500
	permeability (mD)	1000-2000
	aperture (ft)	0.001
	length (ft)	80-150
Matrix	porosity	0.05
	permeability (mD)	0.001
	initial pressure (psi)	8000
	production pressure (psi)	1000
	initial water saturation	0.17
	compressibility (psi^{-1})	1.0×10^{-6}

5.4.2 Results

5.4.2.1 Efficiency of the time-stepping algorithm

Figure 5.16 presents results of coarse models generated by unsteady state method with only 1 time step using $\frac{c_l}{\triangle t} = 5.0E-8$ and $\frac{c_l}{\triangle t} = 1.0E-9$ respectively. By setting $\frac{c_l}{\triangle t} = 5.0E-8$, pressure changes are very small after 1 time step, and production result deviate a lot from the ground truth solution. This indicate that insufficient pressure change in the unsteady state problem would generate bad transmissibilities for the reduced models. After setting $\frac{c_l}{\triangle t} = 1.0E-9$ as a small value, though pressure changes are sufficient, the pressure solution is wrong even we used implicit scheme, and the transmissibil-

ities are also far from the truth, which will generate wrong production profile. Thus, it is important to generate sufficient pressure change and adjust value of $\frac{c_l}{\triangle t}$ in the unsteady state method.

By using the time-stepping algorithm proposed in Chapter 4, we generate new coarse models by setting $\frac{c_l}{\triangle t} = 5.0E-6$, $\frac{c_l}{\triangle t} = 5.0E-7$, and $\frac{c_l}{\triangle t} = 5.0E-8$ respectively, and the production results of which are displayed in Figure 5.17. The accuracy of this algorithm can be observed from the figure since results of all these three cases match the fine solution very well. The efficiency of the time-stepping algorithm is revealed in Table 5.12. The termination criteria of this algorithm determine that sufficient pressure changes can be achieved. Furthermore, adjustment of time steps in this algorithm can control the total CG iterations. We note that though this algorithm has an advantage of setting random values for $\frac{c_l}{\triangle t}$, ill-conditioning caused by small values should be avoided.

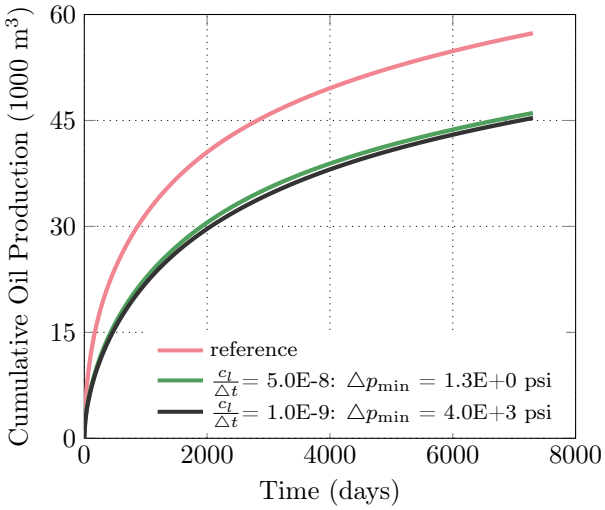


Figure 5.16: Importance of choosing ratio of compressibility over time step using multi-level partitioning method on a coarse model (results represent models generated by only one time step).

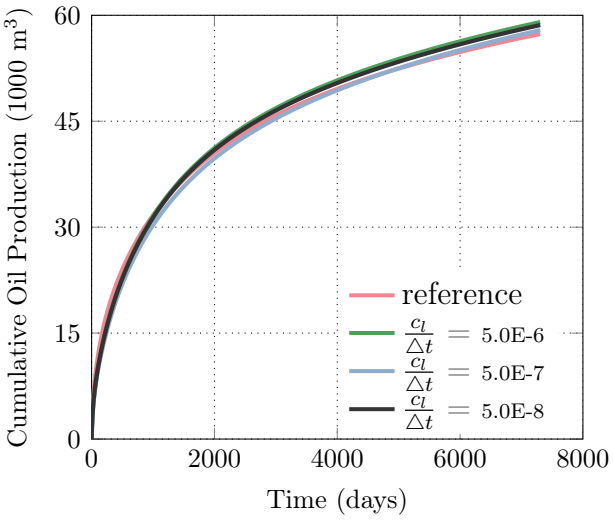


Figure 5.17: Cumulative oil production results of coarse models after using time-stepping algorithm.

