Copyright
by
Omar A. Al-Hinai
2014

The Dissertation Committee for Omar A. Al-Hinai
certifies that this is the approved version of the following dissertation:

Mimetic Finite Differences for Porous Media Applications

Committee:

Mary Wheeler, Supervisor

Todd Arbogast

Mojdeh Delshad

Leszek Demkowicz

Robert van de Geijn

Ivan Yotov

Mimetic Finite Differences for Porous Media Applications

by

Omar A. Al-Hinai, B.A., M.A.

DISSERTATION
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

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN
May 2014

For Emily

Acknowledgments

I am grateful for the support and help of many friends, family and colleagues. I want to thank my parents for all they have done. My brothers Zaid and Muhannad, and my sister Bushra. My girlfriend, Emily Rowlinson, who very patiently supported me throughout writing my dissertation.

So many wonderful people at ICES for their support and friendship, especially Thomas Kirschenmann, Andrea Hawkins, Jack Poulson, Talea Lashea, John Evans, Corey Bryant, Kathryn Farrell, Avi Taicher, Pearl Flath, Rebecca Morrison, John Hawkins, Vikram Garg, Rhys Ulerich, Nick Alger, Jeff Hussmann, Henry Chang, Chris Mirabito, Paul Tsuji, Jesse Windle, Jeff Zitelli, Prapti Neupane. Many thanks to our graduate coordinator, Stephanie Rodriguez. My colleagues at the Center for Subsurface Modeling, Gergina Pancheva, Guangri Xue, Sunil Thomas, Tameem Almani, Deandra White, and Connie Baxter.

My friends from Boston University, Matthew Allen, Gabriel Redner, Drew McGhie. My undergraduate and masters advisor, Margrit Betke.

I'm very thankful for so many of my teachers, especially Kristin Walton, who introduced me to both Calculus and whiffle ball. My highschool math teacher, Noor Al-Deen Salem.

From Saudi Aramco, my boss and mentor Ali Dogru, who got me

started in reservoir simulation. Also colleagues Usuf Middya and Larry Fung and Nabeel Al Zamil, who introduced me to Python. I especially want to thank Amin Al-Nasser, I will never forget what you have done for me.

My committee members, Todd Arbogast, Mojdeh Delshad, Leszek Demkowicz, Robert van de Geijn, and Ivan Yotov. I learned so much both during the proposal and the defense from your input. Thank you.

And finally, my teacher and mentor, my doctoral supervisor Mary Wheeler. You have guided me both in mathematics and in life. It has been an honor to be your student.

Mimetic Finite Differences for Porous Media Applications

Omar A. Al-Hinai, Ph.D.
The University of Texas at Austin, 2014

Supervisor: Mary Wheeler

We connect the Mimetic Finite Difference method (MFD) with the finite-volume two-point flux scheme (TPFA) for Voronoi meshes. The main effect is reducing the saddle-point system to a much smaller symmetric-positive definite matrix. In addition, the generalization allows MFD to seamlessly integrate with existing porous media modeling technology. The generalization also imparts the monotonicty property of the TPFA method on MFD. The connection is achieved by altering the consistency condition of the velocity bilinear operator. First-order convergence theory is presented as well as numerical results that support the claims.

We demonstrate a methodology for using MFD in modeling fluid flow in fractures coupled with a reservoir. The method can be used for nonplanar fractures. We use the method to demonstrate the effects of fracture curvature on single-phase and multi-phase flows. Standard benchmarks are used to demonstrate the accuracy of the method. The approach is coupled with existing reservoir simulation technology.

Table of Contents

Acknowledgments **v**

Abstract **vii**

List of Tables **xi**

List of Figures **xii**

Nomenclature **xv**

Chapter 1. Introduction **1**

1.1 Overview 1

1.2 Historical Development of MFD 5

Chapter 2. Generalized Mimetic Finite Differences **8**

2.1 Basic Definitions 8

2.2 Stability Analysis 13

2.3 Velocity Convergence 17

2.4 Pressure Convergence 20

2.5 Weighting Functions 22

2.6 Constructing the Linear System 24

Chapter 3. MFD and Two-Point Fluxes **30**

3.1 Finite Volume Methods (FVM) and MFD 30

3.1.1 Previous Work 33

3.1.2 General Voronoi Diagrams 35

3.2 Cell-Centered Schemes 38

3.3 Point-Centered Schemes 42

Chapter 4. Fracture Modeling **45**
4.1 Previous Work 46
4.2 Problem Formulation 47
4.2.1 Basic Framework 47
4.2.2 Single-Phase Slightly Compressible Flow 49
4.2.3 Two-Phase Slightly Compressible Flow 51
4.3 Discretization 52
4.3.1 Single-Phase Slightly Compressible Flow 53
4.3.2 Two-Phase Slightly Compressible Flow 54

Chapter 5. MFD Code in Python **56**
5.1 Boundaries 58
5.2 Associated Codes 59

Chapter 6. Numerical Results **60**
6.1 Two-Dimensional Convergence 61
6.1.1 Rectangular Grids 62
6.1.2 Voronoi Grids 63
6.2 Three-Dimensional Convergence 65
6.2.1 Rectangular Grids 68
6.2.2 Voronoi Grids 71
6.3 Two-phase Problems 73
6.3.1 Low Permeability Barrier 73
6.3.2 SPE10 76
6.4 Fracture Problems 81
6.4.1 Cinco-Ley Benchmark 81
6.4.2 Single-Phase Slightly Compressible Flow with IPARS . . 85
6.4.3 Fracture Curvature 86
6.4.4 Two-Phase Flow with Planar and Curved Fractures . . 89

Chapter 7. Conclusions and Future Work **93**
7.1 Future Work 94

Appendices 97

Appendix A. Surface and Nodal Approaches 98
A.1 Cell-Node Discretization 98
A.2 Cell-Surface Discretization 101

Appendix B. Two-Phase Flow using MFD 104

Bibliography 108

Vita 118

List of Tables

6.1 Convergence errors and rates for the four test cases with rectangular grids over two-dimensional domains. 64

6.2 Convergence errors and rates for the four test cases with random Voronoi grids. Notice how second-order convergence of pressure always occurs at the shifted point x_E. 67

6.3 Convergence errors and rates for the four test cases with rectangular grids in three-dimensions. We consistently find at least first-order convergence for both pressure and velocity. Also notice that the pressure solution is most accurate when the method and error is calculated using the x_E point. 70

6.4 Convergence errors and rates for the four test cases with random Voronoi meshes. We consistently find at least first-order convergence for both pressure and velocity. Also notice that the pressure solution is most accurate when the error is calculated at the x_E point. 72

6.5 The five fractures are shaped as quadratic polynomials starting at p_1, ending at p_3 and pass through p_2. 87

List of Figures

1.1 Examples of geometrically complex reservoir features. 3

2.1 The points needed in the construction of weighting function w_E. The weighting function is the sum of two piece-wise constants: one is defined over the entire cell, and the other is defined over the ball centered at A. The result is a shifting of the center of gravity from C to A. 25

2.2 Example of a cell with boundary points shifted to the top right and bottom left corners. In this example, matrix $R_E^T N_E$ is singular. 29

3.1 The vectors r are constructed by connecting x_E with the appropriate point x_e. These vectors make up the rows of matrix R_E, $(|e_1|r_1^T...|e_4|r_4^T)$. The rows of matrix N_E are the normal vectors multiplied by $\mathbf{K}_E$, $((\mathbf{K}_E n_1)^T...(\mathbf{K}_E n_4)^T)$. The original MFD sets the point x_E to the centroid of the cells and the points x_e to the the centroids of the faces. Our generalization allows the point x_E to be shifted on the interior of the cell, and the points x_e to be shifted on the plane of the faces. By shifting these points, we can establish collinearity vectors r and n, resulting in a diagonal matrix M_E. 32

3.2 A two-dimensional Voronoi diagram (solid line) and its geometric dual, the Delaunay triangulation (dashed line). The dashed lines are always orthogonal to the corresponding solid line. The red circle indicates how the intersection of the two lines may occur outside of the face boundaries. 38

4.1 Define two separate domains with two separate models, one for the flow in the reservoir and one for flow in the fracture. . . . 49

6.1 The four cases 62

6.2 The rectangular meshes used in the convergence study. 63

6.3 The MFD pressure solution (left) and the analytic pressure solution (right) for using rectangular meshes. 65

6.4 The random Voronoi meshes used in the convergence study. . 66

6.5 The MFD pressure solution (left) and the analytic pressure solution computed at the cell centroids (right) using two-dimensional Voronoi diagrams. 66

6.6 The rectangular meshes used in the convergence study. The two meshes correspond to h = $\frac{1}{16}$, and $\frac{1}{32}$. 69

6.7 Comparison between the MFD solution (left) and the exact solution computed at the cell centroids (right) over a three dimensional domain with full permeability tensor. 69

6.8 Unstructured three dimensional Voronoi diagrams. The plot on the left is the full domain of the convergence test problem in which we can only see the outer faces of the boundary. The plot on the right is a slice of the mesh showing the inner structure of the diagram. 71

6.9 Pressure solution for three dimensional Voronoi diagram. The plot on the left represents the MFD approximation to pressure, and the plot on the right is the analytical solution computed at the cell centroids. 73

6.10 In the low permeability barrier example, three different orientations of the barrier are generated and tested. The blue streak corresponds to the area in the reservoir with low permeability. The geometry of the barrier is captured using a Voronoi mesh. 74

6.11 The Voronoi mesh can capture the shape of the barrier while maintaining a rectangular mesh in the rest of the domain. . . 75

6.12 Pressure solution to the low-permeability barrier problem with different barrier orientations. 76

6.13 The saturation solution at different times for different barrier orientations. 77

6.14 The log permeability of the 85th layer of SPE 10. The bottom layers of the SPE 10 problem are characterized by channelized permeabilities. A rate specified injector is placed at the bottom-right corner, and a pressure specified producer is placed in the top-left corner of the domain. 79

6.15 SPE 10 saturation profile after 579, 1158, 1737 and 2135 days. Notice the highly channelized nature of the saturation in the reservoir. 80

6.16 Problem setup for Cinco-Ley benchmark. The entire domain is 65×85m, with a fracture of length 10m in the center. The well is located at the left-most point of the fracture. 83

6.17 Pressure profiles with different fracture conductivities C_{fD}. Higher C_{fD} values correspond to a higher conductivity of the fracture. As the conductivity decreases, the pressure solution approaches a radial profile. 83

6.18 The leakage profile for the fracture problem with conductivity factors ($C_{fD} = [100, 5, .2]$) at different times. The blue line is the semi-analytic solution by Cinco-Ley and Meng, and the red line is the MFD solution. As the conductivity factor decreases, the leakage increases at the well, which is located at $x = 0$. The MFD fracture method is able to accurately capture the leakage for these different cases. 84

6.19 Reservoir description for MFD and IPARS coupling example. . 85

6.20 Reservoir cross-section for the MFD and IPARS coupling example. 86

6.21 The problem setup for testing the effects of curvature on fracture flow. We have a square domain with a Dirichlet (pressure) specified boundaries. The fracture is placed in the middle with rate specified well injecting fluid directly into the fracture. . . 88

6.22 Three of the meshes used to test the different degrees of curvature. In the first mesh we have a planar fracture. The curvature is increased by applying a quadratic perturbation to the mesh. 88

6.23 Effects of curvature on the leakage around a fracture. The blue line is the leakage from the top of the fracture, and the red line is the leakage from the bottom. As the fracture curvature is increased, we notice an increase in leakage from the top compared to the bottom. 89

6.24 Outline of the two fractures used in the two-phase comparison. 91

6.25 Pressure profile for the two-phase flow without curvature (left) and with curvature (right). 91

6.26 A close look at the reservoir mesh for the curved fracture case. 92

6.27 The saturation profile near the fracture at early time (0.23 days). Note the higher saturation on the top of the fracture compared to the bottom in the case of a curved fracture surface. 92

A.1 Unknowns $U_{i,j}$ and $W_{i,j}$ 99

Nomenclature

X_h = Discrete velocity space
Q_h = Dicrete pressure space
N_X = Number of faces in mesh
N_Q = Number of cells in mesh
k_E = Number of faces for cell E
E = Cell E in mesh
$|E|$ = Volume of cell E
e = Face e in mesh
$|e|$ = Area of face e
$\mathbf{n}^e$ = Unit normal on face e.
$\mathbf{n}^e_E$ = Unit normal on face e pointing out of element E.
ρ = Density
Q = Source or sink term (producer or injector)
K = Continuous Permeability tensor
$\mathbf{K}$ = Piece-wise constant Permeability tensor
$k_{r\alpha}$ = Relative permeability of phase α
p = Pressure
p_c = Capillary pressure
s_α = Saturation of phase α
$s_{r\alpha}$ = Residual saturation of phase α
μ = Fluid viscosity
t = Time
w = Fracture width
X_f = Fracture length

Chapter 1

Introduction

1.1 Overview

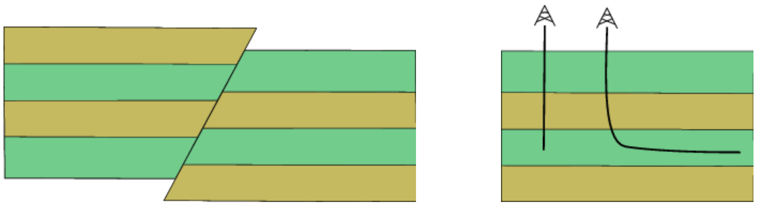
Early subsurface flow models often assumed a simple rectangular domain with a structured, rectangular mesh. With advances in reservoir characterization methods, there has been an increasing demand for models that honor complex geometric features. Further model complexity arises from new drilling technology. Wells have gone from perfectly straight and vertical to horizontal and sometimes branching structures [Fig. 1.1]. These effects are compounded when considering hydraulic fractures. Properly capturing the geometry of such problems while solving multiphase flow is a challenging task. In this work, we demonstrate how the Mimetic Finite Difference method is ideally suited for modeling flow with complex subsurface geometries.

The Mimetic Finite Difference (MFD) method has found widespread application in many areas of numerical modeling. The basic premise of MFD has been around for many years. The central theme is to create discretizations that "mimic" the fundamental properties of the underlying partial differential equation. The MFD method is a constructive approach to discretization, MFD discretizations defined in the literature follow the same basic steps. MFD has been used to solve elliptic and parabolic equations and has found applications in the areas of gas dynamics [54], Maxwell's equations [13, 31] and Stokes flow [22]. It is also possible to apply the Mimetic Finite Difference method to cell-faced [15] and nodal discretization [14]. MFD can be used both to define higher-order methods [57], and on meshes with curved faces [39]. More recent work in higher order methods related to MFD has been in the area of the Virtual Element Method [21].

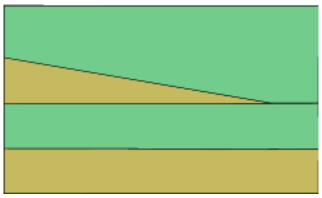
Early examples of MFD were defined with certain mesh restrictions, *e.g.*, logically rectangular grids [30, 54]. In 2005, Brezzi *et al.* presented a definition of MFD that applied to a general set of polygonal and polyhedral meshes [15, 16]. This opened a door to many new applications of MFD in the modeling community. Properties that make MFD an attractive option for porous media problems include the following:

1. Local conservation by construction;
2. Approximation of both pressure and velocity unknowns;
3. Handling of full tensor coefficients;
4. Accommodation of hanging nodes and non-matching grids.

While these properties are powerful, there remain challenges to overcome. First, MFD results in a saddle-point linear system. Such systems are more difficult and time-consuming to solve than positive definite problems. Second, MFD generates a large system of equations by solving simultaneously for both



(a) Fault (b) Horizontal Well



(c) Pinch-Out

Figure 1.1: Examples of geometrically complex reservoir features.

pressure and velocity unknowns. Finally, the MFD method will not, in general, produce a solution that satisfies the maximum principle of elliptic PDE.

These challenges are shared by the related mixed finite element (MFE) method. In 1983, Wheeler and Russell proposed a popular approach to addressing these challenges [50]. They demonstrated a connection between cell-centered finite volumes and mixed finite elements on rectangular grids. This connection was accomplished by defining a special quadrature rule. The resulting linear system was sparse, symmetric positive definite, and it had only pressures as unknowns. Their technique led to an elegant proof of convergence for finite volumes over variably spaced grids based on the theory of MFE.

Our approach follows in the same spirit as the approach developed by Wheeler and Russell. However, instead of using RT_0 as the underlying discretization, we will use MFD. For orthogonal, rectangular grids, the relation between MFD and cell-centered finite volumes has already been established. In fact, over simplicial grids, RT_0 spaces are a special case of MFD [41]. We present a generalization of MFD that reduces to a TPFA approximation for a more general set of meshes. The new method is based on combining and modifying two generalizations existing in the literature [24,38]. The new form of MFD is defined in Chapter 2 with proof of first-order convergence. In Chapter 3, we examine the implications of the modification for general Voronoi diagrams, 2.5-dimensional Voronoi diagrams. In addition, we demonstrate the relationship to point-centered schemes.

The second part of this work is to apply the MFD method to solving flow problems with fractures. We use MFD to create a coupled system of flow in a fracture with flow in the reservoir. Due to the geometric flexibility of MFD, we are able to demonstrate the effects of fracture curvature on flow. In addition, we use the MFD method implemented in Python to couple with an existing reservoir simulator, IPARS. Details of this work are provided in Chapter 4.

In Chapter 5 we present an overview of the code used to run the numerical results of this work. Our code leverages both the general nature of MFD and its simplicity to create an extensible research platform. We demonstrate the use of MFD for single-phase, two-phase, and fracture flow problems.

1.2 Historical Development of MFD

The Mimetic Finite Difference method finds its origins in work by Favorskii, Samarskii, Tishkin and Shashkov [26,53], with earlier papers published in Russian. The first translations of their work referred to the MFD method as the support-operator method (SOM). The term "support-operator" is a translation of the original Russian name, "опорных операторов," phonetically pronounced "opornykh operatorov" [54]. The term means primary-operator or main-operator, and it refers to the constructive way in which the method is formulated.

The MFD method was originally motivated by a desire to define conservative finite difference methods over unstructured grids. The idea was to construct a discrete form of vector calculus. To this end, first-order discrete differential operators are defined to bear the same relationships to each other as their continuous counterparts. Let us take Laplace's equation as an example,

$$-\nabla \cdot \nabla u = 0. \tag{1.1}$$

The two operators needed to create a Laplacian are the divergence ($\nabla \cdot$) and the gradient (∇). These two operators share a connection through the integration by parts formula,

$$\int_{\Omega} \nabla p \cdot v \ \ dV = \int_{\partial\Omega} (pv) \cdot \mathbf{n} \ \ dS - \int_{\Omega} p \, \nabla \cdot v \ \ dV. \tag{1.2}$$

One may state that the ∇ and the $\nabla\cdot$ operators are formal adjoints relative to the $L^2(\Omega)$ and $(L^2(\Omega))^d$ inner products,

$$(\nabla p, v)_{(L^2(\Omega))^d} = -(p, \nabla \cdot v)_{L^2(\Omega)}, \tag{1.3}$$

and, therefore,

$$\nabla\cdot = (\nabla)^*. \tag{1.4}$$

Implicit in this relationship is the divergence theorem, which is the conservative property for the diffusion problem. By defining appropriate discrete inner products that correspond to the continuum case, it is possible to create discrete ∇ and $\nabla\cdot$ operators, denoted by $\mathcal{G}$ and $\mathcal{DIV}$, that share the adjoint relationship,

$$[\mathcal{G}\, p, v]_{vector} = -[p, \mathcal{DIV}\, v]_{scalar}. \tag{1.5}$$

With this objective in mind, the general steps for defining a mimetic discretization for Laplace's problem are as follows:

1. Define discrete scalar and vector spaces, which we denote as Q_h and X_h, respectively.

2. Define discrete scalar and vector inner products $[\cdot,\cdot]_{Q_h}$ and $[\cdot,\cdot]_{X_h}$.

3. Define one of the differential operators in the equation. This choice is typically dependent on the discrete vector and scalar spaces chosen. For our purposes we first define $\mathcal{DIV}$. This is the "reference" operator for the method, which is where the original name is sourced.

4. Derive the second operator based on the adjoint relationship. For example, if we have defined $\mathcal{DIV}$, then $\mathcal{G}$ is defined as

$$[\mathcal{G}\, p, v]_{X_h} = [p, \mathcal{DIV}\, v]_{Q_h} \;\; \forall v \in X_h \,\text{and}\, \forall p \in Q_h. \tag{1.6}$$

All MFD discretizations defined in the literature follow this basic model. In the case of cell-centered schemes, the natural primary operator is $\mathcal{DIV}$, and, for nodal discretizations, $\mathcal{G}$ is the primary operator. Typically, one of the inner products is defined with ease, while the other requires special care. In Appendix A, we present a brief overview of the the cell-node and cell-surface methods. These are presented for historical reference.

Chapter 2

Generalized Mimetic Finite Differences

We introduce a generalized form of the original MFD defined in [15]. In the following sections, we establish stability of the method, as well as first-order convergence for the velocity and pressure unknowns. Our methodology in developing the theoretical work is to use the same approach followed by the authors of [38], making modifications where needed due to the modified consistency assumption suggested.

2.1 Basic Definitions

Consider the homogeneous Dirichlet problem,

$$\begin{aligned} v &= -K\nabla p && \text{in } \Omega, \\ \nabla\cdot v &= f && \text{in } \Omega, \\ p &= 0 && \text{on } \partial\Omega. \end{aligned} \tag{2.1}$$

The domain $\Omega \subset \mathbb{R}^d$ is divided into a nonoverlapping, conformal partition $\mathcal{T}$ of elements E. In three dimensions, E is a polyhedron with planar polygonal faces. In two dimensions, E is polygonal with line segments for faces. For the remainder of this work, we refer to elements E as polyhedra, with the understanding that the results also apply directly to the two-dimensional case.

Let the permeability tensor K be elliptic, and every component of K is in $W^{1,\infty}(\Omega)$. We make the following assumptions about the mesh:

(1) Ω is a polyhedron with a Lipschitz boundary.

(2) There exists an upper bound on the number of faces and edges in the mesh.

(3) There exist constants v_* and a_* such that for all E

$$v_* h_E^d \leq |E|,$$
$$a_* h_E^{d-1} \leq |e|.$$

(4) The mesh faces are planar and star-shaped.

(5) The mesh cells are non-degenerate and shape regular.

Define the space of discrete pressures and velocities as $Q_h \subset \mathbb{R}^{N_Q}$ and $X_h \subset \mathbb{R}^{N_X}$, respectively, with

$$p_h \in Q_h \text{ and } p_h = \{p_E\}_{E \in \mathcal{T}_h} \text{ such that } p_E \in \mathbb{R}, \tag{2.2}$$

$$v_h \in X_h \text{ and } v_h = \{v_E^e\}_{E \in \mathcal{T}}^{e \in \partial E} \text{ such that } v_E \in \mathbb{R}. \tag{2.3}$$

Each velocity degree of freedom v_E^e represents the flux normal to the face e facing out of cell E. The above definition for X_h results in two degrees of freedom per internal face in the mesh and a single degree of freedom for each boundary face. We require continuity of velocity in X_h, that is, for two adjacent elements E_a and E_b sharing a face e, let

$$v^e_{E_a} + v^e_{E_b} = 0.$$

Continuity of velocity reduces the total number of degrees of freedom in X_h to N_X.

For $q \in L^1(\Omega)$, define a projection operator on Q_h as

$$(q^I)|_E = \frac{1}{|E|} \int_E q\, dV. \tag{2.4}$$

For $v \in (L^s(\Omega))^d$, $s > 2$ and $\nabla \cdot v \in (L^2(\Omega))^d$, define the projection operator on X_h as

$$(v^I)^e_E = \frac{1}{|e|} \int_e v \cdot \mathbf{n}^e_E\, dS. \tag{2.5}$$

Let $\mathbf{K}$ denote a constant tensor over E such that

$$\max_{ij} \|K_{ij} - \mathbf{K}_{E,ij}\|_{L^\infty(E)} \le Ch_E.$$

Define the scalar inner product as,

$$[q_h, p_h]_{Q_h} = \sum_{E \in \mathcal{T}} p_E q_E |E|.$$

Define the vector bilinear form as,

$$[v_h, u_h]_{X_h} = \sum_{E \in \mathcal{T}} [v_h, u_h]_E = \sum_{E \in \mathcal{T}} u_h^T M_E v_h.$$

This form approximates the continuous inner-product, that is,

$$[v_h, u_h]_{X_h} \approx \int_\Omega K^{-1} v \cdot u \ \ dV.$$

We require the following conditions on the velocity inner product:

(S1) *(Stability)* There exist two positive constants, s_* and S^*, such that, for all $\xi \in \mathbb{R}^{k_E}$ and $E \in \mathcal{T}$,

$$s_*|E|\xi^T\xi \leq \xi^T M_E \xi \leq S^*|E|\xi^T\xi, \tag{2.6}$$

and

$$\xi^T M_E^T M_E \xi \leq (S^*)^2 |E|^2 \xi^T \xi.$$

($\tilde{\text{S}}$2) *(Consistency)* For every element E, for every linear function q^1 on E, and for every $v_h \in X_h$, there exists w_E and w_e such that

$$[(\mathbf{K}_E \nabla q^1)^I, v_h]_E = \sum_{e\in\partial E} v_E^e \int_e w_e q^1 \, dS - \int_E w_E q^1 (\mathcal{DIV}\, v_h)_E \, dV. \tag{2.7}$$

where the function $w_E : \mathbb{R}^d \to \mathbb{R}$ satisfies

$$\int_E w_E \, dx = |E|, \tag{2.8}$$

$$\int_E g w_E \, dx = |E| x_E, \tag{2.9}$$

where x_E is a point in E and $g : \mathbb{R}^d \to \mathbb{R}^d$ is the linear function,

$$g(x, y, z) = \begin{pmatrix} x \\ y \\ z \end{pmatrix},$$

and where the function $w_e : \mathbb{R}^d \to \mathbb{R}$ satisfies,

$$\int_e w_e \, dx = |e|,$$

$$\int_e g w_e \, dx = |e| x_e,$$

where x_e is a point in the plane of e. The functions w_e and w_E must be bounded by positive constants independent of the mesh $w_e^{\max}$ and $w_E^{\max}$,

$$\max_{E\in\mathcal{T}} \|w_E\|_{L^\infty(E)} < w_E^{\max},$$
$$\max_{E\in\mathcal{T}} \max_{e\in\partial E} \|w_e\|_{L^\infty(e)} < w_e^{\max}.$$

We further require that w_E is strictly positive over E, that is,

$$w_E(x) > w_E^{\min} > 0, \quad \forall E \in \mathcal{T} \text{ and } \forall x \in E.$$

The addition of weighting functions in ($\tilde{\text{S}}$2) is only modification of the original (S2) condition in [15]. The addition of weighting functions on the integrals was inspired by the modification suggested in [25] and [38]. Details about the relation to previous work can be found in Chapter 3.

Define the discrete divergence operator $\mathcal{DIV} : X_h \rightarrow Q_h$ as,

$$(\mathcal{DIV}\, v_h)_E = \frac{1}{|E|} \sum_{e\in\partial E} v_E^e |e|.$$

Finally, define a discrete gradient operator $\mathcal{G} : Q_h \rightarrow X_h$ as the adjoint of $\mathcal{DIV}$,

$$[\mathcal{G}\, p_h, v_h]_{X_h} = -[p_h, \mathcal{DIV}\, v_h]_{Q_h}. \tag{2.10}$$

The final problem can be represented in "weak" saddle-point form. That is, find $v_h \in X_h$ and $p_h \in Q_h$ such that,

$$[v_h, u_h]_{X_h} - [p_h, \mathcal{DIV}\, u_h]_{Q_h} = 0, \qquad \forall u_h \in X_h, \tag{2.11}$$
$$[\mathcal{DIV}\, v_h, q_h]_{Q_h} = [f^I, q_h]_{Q_h}, \qquad \forall q_h \in Q_h. \tag{2.12}$$

2.2 Stability Analysis

The analysis follows the classical approach to stability for mixed methods. We start by defining the following norms over the pressure and velocity spaces,

$$\|v_h\|_{X_h}^2 = [v_h, v_h]_{X_h},$$
$$\|p_h\|_{Q_h}^2 = [p_h, p_h]_{Q_h}.$$

We also define a "div" norm

$$\|v_h\|_{\mathrm{div}}^2 = \|v_h\|_{X_h}^2 + \|\mathcal{DIV}\, v_h\|_{Q_h}^2.$$

Let Z_h represent the divergence-free subspace of X_h,

$$Z_h = \{u_h \in X_h \mid \mathcal{DIV}\, u_h = 0\}\,. \tag{2.13}$$

Lemma 1. *The bilinear operators* $[\cdot\,,\cdot]_{X_h}$ *and* $[\cdot\,, \mathcal{DIV}\,\cdot]_{Q_h}$ *are continuous, in the sense that*

$$[v_h, u_h]_{X_h} \leq C\|v_h\|_{X_h}\|u_h\|_{X_h} \;\; \forall v_h, u_h \in X_h, \tag{2.14}$$
$$[p_h, \mathcal{DIV}\, v_h]_{Q_h} \leq C\|p_h\|_{Q_h}\|v_h\|_{div} \;\forall p_h \in Q_h, v_h \in X_h. \tag{2.15}$$

Inequality (2.14) is a consequence of Cauchy-Schwarz inequality and condition (S1). Inequality (2.15) is a result of Cauchy-Schwarz.

Lemma 2. *The bilinear operator* $[\cdot\,,\cdot]_{X_h}$ *is coercive on* Z_h*:*

$$[v_h, v_h]_{X_h} \geq C\|v_h\|_{div}^2 \;\forall v_h \in Z_h. \tag{2.16}$$

Inequality (2.16) is a direct consequence of condition (S1). Before we can establish the inf-sup condition for the bilinear operator $[\cdot\,, \mathcal{DIV}\,\cdot]_{Q_h}$, we will need the following two lemmas.

Lemma 3. *For any* $q \in L^2(\Omega)$ *there exists* $v \in (H^1(\Omega))^d$ *such that*

$$\nabla \cdot v = q \tag{2.17}$$

and

$$\|v\|_{(H^1(\Omega))^d} \leq C\|q\|_{L^2(\Omega)}.$$

Proof. The function v can chosen by solving the problem,

$$\Delta\phi = q,$$

with homogeneous Dirichlet boundaries. We can then set $v = \nabla\phi$ and use Lax-Milgram and elliptic regularity. $\square$

We also use the following trace inequality originally attributed to Agmon [1].

Lemma 4. *For all* $\chi \in H^1(E)$

$$\|\chi\|^2_{L^2(e)} \leq C\left(h_E^{-1}\|\chi\|^2_{L^2(E)} + h_E|\chi|^2_{H^1(E)}\right).$$

Theorem 1 (inf-sup)**.** *There exists a positive* β *such that, for any* $q_h \in Q_h$,

$$\sup_{\{v_h \in X_h, v_h \neq 0\}} \frac{[\mathcal{DIV}\, v_h, q_h]_{Q_h}}{\|v_h\|_{div}} \geq \beta\|q_h\|_{Q_h}.$$

Proof. It is sufficient to show that there exists $v_h \in X_h$ for each $q_h \in Q_h$ such that

$$\begin{aligned} \mathcal{DIV}\, v_h &= q_h, \\ \|v_h\|_{X_h} &\leq C\|q_h\|_{Q_h}. \end{aligned}$$

Since

$$\begin{aligned} \|v_h\|_{\text{div}}^2 &= \|v_h\|_{X_h}^2 + \|\mathcal{DIV}\, v_h\|_{Q_h}^2 \\ &= \|v_h\|_{X_h}^2 + \|q_h\|_{Q_h}^2 \\ &\leq (1 + C)\|q_h\|_{Q_h}^2. \end{aligned} \tag{2.18}$$

We start by defining $q \in L^2(\Omega)$,

$$q = q_h|_E.$$

We find $v \in (H^1(\Omega))^d$ by Lemma 3 such that

$$\nabla \cdot\, v = q.$$

Next, we set $v_h = (v)^I \in X_h$, and note that

$$\begin{aligned}
[v_h, v_h]_E &\le C|E| \sum_{e\in\partial E} |v_E^e|^2 \text{ (condition S1)} \\
&= C|E| \sum_{e\in\partial E} \left(\frac{1}{|e|}| \int_e v \cdot \mathbf{n}_E^e \ dS|\right)^2 \\
&\le C \sum_{e\in\partial E} \frac{|E|}{|e|} \|v\|_{L^2(e)}^2 \\
&\le C \sum_{e\in\partial E} \frac{|E|}{|e|} \left(h_E^{-1} \|v\|_{L^2(E)^d}^2 + h_E |v|_{H^1(E)^d}^2\right) \text{ (Lemma 4)} \\
&\le C \sum_{e\in\partial E} \left(\|v\|_{L^2(E)^d}^2 + h_E^2 |v|_{H^1(E)^d}^2\right) \\
&\le C_2 \|v\|_{(H^1(E))^d}^2. \qquad (2.19)
\end{aligned}$$

Therefore,

$$\|v_h\|_{X_h} \le C_2 \|v\|_{(H^1(\Omega))^d}. \qquad (2.20)$$

From Lemma 3 we have that

$$\begin{aligned}
\|v\|_{(H^1(\Omega))^d} &\le C_1 \|q\|_{L^2(\Omega)} \\
&= C_1 \|q_h\|_{L^2(\Omega)} \\
&= C_1 \|q_h\|_{Q_h}. \qquad (2.21)
\end{aligned}$$

Therefore,

$$\|v_h\|_{X_h} \le C \|q_h\|_{Q_h}. \qquad (2.22)$$

□

2.3 Velocity Convergence

In this section we establish a first-order *a-priori* error estimate for the velocity variable. We will use the following lemma from [12].

Lemma 5. *For $\phi \in H^2(E)$, there exists a linear function ϕ_E^1 such that*

$$\|\phi - \phi_E^1\|_{L^2(E)} \leq Ch_E^2 |\phi|_{H^2(E)}, \tag{2.23}$$

$$\|\phi - \phi_E^1\|_{H^1(E)} \leq Ch_E |\phi|_{H^2(E)}. \tag{2.24}$$

Combining Lemma 5 and Lemma 4, we get the following lemma.

Lemma 6. *For $\phi \in H^2$, there exists a linear function ϕ_E^1 such that*

$$\|\phi - \phi_E^1\|_{L^2(e)}^2 \leq Ch_E^3 |\phi|_{H^2(E)}^2. \tag{2.25}$$

Lemma 7. *If $v \in (H^1(E))^d$, then, for any face e,*

$$\|v \cdot n\|_{H^{1/2}(e)}^2 \leq c \left(h_E^{-1} \|v\|_{(L^2(E))^d}^2 + h_E |v|_{(H^1(E))^d}^2 \right). \tag{2.26}$$

Lemma 7 was proven in [38]. The main result of this section is the following theorem bounding the velocity error for the proposed form.

Theorem 2 (Velocity Estimate)**.** *For the exact solution (p, v) of (2.1) and MFD approximation (p_h, v_h) solving (2.11-2.12) , there exists a C, independent of h, such that*

$$\|v^I - v_h\|_{X_h} \leq Ch(|p|_{H^2(\Omega)} + |p|_{H^1(\Omega)} + |v|_{(H^1(\Omega))^d}). \tag{2.27}$$

Proof. Set $\mathbf{v} = v^I - v_h$. Note that

$$\mathcal{DIV}\,\mathbf{v} = \mathcal{DIV}\,(v^I - v_h) = f^I - f^I = 0.$$

Thus,

$$\begin{aligned} \|v^I - v_h\|^2_{X_h} &= [v^I - v_h, \mathbf{v}]_{X_h} \\ &= [v^I, \mathbf{v}]_{X_h} - [p_h, \mathcal{DIV}\,\mathbf{v}]_{Q_h} \\ &= [v^I, \mathbf{v}]_{X_h}. \end{aligned} \tag{2.28}$$

By adding and subtracting $(\mathbf{K}\nabla p^1)^I$,

$$[v^I, \mathbf{v}]_{X_h} = \underbrace{[v^I + (\mathbf{K}\nabla p^1)^I, \mathbf{v}]_{X_h}}_{I_1} \underbrace{-[(\mathbf{K}\nabla p^1)^I, \mathbf{v}]_{X_h}}_{I_2}. \tag{2.29}$$

We now bound I_1 and I_2. Starting with I_1,

$$\begin{aligned} |I_1| &\le C\|(v + \mathbf{K}\nabla p^1)^I\|_{X_h}\|\mathbf{v}\|_{X_h} \\ &\le C\left(\sum_{E\in\Omega}\sum_{e\in\partial E}\left(((v + \mathbf{K}_E\nabla p^1)^I)^e_E\right)^2|E|\right)^{1/2}\|\mathbf{v}\|_{X_h} \quad \text{(using (2.6))} \\ &= C\left(\sum_{E\in\Omega}\sum_{e\in\partial E}\left(\frac{1}{|e|}\int_e (v + \mathbf{K}_E\nabla p^1)\cdot\mathbf{n}^e_E\ dS\right)^2|E|\right)^{1/2}\|\mathbf{v}\|_{X_h} \quad \text{(using (2.5))} \\ &\le C\left(\sum_{E\in\Omega}\sum_{e\in\partial E}\left(\frac{1}{|e|}\|(v + \mathbf{K}_E\nabla p^1)\cdot\mathbf{n}^e_E\|^2_{L^2(e)}|E|\right)^{1/2}\right)\|\mathbf{v}\|_{X_h} \quad \text{(using Hölder)} \\ &\le C\left(\sum_{E\in\Omega}\sum_{e\in\partial E}\left(\|(v + \mathbf{K}_E\nabla p^1)\cdot\mathbf{n}^e_E\|^2_{H^{1/2}(e))}h_E\right)^{1/2}\right)\|\mathbf{v}\|_{X_h} \\ &\le C\left(\sum_{E\in\Omega}\left(h_E^{-1}\|(v + \mathbf{K}_E\nabla p^1\|^2_{(L^2(E))^d} + h_E|v|^2_{(H^1(E))^d}\right)h_E\right)^{1/2}\|\mathbf{v}\|_{X_h} \\ &\qquad \text{(using Lemma (7))}. \end{aligned} \tag{2.30}$$

Taking the first term, we add and subtract both $K\nabla p$ and $K\nabla p_E^1$. By doing so, we are left with

$$
\begin{aligned}
||v + \mathbf{K}_E \nabla p_E^1||_{(L^2(E))^d} &\leq ||K\nabla(p - p_E^1)||_{(L^2(E))^d} + ||(K - \mathbf{K}_E)\nabla p_E^1||_{(L^2(E))^d} \\
&\leq C\left(h_E|p|_{H^2(E)} + h_E||\nabla p_E^1||_{(L^2(E))^d}\right) \qquad \text{(using Lemma(5))} \\
&\leq C\left(h_E|p|_{H^2(E)} + h_E|p_E^1|_{H^1(E)}\right).
\end{aligned}
$$

For the last inequality, we use

$$
||\nabla p_E^1||_{(L^2(E))^d} \leq ||\nabla p||_{(L^2(E))^d} + ||\nabla(p - p_E^1)||_{(L^2(E))^d} \leq C|p|_{H^1(E)}.
$$

The final bound on I_1 is,

$$
|I_1| \leq Ch\left(|p|_{H^2(\Omega)} + |p|_{H^1(\Omega)} + |v|_{(H^1(\Omega))^d}\right)\|\mathbf{v}\|_{X_h}.
$$

Given that $\mathcal{DIV}\,\mathbf{v} = 0$ and condition ($\tilde{\text{S}}$2), the expression for I_2 becomes

$$
I_2 = -\sum_{E\in\Omega}\sum_{e\in\partial E} v_E^e \int_e w_e p_E^1 \, dS. \tag{2.31}
$$

Due to the continuity of p, we can subtract it from each element face in the summation above, giving us,

$$
\begin{aligned}
|I_2| &= \left|\sum_{E\in\Omega}\sum_{e\in\partial E} v_E^e \int_e w_e(p_E^1 - p)\, dS\right| \\
&\leq \sum_{E\in\Omega}\sum_{e\in\partial E} |e|^{1/2}|w_e^{max}|\; |v_E^e|\; \|p_E^1 - p\|_{L^2(e)} \\
&\leq C\sum_{E\in\Omega}\left(|E|\sum_{e\in\partial E}|v_E^e|^2\right)^{1/2} h_E|p|_{H^2(E)} \\
&\leq Ch|p|_{H^2(\Omega)}\|\mathbf{v}\|_{X_h}.
\end{aligned} \tag{2.32}
$$

Combining the bounds on I_1 and I_2 gives us our desired estimate.

□

2.4 Pressure Convergence

We start by defining a weighted projection operator, I_w,

$$(p)^{I_w} = \frac{1}{|E|} \int_E w_E p \; dV \tag{2.33}$$

We proceed by establishing pressure convergence for a weighted version of the norm. We later establish a bound on the unweighted norm.

Theorem 3. *Let* (p, v) *be the exact solution to (2.1), and let* (p_h, v_h) *be the MFD approximation. Assuming* $p \in H^2(\Omega)$ *and* $v \in (H^1(\Omega))^d$*, there exists a constant* C*, independent of* h*, such that,*

$$\|p^{I_w} - p_h\|_{Q_h} \leq Ch(|p|_{H^2(\Omega)} + |v|_{(H^1(\Omega))^d}). \tag{2.34}$$

Proof. From Theorem (1) we know that

$$\|p^{I_w} - p_h\|_{Q_h} \leq \frac{1}{\beta} \sup_{\mathbf{v} \in X_h, \mathbf{v} \neq 0} \frac{[\mathcal{DIV}\,\mathbf{v}, p^{I_w} - p_h]_{Q_h}}{\|\mathbf{v}\|_{\text{div}}}.$$

Adding and subtracting $(w_E p^1)^I_E$, we get

$$[\mathcal{DIV}\,\mathbf{v}, p^{I_w} - p_h]_{Q_h} = [\mathcal{DIV}\,\mathbf{v}, (p - p^1)^{I_w}]_{Q_h} - [\mathcal{DIV}\,\mathbf{v}, p_h]_{Q_h} + [\mathcal{DIV}\,\mathbf{v}, (p^1)^{I_w}]_{Q_h}. \tag{2.35}$$

Using for the second term Equation (2.11) and condition ($\tilde{\text{S}}2$) for the third term, we have

$$\begin{aligned} [\mathcal{DIV}\,\mathbf{v}, p^I - p_h]_{Q_h} = {} & \underbrace{[\mathcal{DIV}\,\mathbf{v}, (p - p^1)^{I_w}]_{Q_h}}_{I_3} - \underbrace{[v_h, \mathbf{v}]_{X_h}}_{I_4} \\ & + \underbrace{\sum_E \sum_e v_E^e \int_e w_e p_E^1 \; dS}_{I_5} - \underbrace{\sum_E [(K_E \nabla p_E^1)^I, \mathbf{v}]_E}_{I_6}. \end{aligned} \tag{2.36}$$

We can bound I_3 by

$$[\mathcal{DIV}\,\mathbf{v}, (p-p^1)^{I_w}]_{Q_h} \le \|(p-p^1)^{I_w}\|_{Q_h}\|\mathcal{DIV}\,\mathbf{v}\|_{Q_h} \tag{2.37}$$

$$\le |w_E^{\max}|\|(p-p^1)^I\|_{Q_h}\|\mathcal{DIV}\,\mathbf{v}\|_{Q_h}. \tag{2.38}$$

Then, using Lemma 5, we have

$$|I_3| \le |w_E^{\max}|Ch^2\|\mathbf{v}\|_{div}|p|_{H^2(\Omega)}. \tag{2.39}$$

Expression I_5 is identical to I_2 in (2.31), giving us

$$|I_5| \le Ch\|\mathbf{v}\|_{div}|p|_{H^2(\Omega)}. \tag{2.40}$$

Taking I_4 and I_6, and adding and subtracting u^I, we have

$$I_4 + I_6 = [(K\nabla p^1)^I + v^I, \mathbf{v}]_{X_h} - [v^I - v_h, \mathbf{v}]_{X_h}, \tag{2.41}$$

$$= \tilde{I}_4 + \tilde{I}_6. \tag{2.42}$$

Expression $\tilde{I}_4$ is identical to I_1 in (2.29), giving the bound

$$|\tilde{I}_4| \le Ch(|p|_{H^2} + |v|_{(H^1(\Omega))^d})\|\mathbf{v}\|_{X_h}. \tag{2.43}$$

Expression $\tilde{I}_6$ is bounded by the velocity estimate in Theorem 2,

$$|\tilde{I}_6| \le \|v^I - v_h\|_{X_h}\|\mathbf{v}\|_{X_h}$$

$$\le Ch(|p|_{H^2} + |v|_{(H^1(\Omega))^d})\|\mathbf{v}\|_{X_h}. \tag{2.44}$$

Combining the bounds on I_3 - I_6 gives the desired result. □

We can now bound the original projection operator (2.4), using the pressure estimate in Theorem 3, giving us,

$$\begin{aligned}\|p^{I_w} - p_h\|_{Q_h}^2 &= \sum_E \left(\frac{1}{|E|} \int_E w_E (p - p_h)\ dV \right)^2 \\ &\geq w_E^{\min} \sum_E \left(\frac{1}{|E|} \int_E (p - p_h)\ dV \right)^2 . \end{aligned} \tag{2.45}$$

Therefore,

$$\|p^I - p_h\|_{Q_h}^2 \leq \frac{1}{w_E^{\min}} \|p^{I_w} - p_h\|_{Q_h}^2 . \tag{2.46}$$

2.5 Weighting Functions

The generalization presented requires the existence of weighting functions w_E and w_e. Since we have placed no restrictions on the positivity of w_e, an affine function will suffice. The authors of [24] presented a proof of the existence of such functions.

Since function w_E must be strictly positive, an affine function may not suffice. To see why this is the case, consider the following example. Let cell E be a line segment of length 1, that is, $E = [0, 1]$. Set the shifted point $x_E = .75$. We seek the function $w_E = ax + b$ such that

$$\int_0^1 w_E \, dV = |E| = 1. \tag{2.47}$$

$$\int_0^1 g w_E \, dV = |E|(.75) = .75 \tag{2.48}$$

To find w_E, construct a linear system of equations with a and b as unknowns. A simple calculation shows that $w_E = 3x - \frac{1}{2}$ is the only affine choice. This function does not satisfy the condition of strict positivity. We can still construct a function w_E that satisfies the conditions if we consider non-affine functions. By inspection, we find that

$$w_E = 3.6x^2 - .6x + .1$$

satisfies Equations (2.47-2.48) and is strictly positive over $[0, 1]$. There are an infinite number of functions that would satisfy the criteria in this example. In the next lemma we will establish the existence of such functions for polyhedra in any dimension.