Table 5.12: Summary of time steps and total iterations of CG solver for coarse models using time-stepping algorithm in unsteady state method.

$\frac{c_t}{\triangle t}$	#time steps	#iterations
5.0E-6	4	638
5.0E-7	3	576
5.0E-8	4	649

5.4.2.2 Efficiency of multi-level partitioning method

From Figure 5.18, we can observe that the greedy method could not capture the fidelity of this model. The reason is that natural fractures intersected with hydraulic fractures in this test, which forms complex fracture networks. However, both local and multi-level partitioning methods could match the ground truth solution. Figure 5.19 shows saturation contours of the three methods, which also indicate that greedy method failed in this test, and local and multi-level methods have similar saturation profiles as the fine solution. Table 5.13 summarizes results for this test. The advantage of multi-level partitioning over the local one in this test is that multi-level generates much smaller number of fracture grids, and thus is more efficient than the local method.

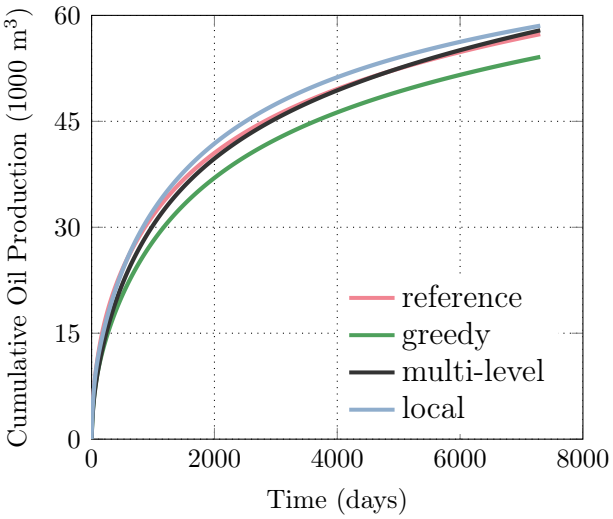
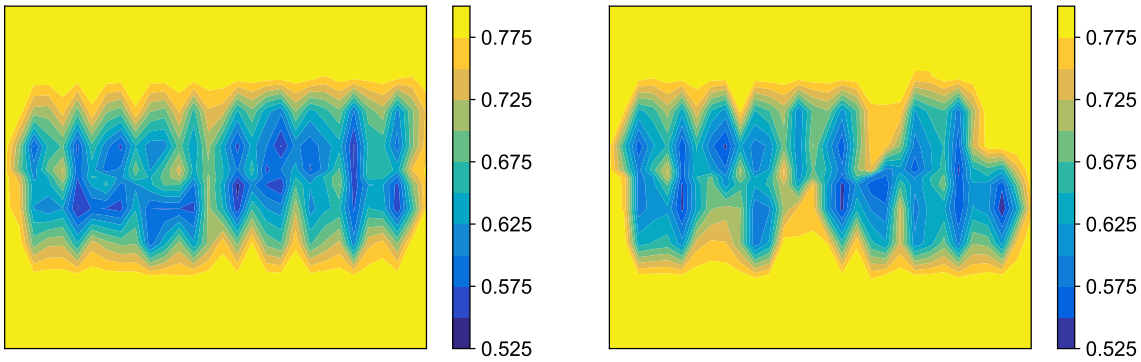


Figure 5.18: Cumulative oil production results of coarse models by using different partitioning methods.