Lemma 2.5.1. *Given polyhedron E and a point x_E strictly on the interior of E, there exists a bounded function w_E that is strictly positive over E such that*

$$\int_E w_E \, dV = |E|,$$
$$\int_E g w_E \, dV = |E| x_E.$$

Proof. Let C be the centroid of E, $d(C, x_E)$ the distance between C and x_E, and r be the minimum distance between x_E and the nearest boundary. Define point A on the line joining C and x_E, with a distance of $r/2$ from x_E, so x_E is between A and C. Define $B_{r/2}(A)$, the ball of radius $r/2$ centered at A (Fig. 2.5). Define the function w_E as,

$$w_E := \frac{d(C, A) - d(C, x_E)}{d(C, A)} + \frac{|E|}{|B_{r/2}(A)|} \frac{d(C, x_E)}{d(C, A)} \mathbf{1}_{B_{r/2}(A)}.$$

The choice of w_E satisfies Equation 2.47,

$$\begin{aligned}\int_E w_E &= \int_E \frac{d(C,A) - d(C,x_E)}{d(C,A)}\,dV + \int_E \frac{|E|}{|B_{r/2}(A)|}\frac{d(C,x_E)}{d(C,A)}\mathbf{1}_{B_{1/2}(A)}\,dV \\ &= |E|\frac{d(C,A) - d(C,x_E)}{d(C,A)} + |E|\frac{d(C,x_E)}{d(C,A)} \\ &= |E|.\end{aligned}$$

In addition, w_E satisfies Equation 2.48,

$$\begin{aligned}\int_E g w_E\,dV &= \int_E g\left(\frac{d(C,A) - d(C,x_E)}{d(C,A)}\right)dV + \int_{B_{r/2}(A)} x\left(\frac{|E|}{|B_{r/2}(A)|}\frac{d(C,x_E)}{d(C,A)}\right)dV \\ &= C|E|\left(\frac{d(C,A) - d(C,x_E)}{d(C,A)}\right) + |E|A\frac{d(C,x_E)}{d(C,A)}.\end{aligned}$$

Without loss of generality, we can place C at the origin. Note that $A\frac{d(C,x_E)}{d(C,A)}$ is a point of distance $d(C,x_E)$ from C on the line joining C and A, which must be x_E. Therefore,

$$\int_E g w_E\,dV = |E|x_E.$$

□

2.6 Constructing the Linear System

We have explicitly defined the $\mathcal{DIV}$ operator, as well as the pressure inner product $[\cdot,\cdot]_{Q_h}$. What remains is to define the velocity inner product $[\cdot,\cdot]_{X_h}$. Recall that this inner product is defined relative to "local" inner products,

$$[v_h, w_h]_{X_h} = \sum_{E\in\mathcal{T}_h} [v_h, w_h]_E. \tag{2.49}$$

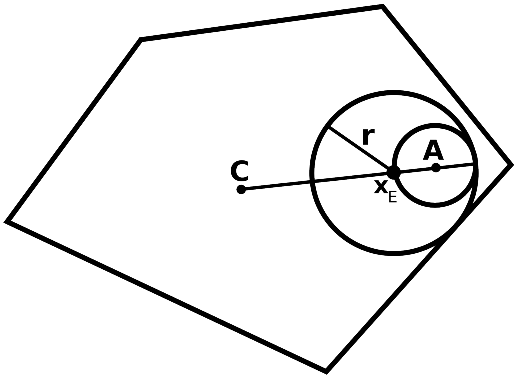


Figure 2.1: The points needed in the construction of weighting function w_E. The weighting function is the sum of two piece-wise constants: one is defined over the entire cell, and the other is defined over the ball centered at A. The result is a shifting of the center of gravity from C to A.

The discretization yields a convergent solution if the inner product satisfies the stability (S1) and consistency ($\tilde{\text{S}}$2) conditions. The second condition is the main focus of constructing the appropriate inner product. It is helpful to take a detailed look at constructing a velocity inner product that satisfies this condition in two dimensions. A linear function q^1 can be expressed as

$$q^1 = a_1 x + a_2 y + a_3. \tag{2.50}$$

We want the following to be exact for all q^1 and all $v_h \in X_h$,

$$[(\mathbf{K}_E \nabla q^1)^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e w_e q^1 \, dS - \int_E w_E q^1 (\mathcal{DIV}\, v_h)_E \, dV. \tag{2.51}$$

Therefore,

$$[(\mathbf{K}_E \nabla (a_1 x + a_2 y + a_3))^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e w_e (a_1 x + a_2 y + a_3) \, dS - \int_E w_E (a_1 x + a_2 y + a_3)(\mathcal{DIV}\, v_h)_E \, dV.$$

The constant term does not contribute as $\nabla a_3 = 0$ and

$$\begin{aligned}\int_E w_E a_3 (\mathcal{DIV}\, v_h)_E \, dV &= a_3 |E| (\mathcal{DIV}\, v_h)_E \\ &= a_3 \sum_{e \in \partial E} v_E^e |e| \\ &= \sum_{e \in \partial E} v_E^e \int_e w_e a_3 \, dS.\end{aligned}$$

Leaving us with

$$\begin{aligned}[(\mathbf{K}_E \nabla (a_1 x + a_2 y))^I, v_h]_E = &\sum_{e \in \partial E} v_E^e \int_e w_e (a_1 x + a_2 y) \, dS \\ &- \int_E w_E (a_1 x + a_2 y)(\mathcal{DIV}\, v_h)_E \, dV.\end{aligned}$$

By linearity of the gradient and integration operators, we note that the equality will hold if it holds for x and y separately,

$$[(\mathbf{K}_E \nabla x)^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e w_e x \, dS - \int_E w_E x (\mathcal{DIV}\, v_h)_E \, dV, \tag{2.52}$$

$$[(\mathbf{K}_E \nabla y)^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e w_e y \, dS - \int_E w_E y (\mathcal{DIV}\, v_h)_E \, dV. \tag{2.53}$$

Recall from the definitions of w_E and w_e that $\int_E g w_E \, dS = |E| x_E$, and $\int_e g w_e \, dS = |e| x_e$, leaving

$$\begin{aligned}[(\mathbf{K}_E \nabla x)^I, v_h]_E &= \sum_{e \in \partial E} |e| v_E^e x_e^x - |E| (\mathcal{DIV}\, v_h)_E x_E^x, \\ [(\mathbf{K}_E \nabla y)^I, v_h]_E &= \sum_{e \in \partial E} |e| v_E^e x_e^y - |E| (\mathcal{DIV}\, v_h)_E x_E^y.\end{aligned}$$

Let us now examine the left side of the condition ($\tilde{\mathrm{S}}$2). First, we partition $\mathbf{K}_E$ by columns,

$$\mathbf{K}_E = [k_1 k_2]. \tag{2.54}$$

Expanding the rest of the expression, we have

$$\begin{aligned}[(\mathbf{K}_E \nabla x)^I, v_h]_E &= [(k_1 \cdot \mathbf{n}_{e_1}, ..., k_1 \cdot \mathbf{n}_{e_{k_E}}), (v_E^{e_1}, ..., v_E^{e_{k_E}})]_E \\ &= (v_E^{e_1}, ..., v_E^{e_{k_E}}) M_E (k_1 \cdot \mathbf{n}_{e_1}, ..., k_1 \cdot \mathbf{n}_{e_{k_E}})^T. \end{aligned} \tag{2.55}$$

Similarly for y,

$$[(\mathbf{K}_E \nabla y)^I, v_h]_E = (v_E^{e_1}, ..., v_E^{e_{k_E}}) M_E (k_2 \cdot \mathbf{n}_{e_1}, ..., k_2 \cdot \mathbf{n}_{e_{k_E}})^T.$$

Taking this equality to hold for each component in v_h and setting

$$R_E = \begin{pmatrix} |e_1|(x_{e_1} - x_E) \\ \vdots \\ |e_{k_E}|(x_{e_{k_E}} - x_E) \end{pmatrix}, \tag{2.56}$$

and

$$N_E = \begin{pmatrix} \mathbf{K}_E \, \mathbf{n}_{e_1} \\ \vdots \\ \mathbf{K}_E \, \mathbf{n}_{e_{k_E}} \end{pmatrix}, \tag{2.57}$$

condition ($\tilde{\text{S}}2$) becomes

$$M_E N_E = R_E. \tag{2.58}$$

Note that by setting $w_E = 1$ and $w_e = 1$ we retrieve the original standard definition of MFD, in which case, x_E corresponds to the centroid of element E, and x_e would correspond to the centroid of face e. Also note that we do not require explicit construction of the weighting functions in order to build the linear system.

The authors of [16] demonstrated how to construct M_E to satisfy (2.58). This is done by first defining $C_E \in \mathbb{R}^{k_E \times k_E - d}$ such that

$$N_E^T C_E = 0.$$

In [16], they noted without weights,

$$N_E^T R_E = |E| \mathbf{K}_E \tag{2.59}$$

so they defined M_0 as

$$M_0 = \frac{1}{|E|} R_E \mathbf{K}_E^{-1} R_E^T. \tag{2.60}$$

The result is that, for any positive-definite $U_E \in \mathbb{R}^{(K_E - d) \times (K_E - d)}$,

$$M_E = M_0 + C_E U_E C_E^T. \tag{2.61}$$

Due to the modification ($\tilde{\text{S}}2$) of condition (S2), equality (2.59) no longer holds. However, we can still proceed by following the same form proposed by [35]

$$M_0 = R_E (R_E^T N_E)^{-1} R_E^T. \tag{2.62}$$

Doing so, we see that (2.61) gives (2.58), since

$$M_0 N_E = R_E (R_E^T N_E)^{-1} R_E^T N_E = R_E. \tag{2.63}$$

Note that $R_E^T N_E$ automatically reduces to $|E| \mathbf{K}_E^{-1}$ in the case when no boundary points are shifted. In general, $R_E^T N_E$ may not always be invertible. An

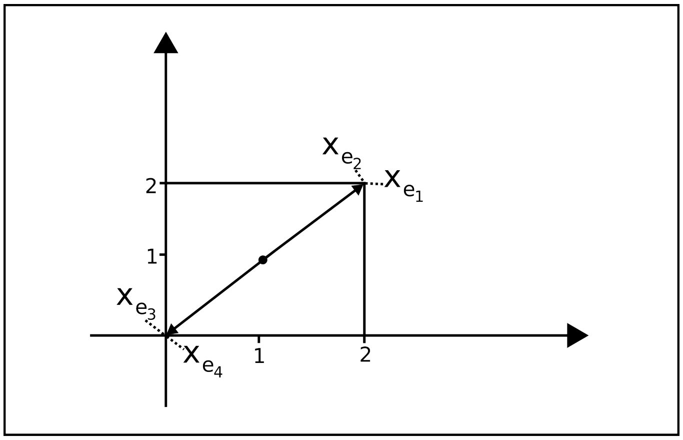


Figure 2.2: Example of a cell with boundary points shifted to the top right and bottom left corners. In this example, matrix $R_E^T N_E$ is singular.

example can be seen in (Fig. 2.2), in which the points on the faces have been shifted to the corners of a square cell. In this case, we have

$$R_E = \begin{pmatrix} 2 & 2 \\ 2 & 2 \\ -2 & -2 \\ -2 & -2 \end{pmatrix} \text{ and } N_E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}.$$

The resulting matrix is

$$R_E^T N_E = \begin{pmatrix} 4 & 4 \\ 4 & 4 \end{pmatrix},$$

which is singular.

Chapter 3

MFD and Two-Point Fluxes

In this chapter, we take a closer look at finite volume methods and two-point fluxes (TPFA). We establish the general conditions for equating an MFD method to the two-point flux scheme. The generalized MFD definition in the previous chapter is leveraged to connections to TPFA and MFD in the case of general K-orthogonal meshes. These include general Voronoi diagrams and 2.5 dimensional Voronoi diagrams. In addition, we demonstrate the connection between the new form and point-centered schemes.

3.1 Finite Volume Methods (FVM) and MFD

Finite volume methods (FVM) are based on an application of the divergence theorem to each element E in the domain,

$$\int_E \nabla \cdot v \; dV = \frac{1}{|E|} \sum_{e \in \partial E} |e| v_E^e \cdot \mathbf{n}_E^e. \tag{3.1}$$

Finite volume discretizations are distinguished by the manner in which the flux (v_E^e) is calculated. In the case of porous media applications, the flux of the fluid in the subsurface is a function of the fluid pressure. Each element in the domain is given a single, piece-wise constant pressure degree of freedom (p_E), and each face in the mesh is given a single flux representing the normal component of the velocity across that face (v_E^e). The flux is then related to the pressure via a function

$$v_E^e = F(p_h). \tag{3.2}$$

The choice of function F is what distinguishes a particular FV method. When the velocity at face e is approximated using *two* adjacent pressures, we have a two-point flux approximation (TPFA). When more than two pressures are used, the method is referred to as a multi-point flux approximation (MPFA).

The MFD method, much like the related MFE method, produces a saddle-point problem with both velocity and pressure unknowns:

$$\begin{pmatrix} M & -\mathcal{DIV}^* \\ \mathcal{DIV} & 0 \end{pmatrix} \begin{pmatrix} v_h \\ p_h \end{pmatrix} = \begin{pmatrix} 0 \\ f^I \end{pmatrix}. \tag{3.3}$$

One can observe the relation between FVM and MFD by explicitly forming the velocity unknown as a function of pressure via the Schur compliment of the saddle-point problem,

$$v_h = -M^{-1}\mathcal{DIV}^*\, p_h = F_{\mathrm{MFD}}(p_h). \tag{3.4}$$

The nature of the relation between p_h and v_h is a direct consequence of the structure of M^{-1}. In general, the matrix M^{-1} is dense, causing the velocity at every face to be a function of all the pressure degrees of freedom. However, constructing a diagonal matrix M leads trivially to a diagonal M^{-1}. In this case, due to the structure of matrix $\mathcal{DIV}^*$, velocities are a linear function of two pressures, resulting in a TPFA scheme. Thus our objective in this work will be to generate diagonal matrices M.

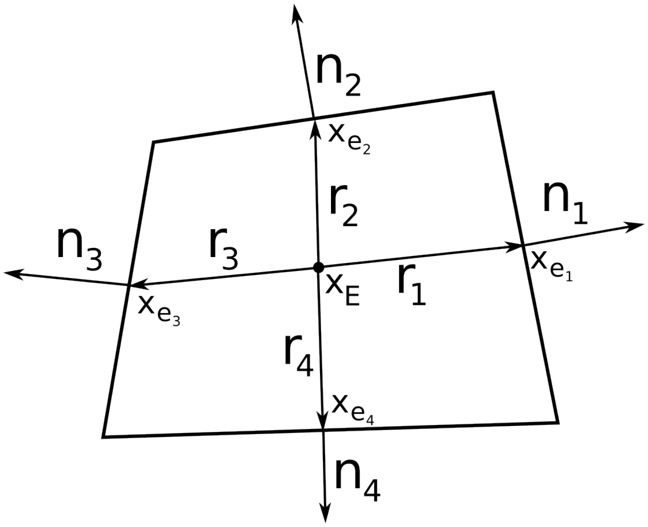


Figure 3.1: The vectors r are constructed by connecting x_E with the appropriate point x_e. These vectors make up the rows of matrix R_E, $(|e_1|r_1^T...|e_4|r_4^T)$. The rows of matrix N_E are the normal vectors multiplied by $\mathbf{K}_E$, $((\mathbf{K}_E n_1)^T...(\mathbf{K}_E n_4)^T)$. The original MFD sets the point x_E to the centroid of the cells and the points x_e to the the centroids of the faces. Our generalization allows the point x_E to be shifted on the interior of the cell, and the points x_e to be shifted on the plane of the faces. By shifting these points, we can establish collinearity vectors r and n, resulting in a diagonal matrix M_E.

Fortunately, there is a simple geometric criterion that indicates when diagonality can be achieved in MFD. Recall that matrix M is a global matrix formed from the summation of local matrices, M_E,

$$M = \sum_E M_E. \tag{3.5}$$

Due to the consistency condition ($\tilde{\text{S}}2$), M_E must satisfy the relation

$$M_E N_E = R_E. \tag{3.6}$$

Recall that R_E is a $k_E \times d$ matrix with each row corresponding to the vectors $|e|(x_e - x_E)$ (2.56). From (2.57), the rows of matrix N_E correspond to the K-normal components to face e [Fig 3.1]. When the rows of N_E are collinear to the rows of R_E, a simple scaling of the rows of N_E satisfies (3.6), *i.e.*, a diagonal matrix M_E suffices. Therefore, our objective is to construct matrices R_E with rows collinear to the rows of N_E.

3.1.1 Previous Work

In the original definition of MFD, a diagonal matrix M_E can be defined immediately for rectangular meshes and for special cases such as regular polygonal meshes. In order to achieve collinearity of the rows of R_E and the normals of the faces to more elements, it is necessary to detach the definition of R_E from the centroids of the element and faces. This “point-shifting” procedure is accomplished by modifying the consistency condition (S2), and such modifications have been suggested in the literature. Recall that the original condition (S2) states that for all linear functions q^1 and $v_h \in X_h$, the local inner-product $[\cdot,\cdot]_E$ must satisfy

$$[(\mathbf{K}_E \nabla q^1)^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e q^1 \, dS - \int_E q^1 (\mathcal{DIV}\, v_h)_E \, dV. \tag{3.7}$$

The first example of a modified condition was provided by Droniou *et al.* [24]. They suggested weighting the volume integral by a linear function w_E:

$$[(\mathbf{K}_E \nabla q^1)^I, v_h]_E = \sum_{e \in \partial E} v_E^e \int_e q^1 \, dS - \int_E w_E q^1 (\mathcal{DIV}\, v_h)_E \, dV. \tag{3.8}$$

The function w_E has the properties $\int_E w_E \;\, dV = |E|$ and $\int_E g w_E \;\, dV = |E| x_E$. The modification results in a shifting of the point used on the interior of the element from the centroid to a new point x_E. This modification allowed Droniou *et al.* to reduce MFD to TPFA for the case of acute triangular meshes in two dimensions. This works by setting point x_E to the intersection of the orthogonal bisectors of the triangle.

In the work by Lipnikov *et al.* [38], the authors suggest another modification of condition (S2). For a subset $\bar{e} \subset e$, and all linear functions q^1, let

$$[(\mathbf{K}_E \nabla q^1)^I, v_h]_E = \sum_{e \in \partial E} v_E^e |\bar{e}| \int_{\bar{e}} q^1 \, dS - \int_E q^1 (\mathcal{DIV}\, v_h)_E \, dV. \tag{3.9}$$

This modification allows the point on the boundary to be shifted to the centroid of $\bar{e}$. The Lipnikov *et al.* form of (S2) extended MFD to TPFA for the case of centroidal Voronoi diagrams.

3.1.2 General Voronoi Diagrams

A Voronoi diagram is a tessellation of $\mathbb{R}^d$ relative to a set of points known as "generators."

Definition 1. *Voronoi Diagram. Given a set of generating points,*

$$V = \{V_i \in \mathbb{R}^d\},$$

we define the Voronoi tessellation $\mathcal{T}_v = \{E_i\}$ *as*

$$E_i = \{x \in \mathbb{R}^d \; | d(x, V_i) \leq d(x, V_j) \; \forall i \neq j\}. \tag{3.10}$$

The set of generating points uniquely defines a Voronoi diagram. Note that, in this definition, the domain is infinite. For our purposes we will focus on what is known as a *bounded Voronoi diagram*, which is defined over a bounded domain Ω as follows.

Definition 2. *Bounded Voronoi Diagram. Given a domain* $\Omega \subset \mathbb{R}^d$*, and a set of generating points* $V = \{V_i \in \mathbb{R}^d \; | V_i \in \Omega\}$*, we define the bounded Voronoi tessellation* $\mathcal{T}_v = \{E_i\}$ *of* Ω *by*

$$E_i = \{x \in \Omega \; | d(x, V_i) \leq d(x, V_j) \; \forall i \neq j\}. \tag{3.11}$$

Since we are only concerned with bounded domains, we refer to a *bounded Voronoi diagram* simply as a *Voronoi diagram.* Each cell E_i is called a Voronoi polygon/polyhedron. It is often noted that Voronoi diagrams are the geometric dual of Delaunay triangulations. This relationship requires the inclusion of degenerate faces ($|e| = 0$) in the Voronoi diagram. For our purposes we exclude these degenerate faces from the tessellation.

We say that two Voronoi cells (and their generating points) are adjacent if they share a non-trivial Voronoi face. It is a direct consequence of the definition of Voronoi diagrams that the line joining two adjacent generating points is always perpendicular to the common face between them. For two adjacent cells, V_i and V_j, with associated face e, we refer to the midpoint between them as b_e.

Bounded diagrams introduce certain challenges because fundamental properties of the Voronoi diagram are broken at the boundary. For example, a non-convex boundary could lead to non-convex cells. An extreme case can be considered by selecting a single generating point in an arbitrarily shaped domain. In order to simplify the problem, we only consider convex boundaries. In addition, we require that the orthogonality of the diagram is maintained at the domain boundary.

Lemma 3.1.1. *For a Voronoi polyhedron with isotropic, piece-wise constant permeability ($\mathbf{K}_E = \kappa_E I$), by setting $x_E = V_E$ and $x_e = b_e$, a diagonal M_E can be constructed that satisfies both conditions (S1) and ($\tilde{S}2$).*

Proof. Since the line joining two adjacent generating points is orthogonal to the Voronoi face between them, the vectors r_e are collinear to n_e. That is,

$$r_e = \frac{|e| \|r_e\|}{\kappa_E} n_e.$$

Therefore, a diagonal matrix M_E with choice of entries $(M_E)_{ii} = \frac{|e_i|\|r_{e_i}\|}{\kappa_E}$ satisfies condition ($\tilde{\text{S}}2$). ☐

Note that the point b_e may fall outside of the boundary faces, see [Fig. 3.2]. This, however, presents no problems for our definition, as we allow the weighting function w_e to shift the boundary point x_e outside of the face.

In porous media applications, it is common to use the so-called 2.5-dimensional Voronoi mesh. These diagrams are constructed by forming a two-dimensional Voronoi diagram. Cells are then vertically extruded into three-dimensional prisms. A 2.5-dimensional Voronoi mesh allows for slightly greater flexibility in permeability tensor when establishing TPFA.

Lemma 3.1.2. *Given a mesh that is a Voronoi diagram in the x-y direction and orthogonally extruded in the z direction, let*

$$K_E = \begin{pmatrix} \kappa_x & 0 & 0 \\ 0 & \kappa_y & 0 \\ 0 & 0 & \kappa_z \end{pmatrix},$$

with $\kappa_x = \kappa_y$. A diagonal M_E that satisfies (S1) and ($\tilde{S}2$) exists.

Proof. In this case, a diagonal M_E can be constructed by setting

$$(M_E)_{ii} = \frac{|e_i|\|r_{e_i}\|}{\kappa_x}$$

for faces e facing in the x-y direction, and

$$(M_E)_{ii} = \frac{|e_i|\|r_{e_i}\|}{\kappa_z}$$

for faces on the z-plane. ☐

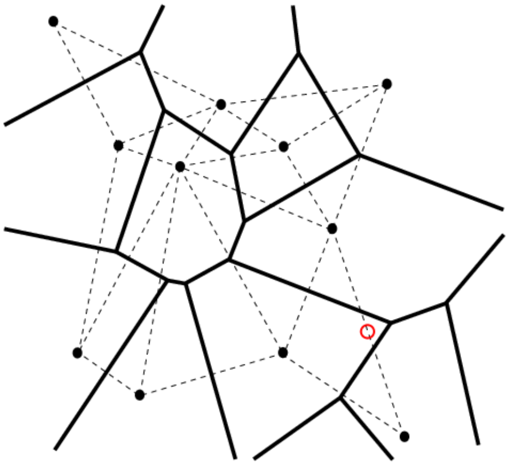


Figure 3.2: A two-dimensional Voronoi diagram (solid line) and its geometric dual, the Delaunay triangulation (dashed line). The dashed lines are always orthogonal to the corresponding solid line. The red circle indicates how the intersection of the two lines may occur outside of the face boundaries.

3.2 Cell-Centered Schemes

In the cell-centered finite difference scheme, the computational domain is first divided into rectangles. These rectangles constitute the "control volumes" of the method. For each rectangle, the center of mass is used as the "center" of the cell. It is from this center that a distance to neighboring cells is computed. Following the notation used in [17], the rectangles are specified by a series of points in x ($x_{-1+1/2}, x_{0+1/2}, x_{1+1/2}, ..., x_{i+1/2}$) and in y $(y_{-1+1/2}, y_{0+1/2}, y_{1+1/2}, ..., y_{j+1/2})$. The cell centers of mass are denoted by the sequences $(x_0, x_1, ..., x_i)$ and $(y_0, y_1, ..., y_j)$. The distance between two adjacent cells is denoted by

$$h_{x,i-1/2} = x_i - x_{i-1}, \tag{3.12}$$

$$h_{y,j-1/2} = y_i - y_{i-1}. \tag{3.13}$$

$$\tag{3.14}$$

Cell widths and heights are computed by

$$h_{x,i} = x_{i+1/2} - x_{i-1/2}, \tag{3.15}$$

$$h_{y,j} = y_{i+1/2} - y_{i-1/2}. \tag{3.16}$$

For cells in the interior of the domain, we have the following stencil [17]:

$$-\left(\frac{p_{i+1,j} - p_{i,j}}{h_{x,i+1/2}} + \frac{p_{i-1,j} - p_{i,j}}{h_{x,i-1/2}}\right)\frac{1}{h_{x,i}} - \left(\frac{p_{i,j+1} - p_{i,j}}{h_{y,i+1/2}} + \frac{p_{i,j-1} - p_{i,j}}{h_{y,i-1/2}}\right)\frac{1}{h_{y,j}} = f_{ij}. \tag{3.17}$$

We can relate this stencil (3.17) directly to MFD. In the case of rectangles, notice that the arrays R_E and N_E (see (2.56), (2.57)) become

$$R_E = \begin{pmatrix} h_{y,j}(x_{i+1/2} - x_i) & 0 \\ 0 & h_{x,i}(y_{j+1/2} - y_j) \\ h_{y,j}(x_{i-1/2} - x_i) & 0 \\ 0 & h_{x,i}(y_{j-1/2} - y_j) \end{pmatrix}, \tag{3.18}$$

$$N_E = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{3.19}$$

Recall that, in order to satisfy condition ($\tilde{\text{S}}2$), we must construct a matrix M_E such that

$$M_E R_E = N_E. \tag{3.20}$$

We can construct a diagonal matrix to satisfy this condition, as follows:

$$M_E = \begin{pmatrix} h_{y,j}(x_{i+1/2} - x_i) & 0 & 0 & 0 \\ 0 & h_{x,i}(y_{j+1/2} - y_j) & 0 & 0 \\ 0 & 0 & h_{y,j}(x_{i-1/2} - x_i) & 0 \\ 0 & 0 & 0 & h_{x,i}(y_{j-1/2} - y_j) \end{pmatrix}. \tag{3.21}$$

This is the expression for the local matrix. In order to assemble the global matrix, we must add the local contributions. For cell faces on the interior of the domain we have the following diagonal entries for M:

$$\begin{aligned} M_{e_{i+1/2,j}} &= h_{y,j}(x_{i+1} - x_i), \\ M_{e_{i-1/2,j}} &= h_{y,j}(x_i - x_{i-1}), \\ M_{e_{i,j+1/2}} &= h_{x,i}(y_{i+1} - y_i), \\ M_{e_{i,j-1/2}} &= h_{x,i}(y_i - y_{i-1}). \end{aligned} \tag{3.22}$$

With the inverse of M being given by the reciprocal of the diagonal entries, taking $M^{-1}(\mathcal{DIV}^*)$,

$$\begin{aligned} M^{-1}_{e_{i+1/2,j}} \mathcal{DIV}^*{}_{e_{i+1/2,j}} &= (0, ..., \frac{h_{y,j}}{h_{y,j}(x_{i+1} - x_i)}, ..., \frac{-h_{y,j}}{h_{x,j}(x_{i+1} - x_i)}, ...0), \\ M^{-1}_{e_{i-1/2,j}} \mathcal{DIV}^*{}_{e_{i-1/2,j}} &= (0, ..., \frac{h_{y,j}}{h_{y,j}(x_i - x_{i-1})}, ..., \frac{-h_{y,j}}{h_{y,j}(x_i - x_{i-1})}, ...0), \\ M^{-1}_{e_{i,j+1/2}} \mathcal{DIV}^*{}_{e_{i,j+1/2}} &= (0, ..., \frac{h_{x,i}}{h_{x,i}(y_{i+1} - y_i)}, ..., \frac{-h_{x,i}}{h_{x,j}(y_{i+1} - y_i)}, ...0), \\ M^{-1}_{e_{i,j-1/2}} \mathcal{DIV}^*{}_{e_{i,j-1/2}} &= (0, ..., \frac{h_{x,i}}{h_{x,i}(y_i - y_{i-1})}, ..., \frac{-h_{x,i}}{h_{x,j}(y_i - y_{i-1})}, ...0), \end{aligned} \tag{3.23}$$

leaving

$$\begin{aligned}
M^{-1}_{e_{i+1/2,j}}\mathcal{DIV}^*\ {}_{e_{i+1/2,j}} &= (0, ..., \frac{1}{(x_{i+1}-x_i)}, ..., \frac{-1}{(x_{i+1}-x_i)}, ..., 0),\\
M^{-1}_{e_{i-1/2,j}}\mathcal{DIV}^*\ {}_{e_{i-1/2,j}} &= (0, ..., \frac{1}{(x_{i}-x_{i-1})}, ..., \frac{-1}{(x_{i}-x_{i-1})}, ..., 0),\\
M^{-1}_{e_{i,j+1/2}}\mathcal{DIV}^*\ {}_{e_{i,j+1/2}} &= (0, ..., \frac{1}{(y_{i+1}-y_i)}, ..., \frac{-1}{(y_{i+1}-y_i)}, ..., 0),\\
M^{-1}_{e_{i,j-1/2}}\mathcal{DIV}^*\ {}_{e_{i,j-1/2}} &= (0, ..., \frac{1}{(y_{i}-y_{i-1})}, ..., \frac{-1}{(y_{i}-y_{i-1})}, ..., 0). \qquad (3.24)
\end{aligned}$$

The result is a two-point expression for the flux function. Given pressures p_h, we determine that

$$v_h = M^{-1}\mathcal{DIV}^*\, p_h. \qquad (3.25)$$

We reconstruct our full pressure stencil by taking $\mathcal{DIV}\, M^{-1}\mathcal{DIV}^*$. The coefficients for p_{ij} (the diagonal entries) are

$$-\frac{h_{x,i}}{(x_{i+1}-x_i)} - \frac{h_{x,i}}{(x_{i+1}-x_i)} - \frac{h_{y,j}}{(y_{i+1}-y_i)} - \frac{h_{y,j}}{(y_{i+1}-y_i)}, \qquad (3.26)$$

or

$$-\frac{h_{x,i}}{h_{1,i+1/2}} - \frac{h_{x,i}}{h_{x,i-1/2}} - \frac{h_{y,j}}{h_{2,j+1/2}} - \frac{h_{y,j}}{h_{y,i-1/2}}. \qquad (3.27)$$

Note that by factoring out the cell volume $(h_{x,i}h_{y,j})$ we are left with

$$\left[-\frac{1}{h_{x,i+1/2}} - \frac{1}{h_{x,i-1/2}}\right]\frac{1}{h_{y,j}} + \left[-\frac{1}{h_{y,j+1/2}} - \frac{1}{h_{y,i-1/2}}\right]\frac{1}{h_{x,i}}.$$

For the off-diagonal entries, $\mathcal{DIV}\,\mathcal{DIV}^*$ are only non-zero at shared faces. The resulting values are as follows:

$$p_{i+1,j} \rightarrow \frac{1}{h_{x,i+1/2}h_{y,j}},$$

$$p_{i-1,j} \rightarrow \frac{1}{h_{x,i-1/2}h_{y,j}},$$

$$p_{i,j+1} \rightarrow \frac{1}{h_{y,j+1/2}h_{x,i}},$$

$$p_{i,j-1} \rightarrow \frac{1}{h_{y,j-1/2}h_{x,i}}.$$

These are the same coefficients produced by the cell-centered method. Notice that the formulation did not require invoking the generalization of MFD. The original method introduced in [15] naturally results in the cell-centered scheme over rectangular grids. This is consistent with the notion that the mixed method with RT_0 basis functions is a special case of MFD.

3.3 Point-Centered Schemes

In the cell-centered scheme, we noticed that the cells are chosen first, and then a selected interior point is used as the "center." The selected point is the centroid of the cell. In point-centered schemes the opposite is done. First points in the domain are selected, and then, from these points, cell boundaries are placed half the distance between adjacent points. We again use the notation of [17] and denote the points in x as ($x_1, x_2, ..., x_i$) and the points in y as ($y_1, y_2, ..., y_i$). We have,

$$x_{i-1/2} = \frac{1}{2}(x_{i-1} + x_i), \tag{3.28}$$

$$y_{i-1/2} = \frac{1}{2}(y_{i-1} + y_i), \tag{3.29}$$

$$h_{x,i} = \frac{1}{2}(x_{i+1} - x_{i-1}), \tag{3.30}$$

$$h_{y,i} = \frac{1}{2}(y_{i+1} - y_{i-1}). \tag{3.31}$$

This gives us the same stencil as before, only with new values for h:

$$-\left(\frac{p_{i+1,j} - p_{i,j}}{h_{x,i+1/2}} + \frac{p_{i-1,j} - p_{i,j}}{h_{x,i-1/2}}\right)\frac{1}{h_{x,i}} - \left(\frac{p_{i,j+1} - p_{i,j}}{h_{y,i+1/2}} + \frac{p_{i,j-1} - p_{i,j}}{h_{y,i-1/2}}\right)\frac{1}{h_{y,j}} = f_{ij}. \tag{3.32}$$

We cannot recreate this stencil using the original MFD formulation because MFD automatically orients the matrices around the centroids of the cells and the cell faces. The proposed generalization allows us to shift these points, thus establishing the connection between MFD and the point-centered method.

We define a point x_E on the interior:

$$x_E = (x_i, y_j).$$

For points on the faces we set

$$x_{e_{i+1/2,j+1/2}} = (x_{i+1/2}, y_{j+1/2}),$$

$$x_{e_{i-1/2,j+1/2}} = (x_{i-1/2}, y_{j+1/2}),$$

$$x_{e_{i+1/2,j-1/2}} = (x_{i+1/2}, y_{j-1/2}),$$

$$x_{e_{i-1/2,j-1/2}} = (x_{i-1/2}, y_{j-1/2}).$$

This choice retrieves the expression for matrix R_E (3.18), and all the subsequent calculations for constructing the final stencil follow suit. Therefore, the resulting matrix corresponds to the one produced in the point-centered schemes.

Chapter 4

Fracture Modeling

Fractures are thin cracks in the subsurface that can exist both naturally in rock formations and as the result of human activity (*e.g.*, hydraulic fracturing). The presence of fractures can have a large impact on subsurface flows, as fractures exhibit higher conductivity rates than rock matrix. Such contrasts present challenges when numerically simulating fractures in reservoir problems. Higher fluid velocities inside fractures, when compared to the reservoir, result in an substantial difference in time scales. In addition, linear Darcy flow may not be sufficient to properly capture flow in the fracture. Though fractures are often modeled as simple planar surfaces, they are in fact far more complex. Fractures may curve, branch and even intersect with each other.

In this chapter we present a general methodology based on MFD for coupling fracture and reservoir flow that is capable of handling multiple physical models. This is accomplished separately representing fracture and reservoir models. The two problems are then coupled using appropriate boundary conditions and forcing functions. We present results for fractures with non-planar geometries and show how fracture curvature can impact leakage. We also present results connecting the MFD implementation with existing reservoir simulation code (IPARS) using the multi-point flux Mixed Finite Element method (MFMFE).

4.1 Previous Work

There currently exists an enormous body of literature devoted to solving reservoir flow problems with fractures. Some commonly used approaches to model naturally fractured reservoirs are the dual-porosity and dual-porosity/dual-permeability methods. In these approaches, fractures are modeled as a continua that occupy the same space as the reservoir. Matrix-fracture and fracture-fracture interactions are quantified through transfer terms [58], [34].

A number of methods have been suggested to model fractures explicitly at the interfaces of adjacent cells. The main advantage of such methods is to preserve the orientation of fractures in the domain. One such technique is described in [3], where they rely on non-overlapping domain decomposition to couple the fracture and reservoir problems. Another approach for explicit modeling of fractures at interfaces is described in [29]. The authors are able to handle cases with heterogeneous capillary pressures across fracture boundaries. A hybridized, mixed finite element method is used with lowest-order Raviart-Thomas elements for flow and a higher-order discontinuous Galerkin method for the saturation equation.

4.2 Problem Formulation

We start with an overview of the basic framework that will be used throughout this paper. We outline the main characteristics of the reservoir and fracture models and the information exchanged between them.

4.2.1 Basic Framework

The reservoir and the fracture are modeled in two separate domains. The reservoir model is set in $\Omega^r \subset \mathbb{R}^d$ $(d = 2 \text{ or } 3)$. In addition to the regular external boundaries Γ^r, we impose internal boundaries Γ^f, representing the surface of the fracture. We denote the "top" and "bottom" sides of the fracture boundary by Γ^f_+ and Γ^f_- [Fig. 4.1a].

We set the fracture model in the domain $\Omega^f \subset \mathbb{R}^d$, as shown in [Fig. 4.1b]. Although the fracture domain is a (d-1)-dimensional surface in the reservoir, the fracture model itself is in a (d)-dimensional domain. The alternative would be to solve the fracture problem over a nonplanar (d-1)-dimensional manifold, which would require the use of specialized differential operators defined over such domains. While there are some benefits to such an approach, a (d)-dimensional problem is simpler to conceptualize and implement. In addition, it allows for refinement in the direction normal to the fracture surface.

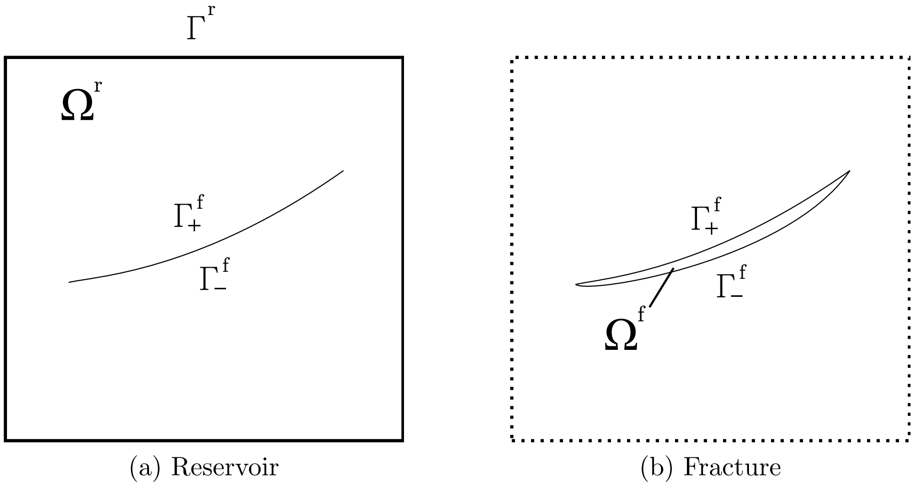
We denote the flow model in the reservoir domain as F_{res}, and the flow model in the fracture domain as F_{frac}. The fracture model F_{frac} specifies the appropriate boundary condition on $\Gamma^f_{\{+,-\}}$ to F_{res}. In turn, F_{res} calculates the jump in flux over the fracture surface, which serves as a leakage term for F_{frac}.

We identify three key components which must be chosen to define a particular system:

1. The flow models F_{res} and F_{frac};
2. The methods used to discretize F_{res} and F_{frac};
3. The solution approach to the resulting system.

Regarding the flow models F_{res} and F_{frac}, there are many possible choices. In this work we focus on two. The first model considers single-phase slightly compressible flow with Darcy law for fracture flow. Using Darcy law for flow in the fracture incorporates the commonly used cubic law flow model. The second one considers a two-phase slightly compressible flow model in both the fracture and the reservoir. We will more fully define the flow models in the following subsections.

Finally, we must choose a solution method. The two basic approaches are the fully coupled and the iteratively coupled methods. In the fully coupled approach, a single system of equations that satisfies both models, F_{res} and F_{frac}, is formed. Alternatively, the iteratively coupled approach solves for one model first, then passes an updated solution to the second, and so on. While the fully coupled approach can greatly simplify the problem, there are advantages to using iterative coupling. First, it allows for interfacing with existing legacy codes without modifying the internal Jacobian construction, and second, the overall time-step size is not dictated by the fracture time step size.



(a) Reservoir (b) Fracture

Figure 4.1: Define two separate domains with two separate models, one for the flow in the reservoir and one for flow in the fracture.

4.2.2 Single-Phase Slightly Compressible Flow

We first consider a single-phase slightly compressible flow problem wherein flow in the reservoir (F_{res}) is governed by the equations

$$\begin{aligned} \phi^r \frac{\partial \rho}{\partial t} - \nabla \cdot \left(\frac{\rho}{\mu} \mathbf{K}^r \nabla p^r \right) &= Q^r \quad \text{in } \Omega^r, \\ p^r &= p^f \quad \text{on } \Gamma^f_{\{+,-\}}, \\ \mathbf{K}^r \nabla p^r \cdot \mathbf{n} &= 0 \quad \text{on } \Gamma^r. \end{aligned} \tag{4.1}$$

Here, ϕ^r represents the porosity of the reservoir, ρ is the density of the fluid, $\mathbf{K}^r$ is the permeability, μ is the viscosity of the fluid, p^r represents pressure, and Q^r denotes the source term. We specify the initial conditions for the problem at $t = 0$ by $p^r(x, 0) = p^{r,0}$. A no-flow boundary condition is imposed for simplicity. For slightly compressible fluids, we can assume that density is an exponential function of pressure:

$$\rho(p) = \rho^{\text{ref}} \exp(1 + c(p - p^{\text{ref}})), \tag{4.2}$$

where ρ^{ref} is the reference density at the reference pressure p^{ref} and c is the fluid compressibility.

The fracture flow F_{frac} is governed by

$$\begin{aligned} \phi^f \frac{\partial \rho}{\partial t} - \nabla \cdot \left(\frac{\rho}{\mu} \mathbf{K}^f \nabla p^f \right) &= Q^f - Q^\ell \quad \text{in } \Omega^f, \\ \mathbf{K}^f \nabla p^f \cdot \mathbf{n} &= 0 \quad \text{on } \Gamma^f_{\{+,-\}}. \end{aligned} \tag{4.3}$$

The source and fracture leakage terms are represented by Q^f and Q^ℓ, respectively. The fracture boundaries are of Neumann type and are set to a no-flow condition. Since the fracture mesh is always a single cell in thickness, the the choice of representing the leakage as a forcing term is mathematically equivalent to using a Neumann boundary. However, from a computational point of view, the use of a forcing term for leakage allows the reservoir model to avoid adding extra degrees of freedom at the internal boundary. That is, the reservoir model only needs to solve for the difference in fluxes rather than individual fluxes over the fracture. The fracture permeability is denoted by $\mathbf{K}^f$. The fracture porosity $\phi^f = 1$. At $t = 0$ the initial condition is $p^f(x, 0) = p^{f,0}$. For a cubic law fluid, permeability is computed as a function of the fracture width w:

$$\mathbf{K}^f = \frac{w^2}{12}. \tag{4.4}$$

As previously discussed, the two models (4.1) and (4.3) are coupled using boundary conditions and the leakage term. The leakage to the fracture and from the fracture is then calculated using

$$Q^{\ell} = [\mathbf{K}^r \nabla p^r \cdot \mathbf{n}]_{\Gamma^f} := \mathbf{K}^r \nabla p^r \cdot \mathbf{n}^+ + \mathbf{K}^r \nabla p^r \cdot \mathbf{n}^-, \tag{4.5}$$

where n^+ and n^- are opposite unit vectors normal to the fracture surface. The pressure boundary condition for the reservoir model is set to the pressure of the fracture, averaged in the normal direction. Combining equations (4.1), (4.3), and (4.5) completes the description of the single-phase reservoir-fracture flow model.

4.2.3 Two-Phase Slightly Compressible Flow

Following the same procedure, we formulate the immiscible two-phase reservoir-fracture flow model. The equations for the reservoir model are

$$\begin{aligned}
v^r_\alpha &= -\frac{\rho_\alpha}{\mu_\alpha} k_{r\alpha} \mathbf{K}^r \nabla p^r_\alpha, \quad \text{in } \Omega^r, \\
\frac{\partial \phi \rho_\alpha S^r_\alpha}{\partial t} &= -\nabla \cdot v^r_\alpha + Q^r_\alpha, \quad \text{in } \Omega^r, \\
p^r_c &= p^r_o - p^r_w = 0, \\
s^r_w + s^r_o &= 1, \\
p^r_\alpha &= p^f_\alpha, \quad \text{on } \Gamma^f_{\{+,-\}}, \\
s^r_\alpha &= s^f_\alpha, \quad \text{on } \Gamma^f_{\{+,-\}}. \\
K^r \nabla u^r_\alpha \cdot \mathbf{n} &= 0, \quad \text{on } \Gamma^r
\end{aligned} \tag{4.6}$$

Here, α denotes the phase (w or o), S_α is the saturation of the phase, $k_{r\alpha}$ is the relative permeability, and p_c is the capillary pressure. The problem starts with initial conditions $p_\alpha^r(x,0) = p_\alpha^{r,0}$ and $s_\alpha^r(x,0) = s_\alpha^{r,0}$. The pressure and saturation at the fracture boundary are set to values averaged over the fracture width.

The fracture model F_{frac} is

$$\begin{aligned}
v_\alpha^f &= -\frac{\rho_\alpha}{\mu_\alpha} k_{r\alpha} \mathbf{K}^f \nabla p_\alpha^f, \quad \text{in } \Omega^f, \\
\frac{\partial \phi \rho_\alpha s_\alpha^f}{\partial t} &= -\nabla \cdot v_\alpha^f + Q_\alpha^f - Q_\alpha^\ell, \quad \text{in } \Omega^f, \\
K^f \nabla u_\alpha^f \cdot \mathbf{n} &= 0, \ \text{ on } \Gamma_{\{+,-\}}^f, \\
p_c^f &= p_o^f - p_w^f = 0, \\
s_w^f + s_o^f &= 1.
\end{aligned} \tag{4.7}$$

Similar to the single-phase problem, the two-phase system also includes a leakage term Q_α^ℓ. The leakage for each phase is computed in the reservoir problem as the jump in fluxes across $\Gamma_{\{+,-\}}^f$,

$$Q_\alpha^\ell = [v_\alpha^r \cdot \mathbf{n}]_{\Gamma^f} := v_\alpha^r \cdot \mathbf{n}^+ + v_\alpha^r \cdot \mathbf{n}^-. \tag{4.8}$$

4.3 Discretization

We now turn to the problem of discretizing the equations stated in the previous section. In this section, we describe the final forms of the discrete problems. Further details can be found in Appendix B, and in [62]. We will also discuss methods for solving the resulting systems of equations.