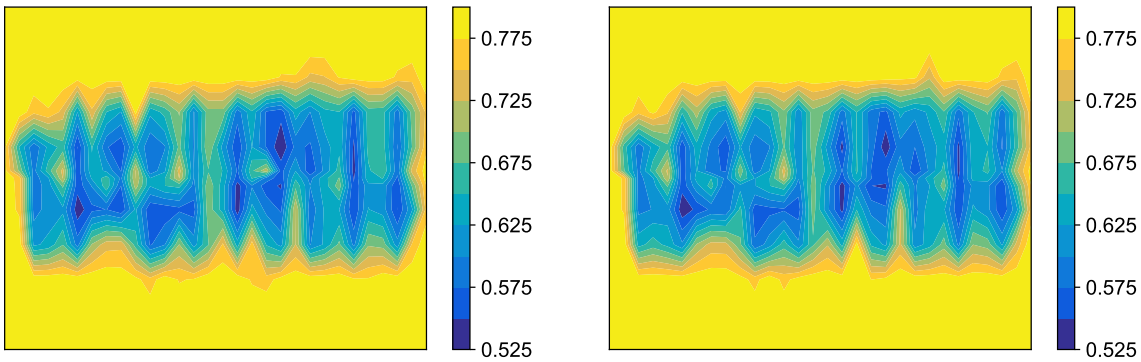
Table 5.13: Summary of simulation results of coarse models by using different partitioning methods in Test 4 (t represents 5 year).

Method	$\epsilon^t_{S_o,\ell_2}$	#m	#f	#total	time	speedup	%grids
fine	-	49830	23935	73765	2099.11	-	-
greedy	5.3E-2	390	68	755	2.78	755	1.0
multi-level	2.7E-2	390	419	809	2.79	752	1.0
local	3.0E-2	390	296	4061	36.91	57	5.5



(a) Averaged fine result

(b) Coarse: greedy partitioning



(c) Coarse: multi-level partitioning

(d) Coarse: local partitioning

Figure 5.19: Saturation profile of the reference solution (a) and saturation errors (b)-(d) of coarse models generated from different partitioning methods after 5 years of production.

Chapter 6

Summary, Conclusions and Recommendations for Future Work

In this chapter, the summary and conclusions are presented followed by recommendations for the future work.

6.1 Summary and conclusions

We present the summary of this work as follows:

- An upscaling framework was proposed to speedup the flow simulations in fractured systems by constructing reduced models to approximate the original fine-scale models. Applicability of this framework was verified by numerical tests presented in this thesis.

- An unsteady state method was introduced in this work to obtain reasonable pressure solutions to generate more accurate coarse models for reservoirs without source term. Applicability of this method for reservoir models without source term was demonstrated by different numerical tests. Furthermore, the unsteady state method showed higher convergence rate than the steady state method for systems with source

terms since matrices formed by unsteady state problems are more well-conditioned than the matrices formed by steady state problems in fractured systems.

- Different partitioning methods were proposed to associate coarse model grids with fine-scale girds. The greedy partitioning method is suitable for systems with numerous short and isolated fractures. While the greedy method might fail in systems with long, well-connected complicated fracture networks, where the local partitioning method generates relatively accurate results.

- Based on the greedy and local partitioning methods that are suitable for different reservoir types, we proposed a more general approach called multi-level partitioning method. The applicability and advantage of this method is presented in the last two numerical tests, where the multi-level approach could generate relatively accurate models with limited fracture grids.

- Implementation issues were discussed for the unsteady state method. We proposed an efficient time-stepping algorithm to adjust the time steps, control the iteration number, and terminate iterations. Efficiency of this algorithm was verified from different perspectives of a hydraulically fractured reservoir model.

6.2 Recommendations for future work

Some recommendations for the future work is presented as follows:

- Verify the upscaling methodology in real reservoir models using field cases to extend the applicability of this method.
- Integrate this upscaling approach with methods used for inverse problems and optimization problems to reduce CPU time for solving such problems where many forward simulations are required.
- Extend and develop upscaling method for the three-dimensional reservoir models.