4.3.1 Single-Phase Slightly Compressible Flow

Using the MFMFE method defined in [62], the discrete form of the reservoir problem can be stated for given initial conditions $p_h^{r,0}$ and $v_h^{r,0}$ as follows:

Find $p_h^{r,n+1} \in W_h$, $v_h^{r,n+1} \in V_h$, such that

$$\begin{aligned}
\left(\frac{\mu}{\rho^{n+1}}(\mathbf{K}^r)^{-1} v_h^{r,n+1}, u_h\right) - (p_h^{r,n+1}, \nabla \cdot u_h) &= \\
&- \int_{\Gamma^f_{\{+,-\}}} p_h^{f,n+1} u_h \cdot \mathbf{n}, \quad \forall u_h \in V_h, \\
\left(\phi^r \frac{\rho(p_h^{r,n+1}) - \rho(p_h^{r,n})}{\Delta t}, w_h\right) + (\nabla \cdot v_h^{r,n+1}, w_h) &= \\
&(Q^{r,n+1}, w_h), \quad \forall w_h \in W_h.
\end{aligned} \tag{4.9}$$

The fracture model is discretized using the MFD method. The problem statement in this case is as follows:

For given initial conditions, $v_h^{f,0}$ and $p_h^{f,0}$, find $v_h^{f,n+1} \in X_h$ and $p_h^{f,n+1} \in Q_h$, such that

$$\begin{aligned}
[v_h^{f,n+1}, u_h]_{X_h} - [p_h^{f,n+1}, \mathcal{DIV}\, u_h]_{Q_h} &= 0, \quad \forall u_h \in X_h, \\
\left[\phi^f \frac{\rho(p_h^{f,n+1}) - \rho(p_h^{f,n})}{\Delta t}, q_h\right]_{Q_h} + [\mathcal{DIV}\, v_h^{f,n+1}, q_h]_{Q_h} &= \\
&[Q^{f,n+1} - Q^{\ell,n+1}, q_h]_{Q_h}, \quad \forall q_h \in Q_h.
\end{aligned} \tag{4.10}$$

There are two different ways to solve the equations in (4.9) and (4.10). One approach is to construct a single linear system for reservoir and fracture flow. This can be interpreted as applying the block Gauss-Seidel method to the coupled system. The other method is to form two separate linear systems for fracture and reservoir flow, and then solve them by iterative coupling. The code coupling was performed with Gurpreet Singh, Gergina Pencheva and Tameem Al-Mani, and our results were published in [2].

4.3.2 Two-Phase Slightly Compressible Flow

For the two-phase flow problem, we use MFD for both the reservoir and the fracture discretizations. A more elaborate description of discretizing two-phase flow using MFD can be found in Appendix B. Applying the construction shown in Appendix B to both the fracture and the reservoir problems gives us two saddle-point systems for the pressure equations:

$$\begin{pmatrix} M^r & -\mathcal{DIV}^{*\,r} \\ \mathcal{DIV}^{\,r} & C^r \end{pmatrix}, \tag{4.11}$$

$$\begin{pmatrix} M^f & -\mathcal{DIV}^{*\,f} \\ \mathcal{DIV}^{\,f} & C^f \end{pmatrix}. \tag{4.12}$$

Formulation of the problem as a saddle-point system keeps the derivation in a general form. Similar to MFMFE, we can perform a procedure to reduce the system to a cell-centered method with only pressure unknowns, as shown in [38]. We combine the equations (4.11) and (4.12), giving

$$\left(\begin{array}{cc|cc} M^r & -\mathcal{DIV}^{*\,r} & 0 & L \\ \mathcal{DIV}^{\,r} & C^r & 0 & 0 \\ \hline 0 & 0 & M^f & -\mathcal{DIV}^{*\,f} \\ -L^T & 0 & \mathcal{DIV}^{\,f} & C^f \end{array}\right). \tag{4.13}$$

Solving the above system results in a set of pressures and velocities for both the reservoir and the fracture problems, (v_t^r, p_o^r) and (v_t^f, p_o^f). We impose no-flow boundary conditions in the fracture model, as a result, fractional flow calculations in the fracture depend solely upon phase saturations and pressures in the fracture. For the reservoir problem, care must be taken when computing the fractional flow at the fracture boundaries.

After solving the pressure equation, the water velocities for the fracture and reservoir, v_w^f and v_w^r, respectively, are computed, followed by separate saturation updates for each model:

$$\begin{aligned}
&\left[\frac{\phi^r}{\Delta t}\left(\rho_w(p_w^{r,n+1})s_w^{r,n+1} - \rho_w(p_w^{r,n})s_w^{r,n}\right), q_h^r\right]_{Q_h^r} = \\
&\qquad [-\mathcal{DIV}\, v_w^r, q_h^r]_{Q_h^r} + [Q_w^r, q_h^r]_{Q_h^r}, \quad \forall q_h^r \in Q_h^r, \\
&\left[\frac{\phi^f}{\Delta t}\left(\rho_w(p_w^{f,n+1})s_w^{f,n+1} - \rho_w(p_w^{f,n})s_w^{f,n}\right), q_h^f\right]_{Q_h^f} = \\
&[-\mathcal{DIV}\, v_w^f, q_h^f]_{Q_h^f} + [Q_w^f - Q_w^\ell, q_h^f]_{Q_h^f}, \quad \forall q_h^f \in Q_h^f.
\end{aligned} \tag{4.14}$$

There is the option of following the standard IMPES technique for solving the system, or the iteratively coupled IMPES [40] may be used. In the iteratively coupled approach, after updating the saturation, another pressure solve is invoked with the latest saturation values. This is continued until the overall system converges.

Chapter 5

MFD Code in Python

In this chapter, we discuss details about the code developed to test the modification of MFD and the fracture flow method. The code is written in Python with some portions extended by C. The working name of the program is "mimpy", which is short for Mimetic Python. The code relies heavily on the NumPy and SciPy libraries [33, 47]. Mimpy has an object-oriented structure with two main classes: the **mesh** class and the **mfd** class. It remains relatively short—less than 10,000 lines—due to the general nature of MFD and the brevity of Python code. All the examples presented in this work are run using this code.

The **mesh** class provides a representation of the computational mesh. The class maintains the geometry of the mesh composed of points, faces and cells. Individual points in $\mathbb{R}^2$ or $\mathbb{R}^3$ are combined to form faces. These faces are then combined to form individual cells in the mesh. Along with the overall geometry, the **mesh** class also stores other properties of the mesh. For the faces, it maintains the following:

- normals;
- areas;

- centroids;
- x_e.

The point x_e refers to the point used in condition ($\tilde{\text{S}}2$). For cells, the following information is maintained:

- face orientation;
- volume;
- centroid;
- x_E.

The normal orientation indicates whether the face normal is pointing in or out of the cell. The cells also allow for the definition of a shifted point, x_E, which comes from condition ($\tilde{\text{S}}2$). In addition, the class stores cell properties, such as the permeability tensor. The **mesh** class also contains functions for performing integrals over faces and elements. These integrals are used in incorporating forcing functions and boundary conditions.

The **mfd** class constructs the saddle-point system based on the information from **mesh**. The matrix construction is based on satisfying the conditions (S1) and ($\tilde{\text{S}}2$) defined in Chapter 2. The class can construct standard MFD matrices as well diagonal matrices in the case of Voronoi diagrams.

The final part to the code are the model classes, which solve subsurface problems using the **mfd** and **mesh** classes. Two examples are the **singlephase** and **twophase** classes. They use the **mfd** linear system to solve the flow problem, and in the case of two-phase, use a simple single-point up-winding method for transport. The details of the geometry and discretization of the problem are abstracted by **mesh** and **mfd**, making the model files relatively simple to code. The result is that a two-phase code written in this framework will function automatically with hexahedra, tetrahedra and Voronoi meshes. In addition to saving time in development, this kind of code reusability is critical for proper code validation.

5.1 Boundaries

One of the main features of mimpy is its ability to handle boundary conditions in a simple and unique fashion. In addition to representing standard Neumann and Dirichlet boundary conditions from explicit functions, the code is able to define boundaries by referencing other elements or faces in the mesh. This feature, allows the code to easily include fractures, as well domain decomposition methods.

Boundary information is stored in the **mesh** class. The two basic boundary conditions are Dirichlet and Neumann boundaries. These are stored as collections of faces, with groups of faces indicated with a boundary marker. It is also possible to represent boundaries that are defined implicitly through other variables in the system. For these cases, the **mesh** class allows what are called "pointer" boundaries, where the boundary value is taken implicitly from another degree of freedom in the system. In the case of fractures, the internal

Dirichlet boundary in the reservoir points to the pressure of the appropriate fracture cell. In turn, the forcing function for the fracture is a function of the flux computed at the reservoir faces. This feature can also be used to test domain-decomposition methods using MFD. There have been some early results in this area with Tameem Al-Mani.

In this fashion, adding fractures and performing domain decomposition with code can be represented directly in the **mesh** class. This modification greatly reduces the number of lines of code and simplifies the overall structure of the program. This kind of programming structure allows the code to easily test various combinations of meshes, discretizations and models.

5.2 Associated Codes

Mimpy relies on the NumPy and SciPy libraries. The code uses sparse direct solvers and GMRES found in spsolve library of SciPy. For larger models, the code has been coupled with PETSc [10] using the petsc4py interface [23]. Plots are produced in the VTK format and can be viewed using the Paraview scientific visualizer. In addition, the program has functions to produce output in Gnuplot in the case of two-dimensional problems.

For cell volume and centroid computations, the code uses the algorithm defined in [43]. The implementation is a ported version of the C code published by the author in Python. The derived classes of **mesh** include a class that can read tetgen meshes for tetrahedral meshes. For Voronoi diagrams, the program reads output from the Voro++ package [51].

Chapter 6

Numerical Results

In this section we present numerical results for MFD and our proposed generalization. For the purposes of the experiments, we will use the following conventions:

1. The permeability tensor $\mathbf{K}_E$ is computed at the centroid of each cell E.

2. The pressure error is computed using,

   $$\|p - p_h\|_{L^2}^2 \approx (\text{L2error}(x))^2 = \sum_E |E|(p(x) - p_E)^2 \tag{6.1}$$

   The exact solution is calculated at a point x on the interior of E. That point is either C (the centroid of E) or V (the generating point for the Voronoi diagram).

3. The velocity error is computed using,

   $$\|v - v_h\|_{(L^2)^d}^2 \approx \sum_E |E| \sum_{e \in \partial E} (v(C) \cdot \mathbf{n} - v_e)^2 \tag{6.2}$$

   We approximate the exact velocity at the centroid C of each face e.

6.1 Two-Dimensional Convergence

We present results confirming first-order convergence of the modified MFD formulation for (2.1) over two-dimensional meshes. We consider the following manufactured solution from [16] over the unit square domain $\Omega = [0,1]^2$,

$$p(x,y,z) = x^3y^2 + x\sin(2\pi xy)\sin(2\pi y) + 1,$$

and a full permeability tensor

$$K = \begin{pmatrix} (x+1)^2 + y^2 & -xy \\ -xy & (x+1)^2 \end{pmatrix}.$$

We solve the problem first over a rectangular mesh, and then over Voronoi meshes. For each kind of mesh, four different cases are presented.

Case 1 x_E is the centroid of cell E and x_e is the centroid of face e.

Case 2 x_E is shifted to the point-centered location (V_E) and x_e is the centroid of face e.

case 3 x_E is the centroid of E and x_e is shifted to the point-centered location (b_e).

Case 4 Both x_E and x_e are shifted to the point-centered locations.

These four cases are illustrated in [Fig. 6.1].

The matrix U_E is chosen to be

$$U_E = \frac{|E|}{\text{trace}(K_E)}\mathbb{I}.$$

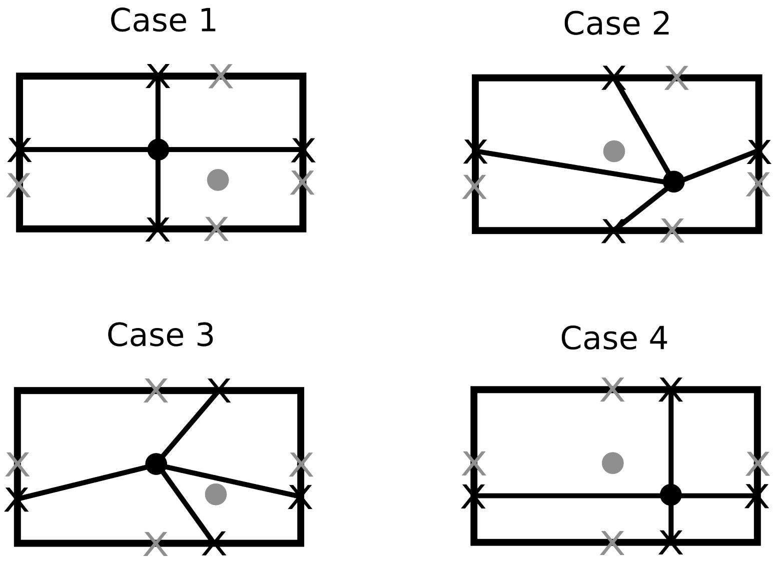


Figure 6.1: The four cases

6.1.1 Rectangular Grids

We carry out the results on the meshes in [Fig. 6.2]. These meshes are generated by first placing points evenly spaced by h. The points are then perturbed as

$$\xi_l = lh + \frac{3}{50}|\sin(4\pi lh)|, \ \ \xi = x, y. \tag{6.3}$$

The mesh is then constructed using these points in a point-centered fashion. That is, the faces are placed midway between two adjacent points. These points serve as the shifted location V_E for point x_E.

The results of these experiments can be seen in Table 6.1. Notice that in all cases we have established at least first-order convergence, and often the method exhibits clear second-order rates. Note that in cases 2 and 4, the

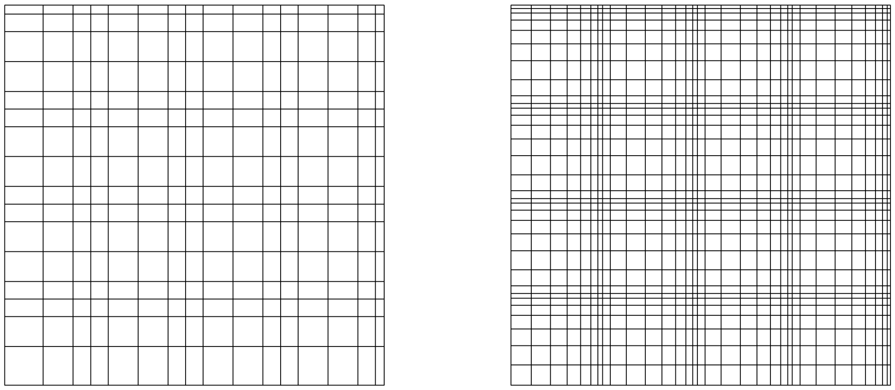


Figure 6.2: The rectangular meshes used in the convergence study.

solution seems to be better at the shifted point rather than at the centroid of the problem. This is consistent with the notion that the solution is most accurate at point x_E.

6.1.2 Voronoi Grids

We test the method using randomly generated Voronoi diagrams. The diagram is constructed by selecting uniformly distributed random points in the domain as generating points for the Voronoi diagram. The mesh is refined by selecting a larger number of randomly generated points unrelated to the previous mesh. Examples of the meshes used can be seen in [Fig. 6.4]. An example of two of the pressure solutions along with the analytical solution can be seen in [Fig. 6.5].

We tested the four different cases, and the results can been seen in Table 6.2. Notice how second-order convergence occurs at the location of

(Case 1—centroid/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	1.129e-02	2.728e-02	6.270e-01
32	3.217e-03	8.850e-03	2.068e-01
64	8.617e-04	2.766e-03	6.191e-02
128	2.224e-04	8.906e-04	1.753e-02
Conv.	1.89	1.65	1.72

(Case 2—point/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	2.439e-02	1.384e-02	9.220e-01
32	7.948e-03	3.638e-03	2.728e-01
64	2.558e-03	9.381e-04	7.654e-02
128	8.491e-04	2.409e-04	2.110e-02
Conv.	1.62	1.95	1.82

(Case 3—centroid/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	1.358e-02	2.899e-02	9.206e-01
32	4.154e-03	9.510e-03	3.177e-01
64	1.134e-03	2.931e-03	9.395e-02
128	2.938e-04	9.256e-04	2.719e-02
Conv.	1.85	1.66	1.70

(Case 4—point/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	2.352e-02	1.272e-02	8.712e-01
32	7.501e-03	3.184e-03	2.844e-01
64	2.438e-03	7.908e-04	8.754e-02
128	8.241e-04	2.008e-04	2.716e-02
Conv.	1.61	1.99	1.67

Table 6.1: Convergence errors and rates for the four test cases with rectangular grids over two-dimensional domains.

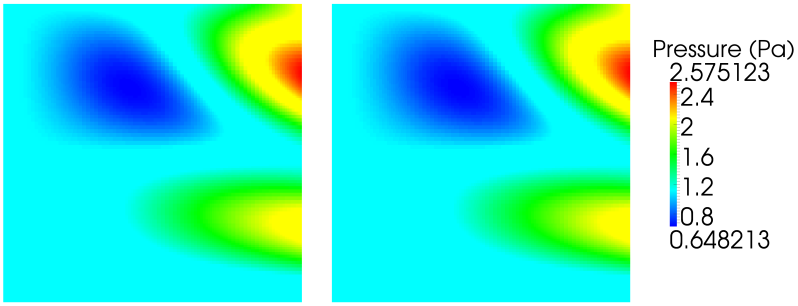


Figure 6.3: The MFD pressure solution (left) and the analytic pressure solution (right) for using rectangular meshes.

point x_E. Shifting of the point on the interior changes where the solution is most accurate. As predicted by the theory, we consistently maintain at least first-order convergence at the centroid of the cell regardless of shifting x_E and x_e. Since we have full-tensor, anisotropic permeability tensor K, we cannot reduce this system to a two-point flux approximation.

6.2 Three-Dimensional Convergence

We now shift our attention to problem in three-dimensions We will consider the following manufactured solution over the unit cube domain $\Omega = [0,1]^3$,

$$p(x,y,z) = x^3y^2z + x\sin(2\pi xy)\sin(2\pi yz)\sin(2\pi z) + 1,$$

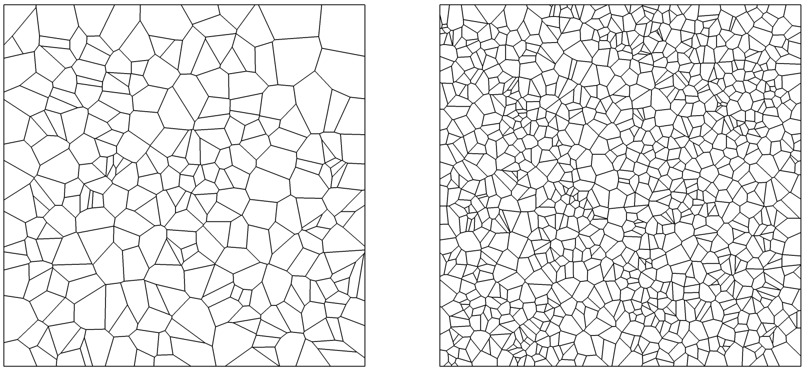


Figure 6.4: The random Voronoi meshes used in the convergence study.

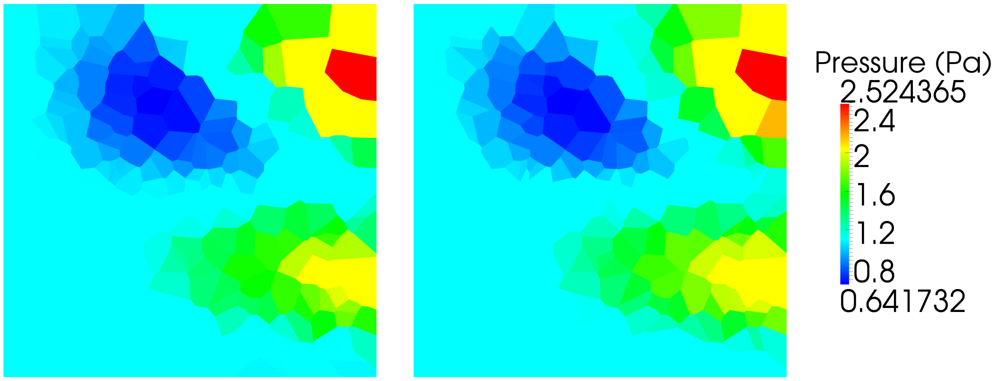


Figure 6.5: The MFD pressure solution (left) and the analytic pressure solution computed at the cell centroids (right) using two-dimensional Voronoi diagrams.

(Case 1—centroid/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	1.827e-02	5.002e-02	2.076e+00
32	4.873e-03	1.729e-02	9.892e-01
64	1.202e-03	8.616e-03	5.072e-01
128	2.858e-04	4.286e-03	2.489e-01
Conv.	2.00	1.16	1.01

(Case 2—point/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	4.435e-02	2.824e-02	2.515e+00
32	1.762e-02	8.682e-03	1.126e+00
64	8.767e-03	1.863e-03	5.060e-01
128	4.307e-03	4.633e-04	2.410e-01
Conv.	1.11	2.00	1.13

(Case 3—centroid/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	2.964e-02	5.885e-02	2.985e+00
32	5.643e-03	1.775e-02	1.244e+00
64	1.567e-03	8.739e-03	6.038e-01
128	3.480e-04	4.293e-03	2.968e-01
Conv.	2.11	1.23	1.10

(Case 4—point/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
16	4.139e-02	2.757e-02	2.771e+00
32	1.701e-02	7.423e-03	1.201e+00
64	8.677e-03	1.682e-03	5.776e-01
128	4.296e-03	3.708e-04	2.831e-01
Conv.	1.08	2.08	1.09

Table 6.2: Convergence errors and rates for the four test cases with random Voronoi grids. Notice how second-order convergence of pressure always occurs at the shifted point x_E.

and a full permeability tensor

$$K = \begin{pmatrix} 1+y^2+z^2 & -xy & -xz \\ -xy & 1+x^2+z^2 & -yz \\ -xz & -yz & 1+x^2+y^2 \end{pmatrix}.$$

We solve the problem first over a rectangular mesh, and then over Voronoi meshes. We follow the same four test cases outlined in the previous section. The matrix U_E is chosen to be

$$U_E = \frac{|E|}{\mathrm{trace}(K_E)}\mathbb{I}.$$

6.2.1 Rectangular Grids

We carry out the results on the meshes in [Fig. 6.6]. These meshes are generated by first placing nodes evenly spaced by h. The nodes are then perturbed as

$$\xi_l = lh + \frac{3}{50}|\sin(4\pi lh)|, \quad \xi = x, y, z. \tag{6.4}$$

The mesh is then constructed using these points in a point-centered fashion. That is, the faces are placed midway between two adjacent points. These points serve as the shifted location V_E for point x_E. The results of these experiments can be seen in Table 6.3. Examples of the pressure solution can been seen in [Fig. 6.7]. Much like the two-dimensional problem, all cases we have established at least first-order convergence, and often the method exhibits clear second-order rates. We also notice again a slightly better pressure solution at the location of point x_E.

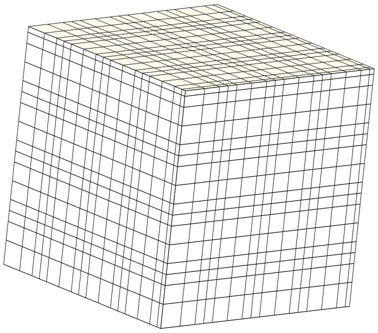


Figure 6.6: The rectangular meshes used in the convergence study. The two meshes correspond to h = $\frac{1}{16}$, and $\frac{1}{32}$.

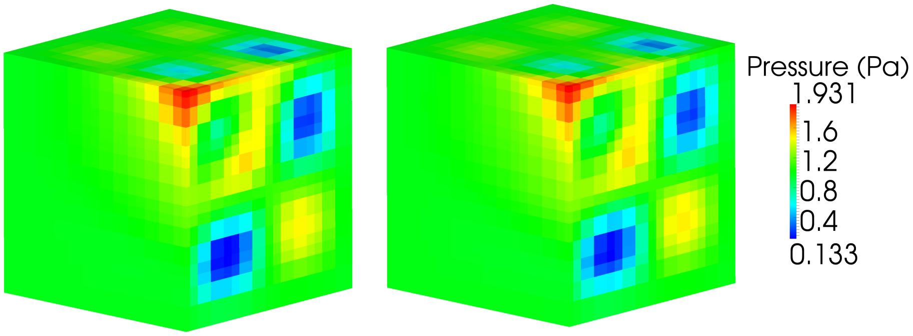


Figure 6.7: Comparison between the MFD solution (left) and the exact solution computed at the cell centroids (right) over a three dimensional domain with full permeability tensor.

(Case 1—centroid/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	1.585e-02	3.573e-02	6.218e-01
16	5.778e-03	1.863e-02	2.434e-01
32	1.699e-03	6.323e-03	7.597e-02
64	4.558e-04	2.055e-03	2.146e-02
Conv.	1.71	1.39	1.62

(Case 2—point/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	3.149e-02	2.828e-02	1.088e+00
16	1.961e-02	9.685e-03	5.552e-01
32	6.978e-03	3.526e-03	1.877e-01
64	2.240e-03	1.020e-03	5.653e-02
Conv.	1.29	1.58	1.44

(Case 3—centroid/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	1.714e-02	3.916e-02	1.049e+00
16	9.344e-03	2.227e-02	5.428e-01
32	3.062e-03	7.423e-03	1.899e-01
64	8.494e-04	2.310e-03	5.898e-02
Conv.	1.46	1.38	1.40

(Case 4—point/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	2.694e-02	2.439e-02	1.052e+00
16	1.686e-02	7.186e-03	4.797e-01
32	5.824e-03	2.080e-03	1.554e-01
64	1.943e-03	5.532e-04	4.880e-02
Conv.	1.29	1.81	1.49

Table 6.3: Convergence errors and rates for the four test cases with rectangular grids in three-dimensions. We consistently find at least first-order convergence for both pressure and velocity. Also notice that the pressure solution is most accurate when the method and error is calculated using the x_E point.

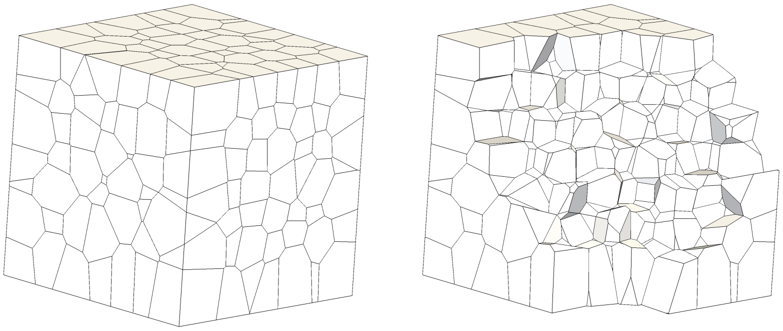


Figure 6.8: Unstructured three dimensional Voronoi diagrams. The plot on the left is the full domain of the convergence test problem in which we can only see the outer faces of the boundary. The plot on the right is a slice of the mesh showing the inner structure of the diagram.

6.2.2 Voronoi Grids

We test the method over three-dimensional Voronoi meshes. The meshes are generated by randomly selecting points uniformly over the domain. The Voronoi diagrams were generated using the Voro++ software package [51]. An example of the produced meshes can be seen in [Fig. 6.8]. The pressure solution can been seen in [Fig. 6.9].

The convergence results are shown in Table 6.4. Much like the two-dimensional case, we find that the pressure solution is most accurate at the location of the point x_E. Since the three-dimensional case is far more computationally intensive than the two-dimensional case, we were not able to carry out the convergence study as far.

(Case 1—centroid/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	5.432e-02	6.791e-02	4.018e+00
16	1.842e-02	2.569e-02	1.943e+00
32	5.501e-03	1.033e-02	8.775e-01
Conv.	1.65	1.36	1.10

(Case 2—point/centroid)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	8.660e-02	7.617e-02	4.500e+00
16	3.116e-02	2.507e-02	2.039e+00
32	1.119e-02	6.815e-03	8.852e-01
Conv.	1.47	1.74	1.17

(Case 3—centroid/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	5.746e-02	7.277e-02	4.482e+00
16	1.982e-02	2.768e-02	2.197e+00
32	5.951e-03	1.070e-02	9.843e-01
Conv.	1.63	1.3825	1.10

(Case 4—point/point)

n	L2error(C)	L2error(V)	$\|v^I - v_h\|_{(L^2)^d}$
8	7.714e-02	6.463e-02	4.443e+00
16	2.686e-02	2.059e-02	2.109e+00
32	1.057e-02	5.898e-03	9.307e-01
Conv.	1.43	1.72	1.13

Table 6.4: Convergence errors and rates for the four test cases with random Voronoi meshes. We consistently find at least first-order convergence for both pressure and velocity. Also notice that the pressure solution is most accurate when the error is calculated at the x_E point.

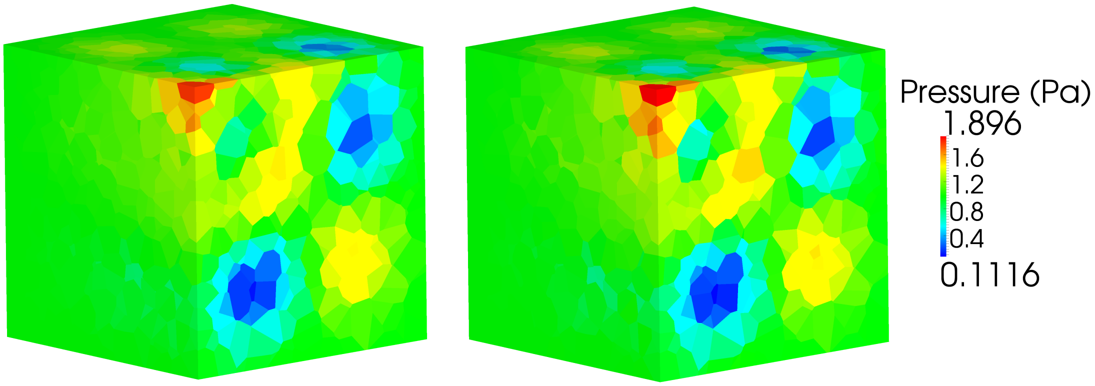


Figure 6.9: Pressure solution for three dimensional Voronoi diagram. The plot on the left represents the MFD approximation to pressure, and the plot on the right is the analytical solution computed at the cell centroids.

6.3 Two-phase Problems

In this section, we use MFD and the developed code for solving nonlinear, two-phase flows. For two-phase examples, se refers to the normalized water saturation,

$$se = \frac{s_w - s_{rw}}{1 - s_{rw} - s_{ro}}$$

6.3.1 Low Permeability Barrier

We demonstrate the use of Voronoi diagrams for modeling a low permeability barrier problem. We test two-phase flow around three different orientations of the barrier as seen in [Fig. 6.10]. The barrier has a constant permeability of 10^{-12}m^2 (10^3md) while the rest of the reservoir is set to 10^{-6}m^2 (10^9md). The Voronoi diagram can capture the geometry of the barrier [Fig. 6.11].

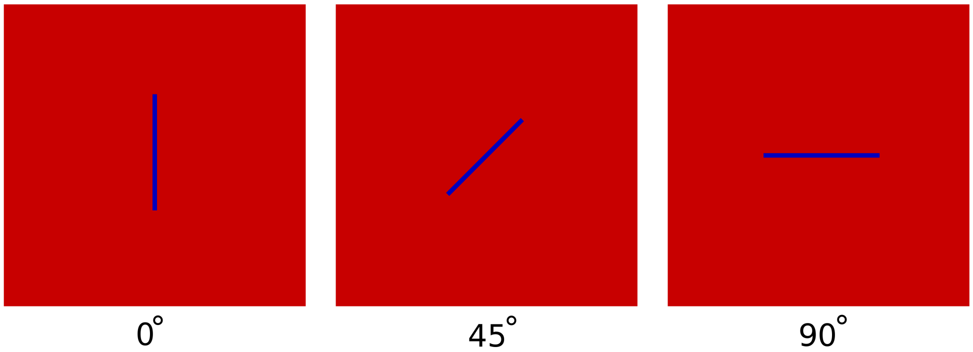


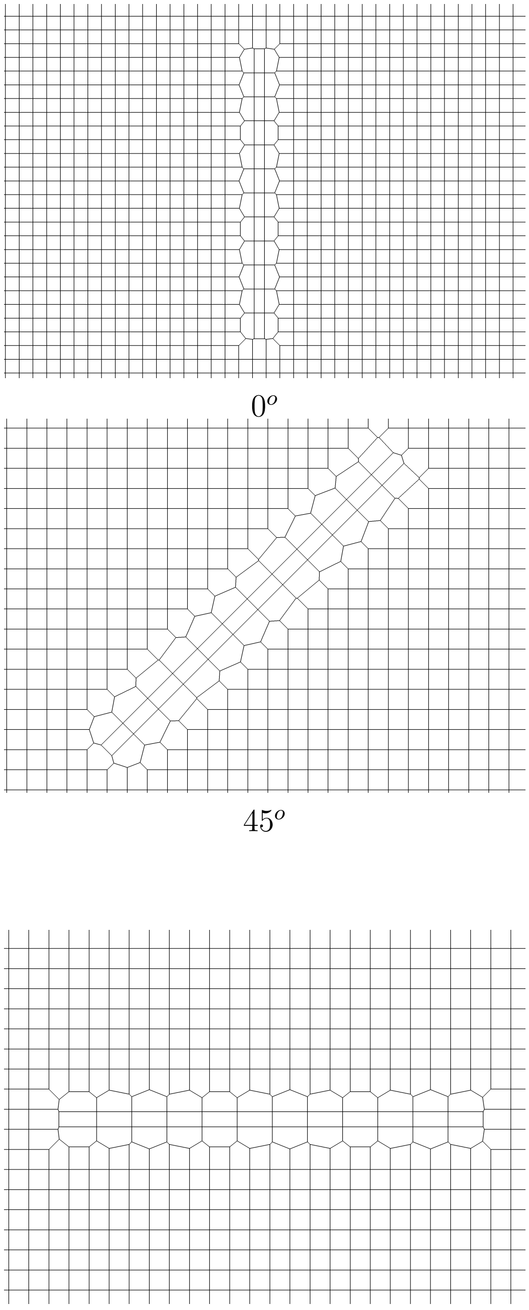
Figure 6.10: In the low permeability barrier example, three different orientations of the barrier are generated and tested. The blue streak corresponds to the area in the reservoir with low permeability. The geometry of the barrier is captured using a Voronoi mesh.

The relative permeabilities are set to linear functions

$$k_{rw} = se, k_{ro} = 1 - se.$$

Both water and oil are slightly compressible, with a compressibility of 10^{-8}Pa^{-1}. The boundary conditions are homogeneous Dirichlet ($p = 0$ Pa). We have a rate specified well injecting water at a constant rate of 1 kg/s. The porosity $\phi = .3$ is constant, as well as the initial saturation of water ($s_w^0 = 0$). We exclude the effects of capillary pressure ($p_c = 0$).

The oil pressure is shown in [Fig. 6.12]. The saturation at different times can be seen in [Fig. 6.13]. Notice that the method is able to capture complex behavior around the barrier while keeping most of the mesh rectangular. The only deviation from a rectangular mesh are localized to the area near the



0^o

45^o

90^o

Figure 6.11: The Voronoi mesh can capture the shape of the barrier while maintaining a rectangular mesh in the rest of the domain.

barrier.

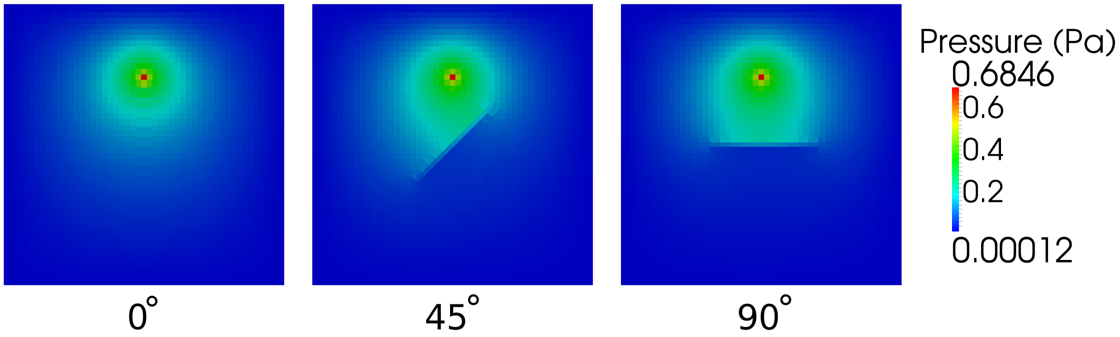


Figure 6.12: Pressure solution to the low-permeability barrier problem with different barrier orientations.

6.3.2 SPE10

We tested the use of MFD with a more heterogeneous example. We use the 85th layer of the SPE10 [19] benchmark problem for the permeability distribution [Fig. 6.14]. We place a rate specified injector in the south-east corner and a pressure specified well in the north-west corner. Here are the

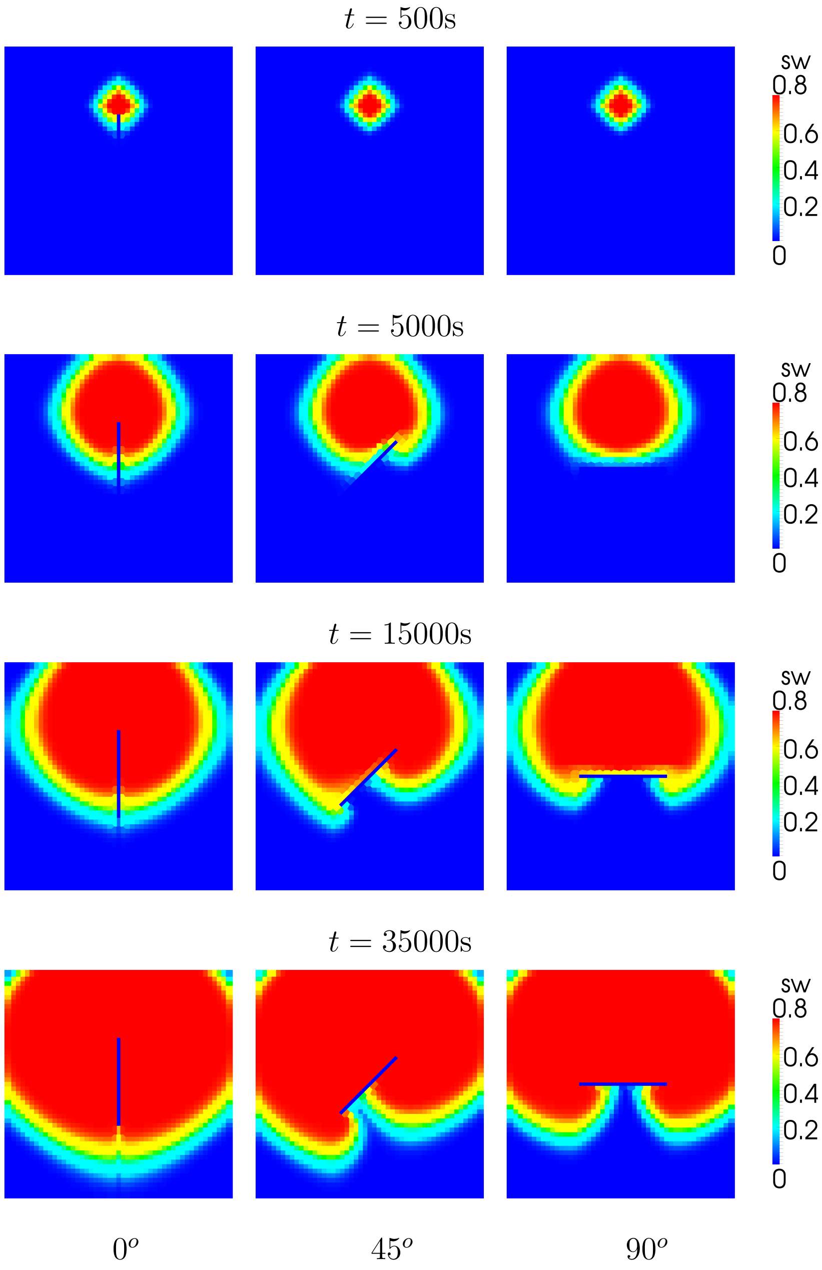


Figure 6.13: The saturation solution at different times for different barrier orientations.

overall properties of the system:

$$\begin{aligned}
\text{porosity} &= 0.3, \\
\text{initial water saturation} &= 0, \\
\text{water viscosity} &= 8.9 \times 10^{-4}\text{Pa s}, \\
\text{oil viscosity} &= 8.9 \times 10^{-4}\text{Pa s}, \\
\text{water density} &= 1000\text{kg/m}^3, \\
\text{oil density} &= 1000\text{kg/m}^3, \\
\text{residual water saturation} &= 0.2, \\
\text{residual oil saturation} &= 0, \\
k_{rw} &= se^2, \\
k_{ow} &= (1 - se)^2, \\
p_c &= 0.
\end{aligned}$$

Water is injected at a rate of 0.1 kg/s and simulator runs for 2315 days with 0.23 day time steps. The purpose of the test is to verify the feasibility of MFD for heterogeneous flow problems. The saturation solution is shown in [Fig. 6.15].

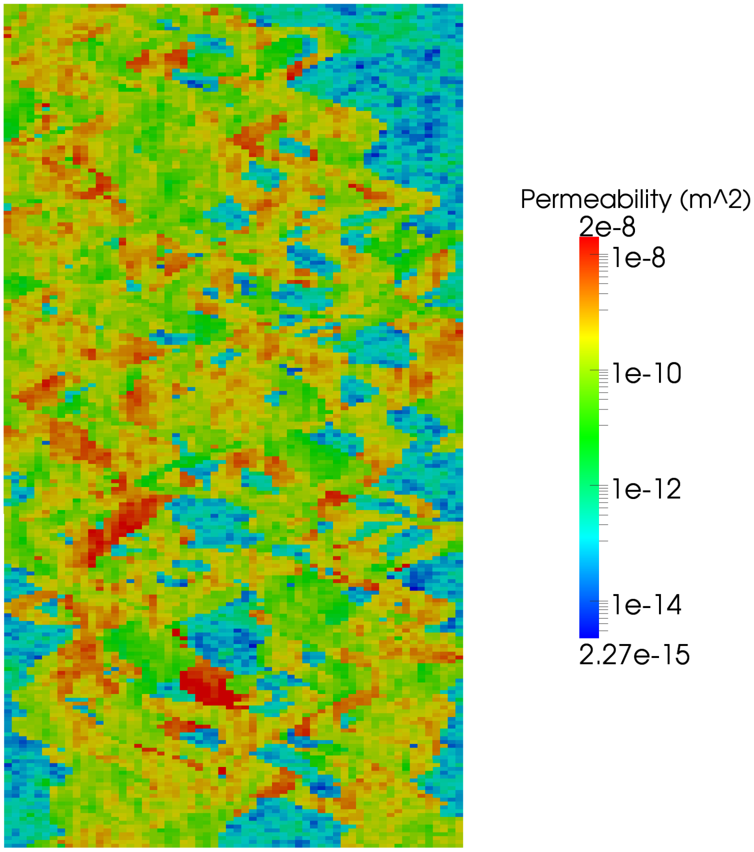


Figure 6.14: The log permeability of the 85th layer of SPE 10. The bottom layers of the SPE 10 problem are characterized by channelized permeabilities. A rate specified injector is placed at the bottom-right corner, and a pressure specified producer is placed in the top-left corner of the domain.

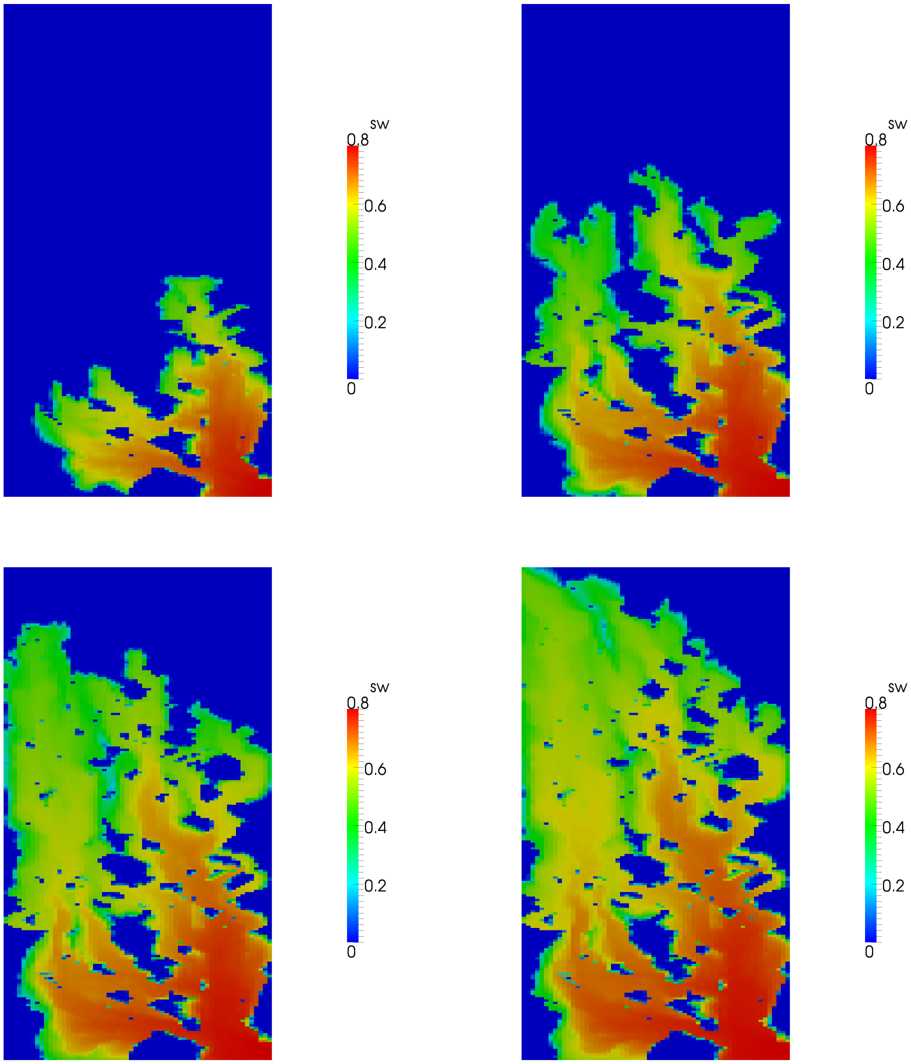


Figure 6.15: SPE 10 saturation profile after 579, 1158, 1737 and 2135 days. Notice the highly channelized nature of the saturation in the reservoir.