Bibliography

[1] M. Acene, S. B. Bosma, M. S. A. Kobaisi, and H. Hajibeygi, *Projection-based embedded discrete fracture model (pedfm)*, Advances in Water Resources, 105 (2017), pp. 205 – 216.

[2] M. Acene, M. S. A. Kobaisi, and H. Hajibeygi, *Algebraic multiscale method for flow in heterogeneous porous media with embedded discrete fractures (f-ams)*, Journal of Computational Physics, 321 (2016), pp. 819 – 845.

[3] M. Ahmed Elfeel and S. Geiger, *Static and dynamic assessment of dfn permeability upscaling*, SPE 154369, (2012).

[4] S. Z. Alessio Fumagalli and L. Formaggia, *Advances in computation of local problems for a flow-based upscaling in fractured reservoirs*, Mathematics and Computers in Simulation, 137 (2017), pp. 299 – 324.

[5] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang, *PETSc Web page*. `http://www.mcs.anl.gov/petsc`, 2018.

[6] G. Barenblatt, I. Zheltov, and I. Kochina, *Basic concepts in the theory of seepage of homogeneous liquids in fissured rocks [strata]*, Journal of Applied Mathematics and Mechanics, 24 (1960), pp. 1286 – 1303.

[7] J.-F. Barthelemy, *Effective permeability of media with a dense network of long and micro fractures*, Transport in Porous Media, 76 (2009), pp. 153–178.

[8] T. E. Bill Dershowitz, Paul LaPointe and L. Wei, *Integration of discrete feature network methods with conventional simulator approaches*, SPE Reservoir Evaluation Engineering, 3 (2000), pp. 165–170.

[9] D. Blessent, R. Therrien, and C. W. Gable, *Large-scale numerical simulation of groundwater flow and solute transport in discretely-fractured crystalline bedrock*, Advances in Water Resources, 34 (2011), pp. 1539 – 1552.

[10] Ø. Bøe, *Analysis of an upscaling method based on conservation of dissipation*, Transport in Porous Media, 17 (1994), pp. 77–86.

[11] I. I. Bogdanov, V. V. Mourzenko, J. Thovert, and P. M. Adler, *Effective permeability of fractured porous media in steady state flow*, Water Resources Research, 39.

[12] S. Bosma, H. Hajibeygi, M. Tene, and H. A. Tchelepi, *Multiscale finite volume method for discrete fracture modeling on unstructured grids (ms-dfm)*, Journal of Computational Physics, 351 (2017), pp. 145 – 164.

[13] M. A. Cavalcante Filho J.S., Shakiba M. and S. K., *Implementation of a preprocessor for embedded discrete fracture modeling in an impec compositional reservoir simulator*, SPE Reservoir Simulation Symposium, 23-25 February, Houston, Texas, USA.

[14] Y. Chen and L. J. Durlofsky, *Adaptive local–global upscaling for general flow scenarios in heterogeneous formations*, Transport in Porous Media, 62 (2006), pp. 157–185.

[15] Z. Chen, *Formulations and numerical methods of the black oil model in porous media*, SIAM Journal on Numerical Analysis, 38 (2000), pp. 489–514.

[16] N. Cottereau, M. H. Garcia, O. R. Gosselin, and L. Vigier, *Effective fracture network permeability: Comparative study of calculation methods*, SPE Journal, (2010).

[17] B. Decroux and O. Gosselin, *Computation of effective dynamic properties of naturally fractured reservoirs: Comparison and validation of methods*, SPE 164846, (2013).

[18] M. Delorme, B. Bourbiaux, V. Allken, and B.atfeh, *Upscaling improvement for heterogeneous fractured reservoir using a geostatistical connectivity index*, (2008).

[19] Y. Ding, *Scaling-up in the vicinity of wells in heterogeneous field*, paper presented in SPE Reservoir Simulation Symposium, 12-15 February in

1995, San Antonio, Texas.

[20] S. Du, B. Liang, and L. Yuanbo, *Field study: Embedded discrete fracture modeling with artificial intelligence in permian basin for shale formation*, SPE Annual Technical Conference and Exhibition, 9-11 October, San Antonio, Texas, USA.

[21] A. Fumagalli, L. Pasquale, S. Zonca, and S. Micheletti, *An upscaling procedure for fractured reservoirs with embedded grids*, Water Resources Research, 52 (2016), pp. 6506–6525.

[22] Q. Gan and D. Elsworth, *Production optimization in fractured geothermal reservoirs by coupled discrete fracture network modeling*, Geothermics, 62 (2016), pp. 131 – 142.

[23] P. K. Ghahfarokhi, *The structured gridding implications for upscaling model discrete fracture networks (dfn) using corrected oda's method*, Journal of Petroleum Science and Engineering, 153 (2017), pp. 70 – 80.

[24] B. Gong, *Effective Models of Fractured Systems*, PhD thesis, Stanford, 2007.

[25] B. Gong, M. Karimi-Fard, and L. J. Durlofsky, *Upscaling discrete fracture characterizations to dual-porosity, dual-permeability models for efficient simulation of flow with strong gravitational effects*, SPE Journal, 13 (2008), pp. 58–67.

[26] H. Hajibeygi, D. Karvounis, and P. Jenny, *A hierarchical fracture model for the iterative multiscale finite volume method*, Journal of Computational Physics, 230 (2011), pp. 8729 – 8743.

[27] M.-H. Hui, B. T. Mallison, M. H. Fyrozjaee, and W. Narr, *The upscaling of discrete fracture models for faster, coarse-scale simulations of ior and eor processes for fractured reservoirs*, SPE Annual Technical Conference and Exhibition, 30 September-2 October, New Orleans, Louisiana, USA.

[28] M.-H. R. Hui, M. Karimi-Fard, B. Mallison, and L. J. Durlofsky, *A general modeling framework for simulating complex recovery processes in fractured reservoirs at different resolutions*, SPE Reservoir Simulation Conference, 20-22 February, Montgomery, Texas, USA.

[29] B. Kai, S. Amgad, and S. Shuyu, *Upscaling of permeability field of fractured rock system: Numerical examples*, Journal of Applied Mathematics, 2012 (2012).

[30] G. Karay and G. Hajnal, *Modelling of groundwater flow in fractured rocks*, Procedia Environmental Sciences, 25 (2015), pp. 142 – 149. 7th Groundwater Symposium of the International Association for Hydro-Environment Engineering and Research (IAHR).

[31] M. Karimi-Fard and L. Durlofsky, *A general gridding, discretization, and coarsening methodology for modeling flow in porous formations*

with discrete geological features, Advances in Water Resources, 96 (2016), pp. 354 – 372.

[32] M. Karimi-Fard, L. Durlofsky, and K. Aziz, *An efficient discrete-fracture model applicable for general-purpose reservoir simulators*, SPE Journal, 9 (2004), pp. 227–236.

[33] M. Karimi-Fard and L. J. Durlofsky, *Accurate resolution of near-well effects in upscaled models using flow-based unstructured local grid refinement*, SPE Journal, 17 (2012), pp. 1084–1095.

[34] M. Karimi-Fard, B. Gong, and L. J. Durlofsky, *Generation of coarse scale continuum flow models from detailed fracture characterizations*, Water Resources Research, 42.

[35] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM Journal on Scientific Computing, 20 (1998), pp. 359–34.

[36] H. Kazemi, L. Merrill, K. Porterfield, and P. Zeman, *Numerical simulation of water-oil flow in naturally fractured reservoirs*, SPE Journal, 16 (1976), pp. 317–326.

[37] C. T. O. Leung, A. R. Hoch, and R. W. Zimmerman, *Comparison of discrete fracture network and equivalent continuum simulations of fluid flow through two-dimensional fracture networks for the decovalexÃćÂĂÂŞ2011 project*, Mineralogical Magazine, 76 (2012), p. 3179.