6.4 Fracture Problems

We demonstrate the use of the proposed MFD method for modeling fractures. The first example is a standard benchmark problem for hydraulic fractures that relies on a semi-analytic solution to a single-phase time dependent problem. We also show results from coupling the MFD code with the IPARS simulator. In the third section, we demonstrate the effects of fracture curvature on leakage around the fracture. That is followed by results showing the effects of curvature on two-phase flows with fractures.

6.4.1 Cinco-Ley Benchmark

In this section, we verify the method for solving fracture problems against standard benchmarks adopted by the petroleum industry. In particular, the well testing community has established semi-analytic solutions to single-phase time dependent fracture problems. We use the solution by Cino-Ley and Meng [20]. The authors provide a semi-analytical form of the solution in the Laplace space. In order to compute the leakages for the fracture, we form a linear system of equations that solve in the Laplace variable s. The solution is converted into the time domain using the Gaver-Stehfest algorithm [56].

The properties of the model are

$$\begin{aligned}
K_f &= 10^{-8}\text{m}^2, \\
X_f &= 10\text{m}, \\
w &= 0.01\text{m}, \\
\mu &= 10^{-3}\text{Pa}\cdot\text{s}, \\
c_t &= 10^{-8}\text{Pa}^{-1}, \\
Q^f &= 6.2832\text{kg/s}.
\end{aligned}$$

We vary the reservoir permeability resulting in three different cases,

$$K_{res} = [1{\times}10^{-13},\ 5\times10^{-12},\ 5\times10^{-11}]\ \text{m}^2,$$

which correspond to the following three dimensionless conductivity factors

$$\frac{K_f w}{K_r X_f} = C_{fD} = [100, 5, 0.2].$$

The setup of the problem is shown in [Fig. 6.16]. The resulting pressure is shown in [Fig. 6.17]. Finally, the comparison between the Cinco-Ley calculated fluxes and the solution produced by MFD are shown in [Fig. 6.18]. The domain is 65 × 85m, with a fracture of length 10m in the center. The well is located at the left-most point of the fracture. The MFD method accurately tracks the predicted solution from the semi-analytic benchmark over time and for all the fracture conductivity factors tested.

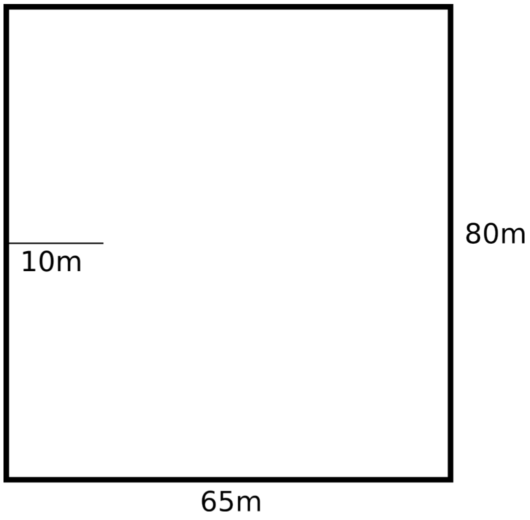
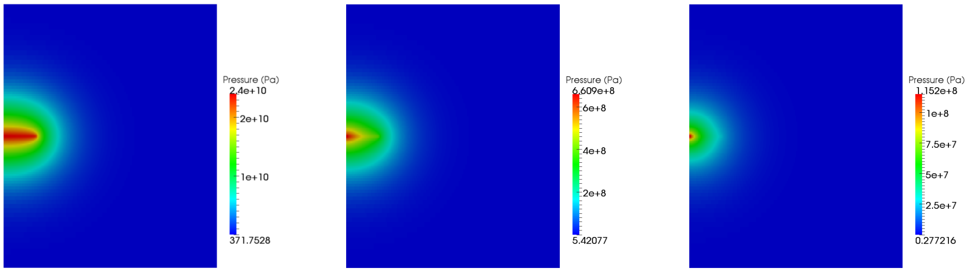


Figure 6.16: Problem setup for Cinco-Ley benchmark. The entire domain is 65×85m, with a fracture of length 10m in the center. The well is located at the left-most point of the fracture.



C_{fD}=100 C_{fD}=5 C_{fD}=.2

Figure 6.17: Pressure profiles with different fracture conductivities C_{fD}. Higher C_{fD} values correspond to a higher conductivity of the fracture. As the conductivity decreases, the pressure solution approaches a radial profile.

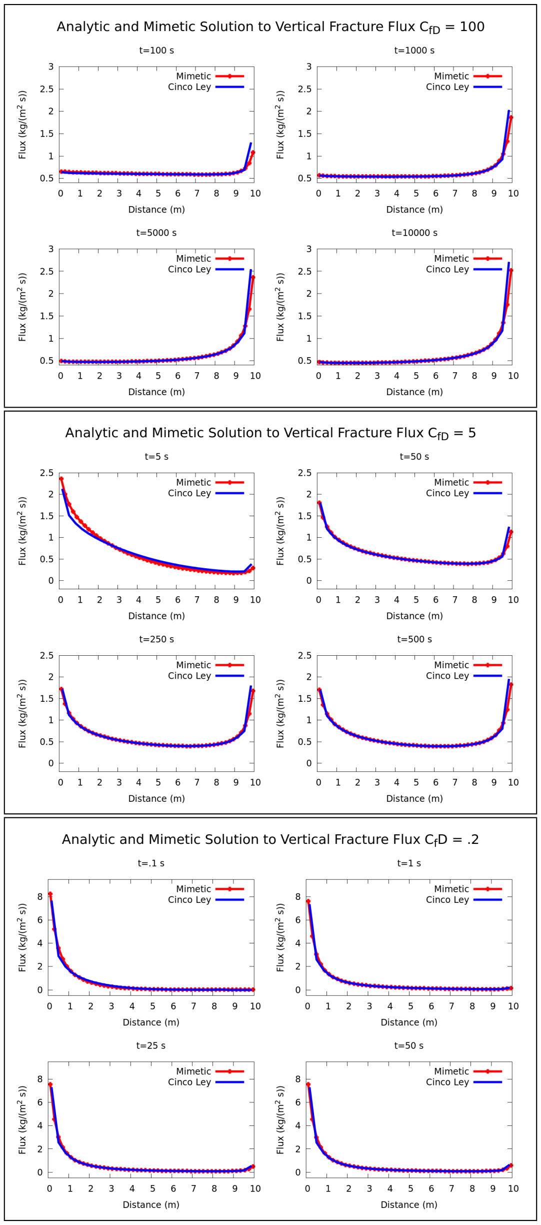


Figure 6.18: The leakage profile for the fracture problem with conductivity factors ($C_{fD} = [100, 5, .2]$) at different times. The blue line is the semi-analytic solution by Cinco-Ley and Meng, and the red line is the MFD solution. As the conductivity factor decreases, the leakage increases at the well, which is located at $x = 0$. The MFD fracture method is able to accurately capture the leakage for these different cases.

6.4.2 Single-Phase Slightly Compressible Flow with IPARS

The choice of iterative coupling scheme and flexibility of MFD and MFMFE inherently lends us the capability of handling multiple flow models simultaneously. Here, we discuss an example which considers coupling of hydraulic and discrete fracture flow models with a reservoir flow model. We solve the Darcy flow equation and the lubrication equation inside the discrete and hydraulic fractures, respectively, to demonstrate the multiphysics capability. In [Fig. 6.19] we see a reservoir with a hydraulic fracture (shown in green) connected to an injection well and a discrete fracture (gray).

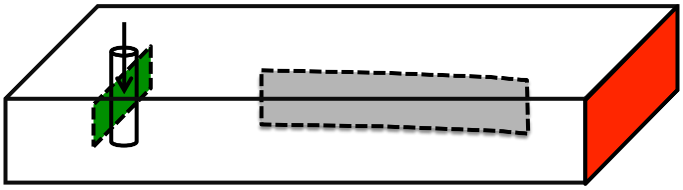
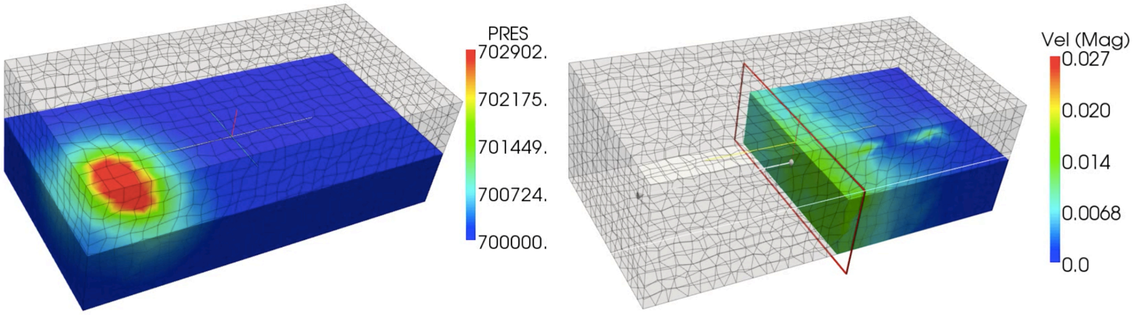


Figure 6.19: Reservoir description for MFD and IPARS coupling example.

The reservoir size is 10 m × 30 m × 15 m, with an injection rate of 10m^3/sec into the hydraulic fracture. The reservoir porosity is 0.3 and permeability is 10^{-10}m^2. The discrete fracture permeability is 10^{-6}m^2 and the hydraulic fracture width $w = 10^{-3}$m. Grid-block sizes are roughly 1m × 1m × 1m, owing to the distorted hexahedral cells. The reservoir has all no flow boundaries except one pressure specified boundary condition of approximately 1000 psi (shaded red) to expedite the simulation run. The choice of fracture flow models is not restrictive and more elaborate flow models can be used for accurate representation. The pressure field is shown in [Fig. 6.20].



(a) Pressure Field

(b) Velocity Field

Figure 6.20: Reservoir cross-section for the MFD and IPARS coupling example.

6.4.3 Fracture Curvature

In this section, we examine the effects of curvature on the flow around fractures. The problem setup is shown in [Fig. 6.21]. Fluid is injected directly into the fracture, and the amount of leakage from the fracture is measured. The meshes used are shown in [Fig. 6.22]. The curvature is increased by applying a quadratic perturbation to the mesh. We have a square domain with a Dirichlet (pressure) specified boundaries. The fracture is placed in the middle with rate specified well injecting fluid directly into the fracture. Five degrees of fracture curvature are tested. The fractures are shaped as quadratic polynomials, and are described using three points shown in Table 6.5. The problem is set over a square domain $\Omega = [0, 10]^2$.

The results of the simulation can be seen in [Fig. 6.23]. We observe a higher rate of leakage from the top of the fracture compared with the bottom of the fracture. The total percentage of leakage from the top for the five cases

	p_1	p_2	p_3
fracture 1	$(3, 5)$	$(5, 5)$	$(7, 5)$
fracture 2	$(2.95, 4.8)$	$(5, 4.9)$	$(7.05, 4.8)$
fracture 3	$(2.94, 4.77)$	$(5, 4.95)$	$(7.06, 4.77)$
fracture 4	$(2.94, 4.7)$	$(5, 4.98)$	$(7.06, 4.7)$
fracture 5	$(2.95, 4.59)$	$(5, 4.94)$	$(7.06, 4.59)$

Table 6.5: The five fractures are shaped as quadratic polynomials starting at p_1, ending at p_3 and pass through p_2.

are [50%, 51.23%, 52.7%, 53.92%, 54.83%]. This means that as we increase the curvature, a larger percentage of leakage is happening from the outer surface of the fracture.

This phenomena is entirely the result of the geometry of the fracture. As the fracture curves inward, a slightly larger pressure develops on the interior, resulting in less flow in that direction. Consider the extreme case, where the fracture curves all the way to form a closed circle. In that case, no fluid would leak into the interior of the circle, and 100% of the leakage would be toward the outside.

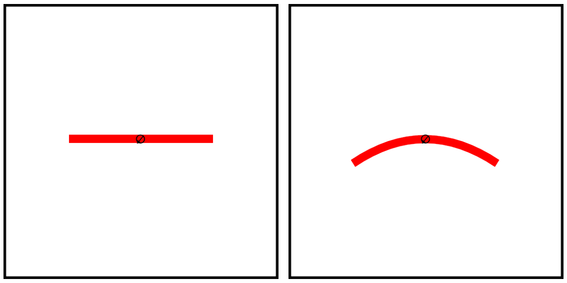


Figure 6.21: The problem setup for testing the effects of curvature on fracture flow. We have a square domain with a Dirichlet (pressure) specified boundaries. The fracture is placed in the middle with rate specified well injecting fluid directly into the fracture.

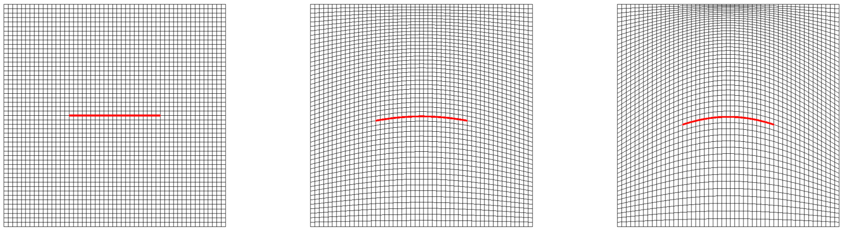


Figure 6.22: Three of the meshes used to test the different degrees of curvature. In the first mesh we have a planar fracture. The curvature is increased by applying a quadratic perturbation to the mesh.

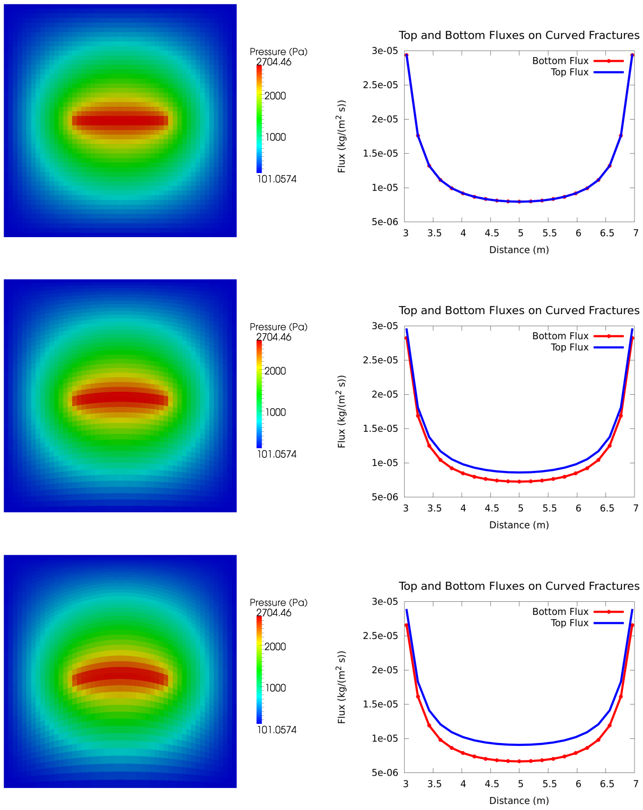


Figure 6.23: Effects of curvature on the leakage around a fracture. The blue line is the leakage from the top of the fracture, and the red line is the leakage from the bottom. As the fracture curvature is increased, we notice an increase in leakage from the top compared to the bottom.

6.4.4 Two-Phase Flow with Planar and Curved Fractures

We take a closer look at the effects of fracture curvature on two-phase flows. We have two different models, one with a flat fracture, and the second with a curved fracture [Fig. 6.24]. We set the boundaries to zero constant pressure. The injection rate in the fracture is $Q^f = 0.01$kg/s. The reservoir properties are

$$\begin{aligned}
\text{porosity} &= 0.3,\\
\text{initial water saturation} &= 0,\\
\text{water viscosity} &= 8.9 \times 10^{-4}\text{Pa s},\\
\text{oil viscosity} &= 2.67 \times 10^{-3}\text{Pa s},\\
\text{water density} &= 1000\text{kg/m}^3,\\
\text{oil density} &= 1000\text{kg/m}^3,\\
\text{residual water saturation} &= 0.2,\\
\text{residual oil saturation} &= 0,\\
k_{rw} &= se^2,\\
k_{ow} &= (1 - se)^2,\\
p_c &= 0.
\end{aligned}$$

The initial saturation $s_w^0 = 0$ in both the reservoir and the fracture. We took a time-step size of 10s. A close up of the curved fracture can been seen in [Fig. 6.26]. The pressure profile of the solution is shown in [Fig. 6.25]. In [Fig. 6.27] we can see the saturation around the fracture at early time. Notice the higher water saturation on the top surface of the fracture compared with the bottom. Note the small difference in saturation for the planar fracture case too, which is the result of a slight difference in cell volumes. For the planar case, top cell volume is 0.083 m^2 and bottom cell volume is 0.079 m^2. In the case of the curved fracture, top cell volume is .080 m^2 and the bottom cell volume .077 m^2.

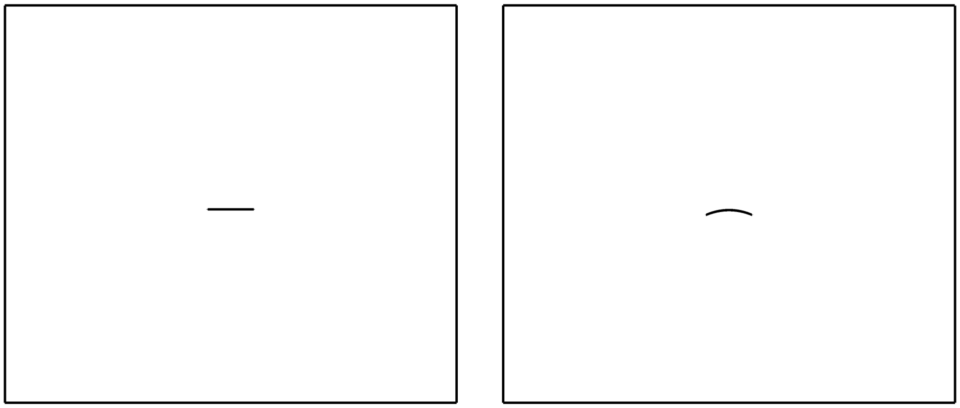


Figure 6.24: Outline of the two fractures used in the two-phase comparison.

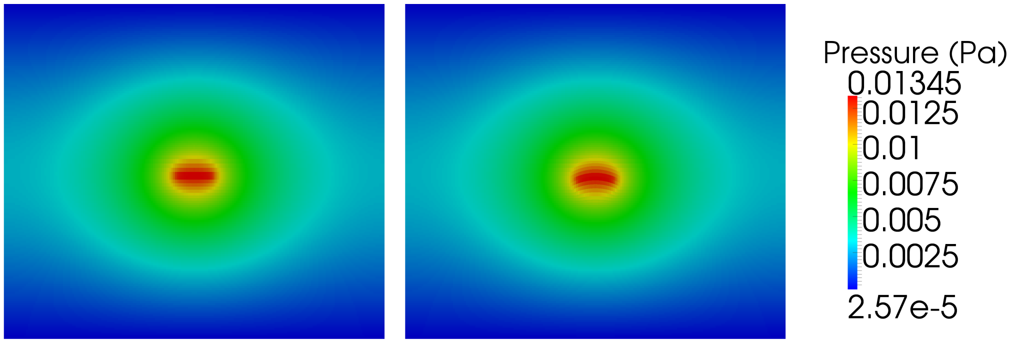


Figure 6.25: Pressure profile for the two-phase flow without curvature (left) and with curvature (right).

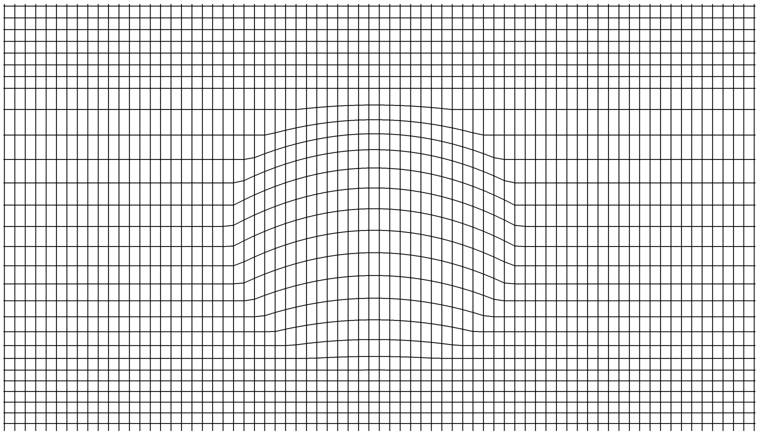


Figure 6.26: A close look at the reservoir mesh for the curved fracture case.

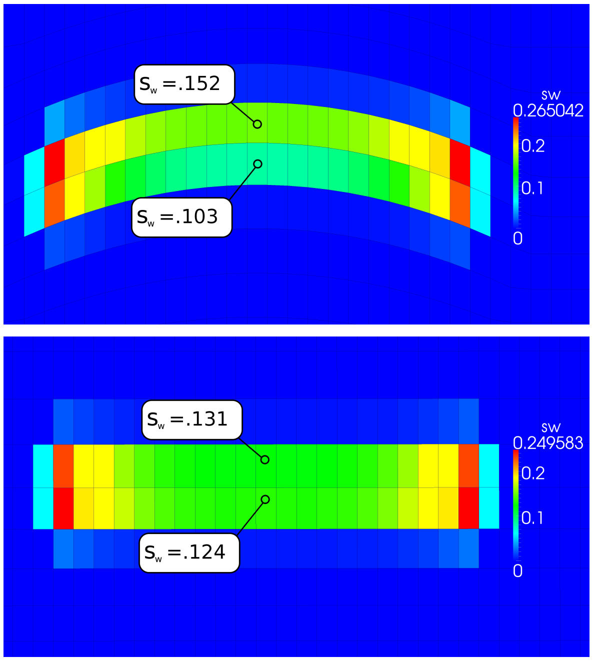


Figure 6.27: The saturation profile near the fracture at early time (0.23 days). Note the higher saturation on the top of the fracture compared to the bottom in the case of a curved fracture surface.

Chapter 7

Conclusions and Future Work

The Mimetic Finite Difference method (MFD) defines a family of discretizations on a very general set of polyhedral meshes. Solving over general polyhedra allows for better representation of complex subsurface features. Special care must be taken when solving for porous media multi-phase flow equations, as they can be sensitive to the kind of discretization used. Methods such as the two-point flux approximation (TPFA) have been well established for such equations. While the original MFD definition encompasses TPFA over rectangular grids, it does not include TPFA for cases such as Voronoi meshes. We have demonstrated the connection between MFD and TPFA over Voronoi grids by defining a generalization of the original MFD. Establishing this connection results in a reduction of the saddle-point system associated with MFD to a symmetric-positive definite system through a Schur's complement. In the case of rectangular grids, we have observed that the generalization connects MFD with both cell-centered and point-centered schemes.