[38] L. Li and S. H. Lee, *Efficient field-scale simulation of black oil in a naturally fractured reservoir through discrete fracture networks and homogenized media*, SPE Reservoir Evaluation and Engineering, 11 (2008), pp. 750–758.

[39] K. Lim and K. Aziz, *Matrix-fracture transfer shape factors for dual-porosity simulators*, Journal of Petroleum Science and Engineering, 13 (1995), pp. 169 – 178.

[40] M. Lough, S. Lee, and J. Kamath, *An efficient boundary integral formulation for flow through fractured porous media*, Journal of Computational Physics, 143 (1998), pp. 462 – 483.

[41] F. G. Michel Garcia and O. R. Gosselin, *Fast and efficient modeling and conditioning of naturally fractured reservoir models using static and dynamic data*, SPE 107525, (2007).

[42] A. Moinfar, A. Varavei, K. Sepehrnoori, and R. T. Johns, *Development of an efficient embedded discrete fracture model for 3d compositional reservoir simulation in fractured reservoirs*, SPE Journal, 19 (2014), pp. 289–303.

[43] T. Nakashima and L. J. Durlofsky, *Accurate representation of near-well effects in coarse-scale models of primary oil production*, Transport in Porous Media, 83 (2010), pp. 741–770.

[44] J. R. Natvig, B. Skaflestad, F. Bratvedt, K. Bratvedt, K.-A. Lie, V. Laptev, and S. Khataniar, *Multiscale mimetic solvers for efficient streamline simulation of fractured reservoirs*, SPE Journal, 16 (2011), pp. 880–888.

[45] M. Oda, *An equivalent continuum model for coupled stress and fluid flow analysis in jointed rock masses*, Water Resources Research, 22 (1986), pp. 1845–1856.

[46] S. Pozdniakov and C. Tsang, *A selfÃćÂĂÂŘconsistent approach for calculating the effective hydraulic conductivity of a binary, heterogeneous medium*, Water Resources Research, 40.

[47] K. Pruess, *A practical method for modeling fluid and heat flow in fractured porous media*, SPE Journal, 25 (1985), pp. 14–26.

[48] R. Rossen and E. Shen, *Simulation of gas/oil drainage and water/oil imbibition in naturally fractured reservoirs*, SPE Reservoir Engineering, 4 (1989), pp. 464–470.

[49] H. Salimi and H. Bruining, *Upscaling in vertically fractured oil reservoirs using homogenization*, Transport in Porous Media, 84 (2010), pp. 21–53.

[50] S. Shah, O. Mazyner, M. Tene, K.-A. Lie, and H. Hajibeygi, *The multiscale restriction smoothed basis method for fractured porous media (f-msrsb)*, Journal of Computational Physics, 318 (2016), pp. 36 – 57.

[51] M. Soleimani, *Naturally fractured hydrocarbon reservoir simulation by elastic fracture modeling*, Petroleum Science, 14 (2017), pp. 286–301.

[52] H. S. Vik, S. Salimzadeh, and H. M. Nick, *Heat recovery from multiple-fracture enhanced geothermal systems: The effect of thermoelastic fracture interactions*, Renewable Energy, 121 (2018), pp. 606 – 622.

[53] M. C. Vincent, *Restimulation of unconventional reservoirs: When are refracs beneficial?*, SPE conference, (2010).

[54] J. Warren and P. Root, *The behavior of naturally fractured reservoirs*, SPE Journal, 3 (1963), pp. 246–255.

[55] Y. Xu, J. S. A. C. Filho, W. Yu, and K. Sepehrnoori, *Discrete-fracture modeling of complex hydraulic-fracture geometries in reservoir simulators*, SPE Reservoir Evaluation and Engineering, 20 (2017), pp. 403–422.

[56] Y. Xu, W. Yu, and K. Sepehrnoori, *Modeling dynamic behaviors of complex fractures in conventional reservoir simulators*, SPE/AAPG/SEG Unconventional Resources Technology Conference, 24-26 July, Austin, Texas, USA.