We have presented a proof of stability and convergence of the generalization using tools from the mixed finite element method and standard MFD. The analysis suggests that alteration made to MFD maintains first-order convergence for both pressure and velocity unknowns. Numerical results corroborate these findings. In addition, they show super-convergence of pressure can still be achieved when the error is measured against a weighted norm.

We have successfully developed a framework for coupling flow in the reservoir with flow in the fracture using MFD. The approach is flexible and permits a variety of flow models, discretizations, and solution techniques. The approach allows for general geometries for both the fracture and the reservoir. The method and code have been verified against industry standard solutions from the well-testing literature.

Because MFD uses a similar framework to mixed finite elements and MPFA methods, we have been able to connect with the simulator IPARS. The results of the coupling are available in [2]. Further work in [28] have used this coupling, wherein the poro-elastic effects have been coupled with reservoir-fracture flow.

The geometry of fractures has an impact on the leakage of fluids from the fracture. We formulate models that demonstrate the asymmetry of leakage in cases of curved fractures. The effects of curvature are shown on both single-phase and two-phase flow problems.

7.1 Future Work

The MFD method is an infinite family of discretizations defined over a very general set of polyhedral meshes. We took that family and added an even broader set of possible discretizations. The generalization proposed has a simple mechanism for defining different approximations. This gives us a simple method of traversing a large portion of the space of convergent, low-order, conservative discretizations. What is the relationship between that space and other existing techniques available today? What useful methods can we find in that space?

Our original purpose of generalizing the MFD method is to establish the relationship between MFD and TPFA over Voronoi grids. It stands to reason that the generalization may have applications beyond saddle-point system reduction. As we have seen, shifting the points alters the location at which the solution is most accurate. Perhaps one can optimize for point locations against overall error, or in light of problem specific quantities of interest. For example, better monotonicity for two-phase flow, or improved well modeling. We also believe that the generalization has application to the area of multi-point flux approximations (MPFA). Much like we did with TPFA, what is the relationship between MFD and existing, well-established MPFA methods in the literature?

For fractures, we have tested problems with a single fracture in the media. Further work is needed to test effects of multiple fractures, especially in the case of fracture intersections. In addition, the current method requires a large reduction in time-step size. One approach would be to use ideas from domain decomposition to separate the fracture problem and the reservoir problem. MFD is an ideal method for such an approach since it can naturally handle non-matching grids.

Appendices

Appendix A

Surface and Nodal Approaches

A.1 Cell-Node Discretization

In an early example of the mimetic method, found in [54], the author defines a cell-centered discretization for an elliptic problem over a logically rectangular grid in two dimensions. The discrete scalar space is denoted by HC, and it represents an average pressure per cell. The discrete vector space is denoted by HN, and it represents a velocity vector at each node. Due to the logically rectangular nature of the mesh, each of these spaces can be indexed with an appropriate i, j coordinate. Each cell is associated with a single pressure uknown u_{ij}, and each node is associated with a velocity unkown (WX_{ij}, WY_{ij}) [Fig. A.1]. The variables x_{ij} and y_{ij} correspond to the physical coordinates of the nodes. The objective is to construct the discrete operators,

$$\mathcal{DIV} : HN \to HC, \tag{A.1}$$

$$\mathcal{G} : HC \to HN. \tag{A.2}$$

We define the inner products

$$(u, v)_{HC} = \sum_{ij} u_{ij} v_{ij} \Omega_{ij} + \text{Boundary Terms}, \tag{A.3}$$

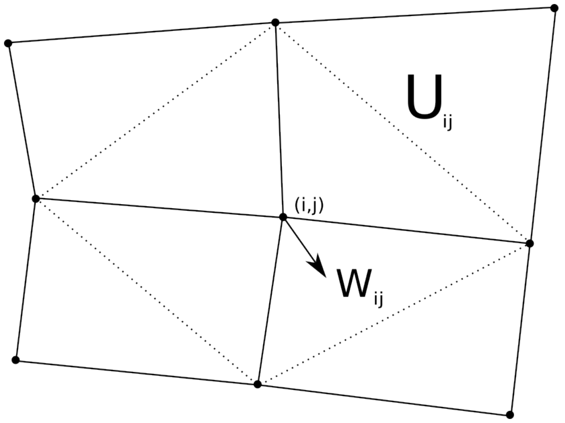


Figure A.1: Unknowns $U_{i,j}$ and $W_{i,j}$

and

$$(\vec{A}, \vec{B})_{HN} = \sum_{ij} (AX_{ij}BX_{ij} + AY_{ij}BY_{ij})VN_{ij}. \tag{A.4}$$

The quantity VN_{ij} is defined as the area of the diamond outlined by the dotted line in [Fig. A.1]. Note how the vector inner product centers around the nodes of the mesh.

Following the steps of MFD, we next define a discrete form of one of the differential operators. Recall the invariant definition of the continuous $\nabla\cdot$ operator,

$$\nabla \cdot\, v = \lim_{V \to 0} \frac{\int_{\partial V} (v, n) dS}{V}. \tag{A.5}$$

Using the trapezoid rule, the above line integral can be approximated for the discrete velocity space, giving the following expression for $\mathcal{DIV}$ at each cell:

$$
\begin{aligned}
(\mathcal{DIV}\,\vec{W})_{ij} = (&(WX_{i+1,j+1} - WX_{ij})(y_{i,j+1} - y_{i+1,j}) - \\
&(WX_{i,j+1} - WX_{i+1,j})(y_{i+1,j+1} - y_{i,j}))/(2\Omega_{ij}) - \\
&((WY_{i+1,j+1} - WY_{ij})(x_{i,j+1} - x_{i+1,j}) - \\
&(WY_{i,j+1} - WY_{i+1,j})(x_{i+1,j+1} - x_{i,j}))/(2\Omega_{ij}).
\end{aligned}
$$

The next step is to define our second operator based on the adjoint relationship, for all $u \in HN$ and $\vec{W} \in HC$,

$$(\mathcal{G}\,u, \vec{W})_{HC} = (u, \mathcal{DIV}\,\vec{W})_{HN}. \tag{A.6}$$

After some algebraic manipulation, an expression for the $\mathcal{G}$ operator is computed from the adjoint relationship with $\mathcal{DIV}$,

$$\mathcal{G}\,u = \begin{pmatrix} GX \\ GY \end{pmatrix}. \tag{A.7}$$

Where GX and GY are given by:

$$
\begin{aligned}
GX_{ij} = (&\frac{y_{i,j+1} - y_{i+1,j}}{2} u_{ij} + \frac{y_{i-1,j} - y_{i,j+1}}{2} u_{i-1,j} + \\
&\frac{y_{i,j-1} - y_{i-1,j}}{2} u_{i-1,j-1} + \frac{y_{i+1,j} - y_{i,j-1}}{2} u_{ij})/VN_{ij},
\end{aligned}
$$

$$
\begin{aligned}
GY_{ij} = (&\frac{x_{i,j+1} - x_{i+1,j}}{2} u_{ij} + \frac{x_{i-1,j} - x_{i,j+1}}{2} u_{i-1,j} + \\
&\frac{x_{i,j-1} - x_{i-1,j}}{2} u_{i-1,j-1} + \frac{x_{i+1,j} - x_{i,j-1}}{2} u_{ij})/VN_{ij}.
\end{aligned}
$$

By applying both operators, we can construct the left-hand side of the problem,

$$\mathcal{DIV}\,\mathcal{G}\,u = f. \tag{A.8}$$

The cell-node approach does result in a local stencil with the pressures as the only unknowns. Constructing a discretization that has this property is one of the goals of this work. However, the cell-node approach presents a problem that prevents us from using it for porous media applications. The problem is easily seen when applying the method to a mesh made of square elements. The resulting stencil has a checkerboard pattern, separating the mesh into two non-communicating sets. This produces spurious oscillations in the final solution [30].

A.2 Cell-Surface Discretization

We now outline what is referred to as the cell-surface method [30]. In contrast to the cell-node method, the cell-surface discretization defines velocities as the average magnitude of the normal component over each face in the mesh. In addition, the cell-surface method includes the permeability tensor in the velocity inner product,

$$(\vec{A}, \vec{B})_{HC} = \int_{\Omega} (\mathbf{K}^{-1}\vec{A}, \vec{B}) \; dV. \tag{A.9}$$

Instead of solving for $\nabla\cdot = -\nabla^*$, the permeability operator is combined with the gradient, resulting in the adjoint relationship

$$\nabla\cdot = (-\mathbf{K}\nabla)^*. \tag{A.10}$$

The scalar inner product is the same as the one defined for the cell-node discretization. The definition of the vector inner product becomes

$$(\vec{A}, \vec{B})_{HS} = \sum_i \sum_j (\mathbf{K}\vec{A}, \vec{B})_{i,j} VC_{ij}, \tag{A.11}$$

with VC_{ij} denoting the volume of the cell. The local inner products are based on a generalization of the inner product for non-orthogonal axes. For $\vec{A} = (AX, AY)$ and $\vec{B} = (BX, BY)$ and θ being the angle between the two polygon sides, we have

$$(\vec{A}, \vec{B}) = \frac{AXBX + AYBY + (AXBY + AYBX)\cos(\theta)}{\sin^2(\theta)}. \tag{A.12}$$

Including the permeability tensor gives

$$(\mathbf{K}\vec{A}, \vec{B}) = \frac{T11AXBX + T22AYBY + T12(AXBY + AYBX)\cos(\theta)}{\sin^2(\theta)}. \tag{A.13}$$

With,

$$\begin{aligned}
T11 &= K_{xx}\cos^2(\theta_1) + 2K_{xy}\cos(\theta_1)\sin(\theta_1) + K_{yy}\sin^2(\theta_1) \\
T12 &= K_{xx}\cos(\theta_1)\cos(\theta_2) \\
&\quad + 2K_{xy}(\cos(\theta_1)\sin(\theta_2) + \sin(\theta_1)\cos(\theta_2)) \\
&\quad + K_{yy}\sin(\theta_1)\sin(\theta_2) \\
T22 &= K_{xx}\cos^2(\theta_2) + 2K_{xy}\cos(\theta_2)\sin(\theta_2) + K_{yy}\sin^2(\theta_2).
\end{aligned}$$

The inner-product for each cell becomes

$$
\begin{aligned}
(\mathbf{K}\vec{A},\vec{B})_{i,j} = & \sum_{k,l}^{1} \frac{VN^{i,j}_{i+k,j+l}}{\sin^2(\theta^{i,j}_{i+k,j+l})} \\
& [T11^{i,j}_{i+k,j+l} AX_{i+k,j} BX_{i+k,j} \\
& + T22^{i,j}_{i+k,j+l} AY_{i,j+l} BY_{i,j+l} \\
& + (-1)^{k+l} T12^{i,j}_{i+k,j+l} (AX_{i+k,j} BY_{i,j+l} + AY_{i,j+l} BX_{i+k,j}).
\end{aligned}
$$

Finally, the $\mathcal{G}$ operator is again defined based on the adjoint relationship with $\mathcal{DIV}$. Unlike the cell-node method, in general, it is not practical to write $\mathcal{G}$ explicitly. Computing $\mathcal{G}$ requires inverting the matrix associated with the velocity inner-product, giving us a dense representation of the operator. We get around this by defining $\mathcal{G}$ implicitly, and forming a saddle-point problem with both pressure and velocity as unknowns.

Appendix B

Two-Phase Flow using MFD

Here we demonstrate an algorithm used for solving the two-phase flow problem using MFD. The formulation is similar to solving the two-phase flow problem with MFMFE [61]. We also note previous work in solving two-phase flows with MFD in [37]. In this formulation, we neglect the effect of capillary pressure and gravity, but allow for slightly compressible fluids. Starting with the two-phase equations,

$$\begin{aligned} v_\alpha &= -\frac{\rho_\alpha}{\mu_\alpha} k_{r\alpha} \mathbf{K} \nabla p_\alpha, \\ \frac{\partial \phi \rho_\alpha s_\alpha}{\partial t} &= -\nabla \cdot v_\alpha + Q_\alpha, \\ p_c &= p_o - p_w = 0, \\ s_w + s_o &= 1, \end{aligned} \tag{B.1}$$

we define the mobility of phase α as

$$\lambda_\alpha = \frac{k_{r\alpha}}{\mu_\alpha}. \tag{B.2}$$

By adding the saturation equations and using the saturation constraint,

$$\phi \frac{\partial(\rho_w s_w + \rho_o(1 - s_w))}{\partial t} = -\nabla \cdot v_t + Q_t, \tag{B.3}$$

where,

$$\begin{aligned} v_t &= -\rho_w \lambda_w \mathbf{K} \nabla p_w - \rho_o \lambda_o \mathbf{K} \nabla p_o && \text{(B.4)} \\ &= -\rho_w \lambda_w \mathbf{K} \nabla p_o - \rho_o \lambda_o \mathbf{K} \nabla p_o && \text{(B.5)} \\ &= -(\rho_w \lambda_w + \rho_o \lambda_o) \mathbf{K} \nabla p_o, && \text{(B.6)} \end{aligned}$$

is the total velocity, and $Q_t = Q_o + Q_w$ is the total source term. Note that we are taking p_o as the primary pressure variable. The key to using the mimetic method here is to replace the exact gradient and divergence operators with their discrete approximations, as follows:

$$\begin{aligned} \mathcal{G}_t \; &\approx (\rho_w \lambda_w + \rho_o \lambda_o) \mathbf{K} \nabla, \\ \mathcal{DIV} \; &\approx \nabla \cdot \; . \end{aligned} \qquad \text{(B.7)}$$

The operators $\mathcal{DIV}$ and $\mathcal{G}$ share the adjoint relation,

$$[\mathcal{G}_t \, p_h, u_h]_{X_h} = -[p_h, \mathcal{DIV} \, u_h]_{Q_h} \qquad \text{(B.8)}$$

The “weak” form of the problem is stated as:

Find $s_w, p_o \in Q_h$ and $v_t \in X_h$, such that

$$[v_t^{n+1}, u_h]_{X_h} - [p_o^{n+1}, \mathcal{DIV} \, u_h]_{Q_h} = 0, \qquad \forall u_h \in X_h \qquad \text{(B.9)}$$

$$[\phi \frac{\partial(\rho_w s_w^n + \rho_o(1 - s_w^n))}{\partial t}, q_h]_{Q_h} + [\mathcal{DIV} \, v_t^{n+1}, q_h]_{Q_h} = [Q_t^{n+1}, q_h]_{Q_h}, \quad \forall q_h \in Q_h. \qquad \text{(B.10)}$$

We denote (B.9) as

$$F_1(p_o^{n+1}, v_t^{n+1}) = 0. \qquad \text{(B.11)}$$

Assuming slight compressibility of the fluid, we have

$$\rho_\alpha(p) = \rho_\alpha^0(1 + C_{f\alpha}p), \tag{B.12}$$

using the backward Euler approximation for the time derivative, we get

$$\begin{aligned} &\left[\frac{\phi}{\Delta t}\rho_w(p_o^{n+1})s_w^{n+1} + \rho_o(p_o^{n+1})(1 - s_w^{n+1})), q_h\right]_{Q_h} \\ &\quad - \left[\frac{\phi}{\Delta t}(\rho_w(p_w^n)s_w^n + \rho_o(p_o^n)(1 - s_w^n)), q_h\right]_{Q_h} \\ &\qquad + \left[\mathcal{DIV}\, v_t^{n+1}, q_h\right]_{Q_h} - [Q_t^{n+1}, q_h]_{Q_h} = 0, \end{aligned} \tag{B.13}$$

which we will denote as

$$F_2(p_o^{n+1}, v_t^{n+1}) = 0. \tag{B.14}$$

The resulting Jacobian matrix is

$$\begin{pmatrix} \frac{\partial F_1}{\partial v_t^{n+1}} & \frac{\partial F_1}{\partial p_o^{n+1}} \\ \frac{\partial F_2}{\partial v_t^{n+1}} & \frac{\partial F_2}{\partial p_o^{n+1}} \end{pmatrix} = \begin{pmatrix} M & -\mathcal{DIV}^* \\ \mathcal{DIV} & C \end{pmatrix}, \tag{B.15}$$

with M, $\mathcal{DIV}$ and $\mathcal{DIV}^*$ constructed as usual. The matrix C is diagonal, and is defined as

$$C(E) = \frac{\phi|E|}{\Delta t}\left(\rho_w^0 C_{fw} s_w^{n+1} + \rho_o^0 C_{fo}(1 - s_w^{n+1})\right). \tag{B.16}$$

Next, we look at the saturation update. The water velocity is computed using a fractional flow function calculated from the upwinded mobility,

$$v_w = f_w v_t. \tag{B.17}$$

Applying the weak form directly to the saturation equation gives us,

$$\left[\phi\frac{\partial \rho_w s_w}{\partial t}, q_h\right]_{Q_h} = -[\mathcal{DIV}\, v_w^{n+1}, q_h]_{Q_h} + [Q_w^{n+1}, q_h]_{Q_h}. \tag{B.18}$$

Using backward Euler, we have

$$\left[\frac{\phi}{\Delta t}\left(\rho_w(p_w^{n+1})s_w^{n+1} - \rho_w(p_w^n)s_w^n\right), q_h\right]_{Q_h} = [-\mathcal{DIV}\, v_w^{n+1}, q_h]_{Q_h} + [Q_w^{n+1}, q_h]_{Q_h}. \tag{B.19}$$

We can now apply the standard IMPES method to solve the system or use iterative coupling for a better solution to the overall system [40].

Bibliography

[1] S. Agmon. *Lectures on elliptic boundary value problems*, volume 2. American Mathematical Soc., 2010.

[2] O. Al-Hinai, G. Singh, T. Almani, G. Pencheva, and M. F. Wheeler. Modeling multiphase flow with nonplanar fractures. In *2013 SPE Reservoir Simulation Symposium*, 2013.

[3] C. Alboin, J. Jaffré, J.E. Roberts, and C. Serres. Modeling fractures as interfaces for flow and transport in porous media. In *Fluid Flow and Transport in Porous Media, Mathematical and Numerical Treatment: (South Hadley, MA, 2001)*, volume 295 of *Contemp. Math.*, pages 13–24. Amer Mathematical Society, 2002.

[4] F. O. Alpak. A mimetic finite-volume-discretization operator for reservoir simulation. In *SPE Reservoir Simulation Symposium Houston, Texas*, 2007.

[5] T. Arbogast, L. C. Cowsar, M. F. Wheeler, and I. Yotov. Mixed finite element methods on non-matching multiblock grids. *SIAM J. Numer. Anal*, 37:1295–1315, 2000.

[6] T. Arbogast, M. F. Wheeler, and I. Yotov. Mixed finite elements for elliptic problems with tensor coefficients as cell-centered finite differences.

SIAM Journal on Numerical Analysis, 34(2):pp. 828–852, 1997.

[7] T. Arbogast, M.F. Wheeler, and I. Yotov. Mixed finite elements for elliptic problems with tensor coefficients as cell-centered finite differences. accepted for publication in SIAM J. Num. Anal., 1995.

[8] K. Aziz. Reservoir simulation grids: Opportunities and problems. *Journal of Petroleum Technology*, pages 658–663, 1993.

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

[10] S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, and H. Zhang. PETSc Web page. `http://www.mcs.anl.gov/petsc`, 2014.

[11] P. Bochev and M. J. Hyman. Principles of mimetic discretizations of differential operators. In Douglas N. Arnold, Pavel B. Bochev, Richard B. Lehoucq, Roy A. Nicolaides, Mikhail Shashkov, Douglas N. Arnold, and Fadil Santosa, editors, *Compatible Spatial Discretizations*, volume 142 of *The IMA Volumes in Mathematics and its Applications*, pages 89–119. Springer New York, 2006.

[12] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 2nd edition, April 2002.

[13] F. Brezzi and A. Buffa. Innovative mimetic discretizations for electromagnetic problems. *Journal of Computational and Applied Mathematics*, 234(6):1980 – 1987, 2010. Eighth International Conference on Mathematical and Numerical Aspects of Waves (Waves 2007).

[14] F. Brezzi, A. Buffa, and K. Lipnikov. Mimetic finite differences for elliptic problems, 2008. Published online December 5, 2008.

[15] F. Brezzi, K. Lipnikov, and M. Shashkov. Convergence of the mimetic finite difference method for diffusion problems on polyhedral meshes. *SIAM J. Numer. Anal.*, 43(5):1872–1896 (electronic), 2005.

[16] F. Brezzi, K. Lipnikov, and V. Simoncini. A family of mimetic finite difference methods on polygonal and polyhedral meshes. *M^3AS. Mathematical Models & Methods in Applied Sciences*, 15(10):1533–1551, 2005.

[17] Z. Chen, G. Huan, and Y Ma. *Computational Methods for Multiphase Flows in Porous Media (Computational Science and Engineering 2).* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[18] M. A. Christie. Tenth spe comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Evaluation & Engineering*, 4:308–317, 2001.

[19] M.A. Christie, M.J. Blunt, et al. Tenth spe comparative solution project: A comparison of upscaling techniques. *SPE Reservoir Evaluation &*

Engineering, 4(04):308–317, 2001.

[20] H Cinco-Ley and H. Z. Meng. Pressure transient analysis of wells with finite conductivity vertical fractures in double porosity reservoirs. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1988.

[21] B. L. da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L.D. Marini, and A. Russo. Basic principles of virtual element methods. *Mathematical Models and Methods in Applied Sciences*, 23(01):199–214, 2013.

[22] L. B. da Veiga, V. Gyrya, K. Lipnikov, and G. Manzini. Mimetic finite difference method for the stokes problem on polygonal meshes. *Journal of Computational Physics*, 228(19):7215 – 7232, 2009.

[23] L. Dalcin. petsc4py. `https://code.google.com/p/petsc4py/`, 2013.

[24] J. Droniou, R. Eymard, T. Gallouët, and R. Herbin. A unified approach to mimetic finite difference, hybrid finite volume and mixed finite volume methods. Technical report, Dec 2008.

[25] R. Eymard, T. Gallouët, and R. Herbin. *Finite volume methods*. Handb. Numer. Anal., VII. North-Holland, Amsterdam, 2000.

[26] A.P. Favorskii, A. A. Samarskii, M. Yu. Shashkov, and V. F. Tishkin. Operational finite-difference schemes. *Differential Equations*, 17:854–862, 1981.

[27] A. Firoozabadi. *Thermodynamics of Hydrocarbon Reservoirs.* 1999.

[28] B. Ganis, V. Girault, M. Mear, G. Singh, and M. F. Wheeler. Modeling fractures in a poro-elastic medium. *Oil & Gas Science and Technology–Revue d'IFP Energies nouvelles*, 2013.

[29] H. Hoteit and A. Firoozabadi. An efficient numerical model for incompressible two-phase flow in fractured media. *Advances in Water Resources*, 31(6):891–905, 2008.

[30] J. M. Hyman, M. J. Shashkov, and S. Steinberg. The numerical solution of diffusion problems in strongly heterogeneous non-isotropic materials. *J. Comput. Phys.*, 132(1):130–148, 1997.

[31] M. J. Hyman and M. Shashkov. Mimetic discretizations for maxwell's equations. *Journal of Computational Physics*, 151(2):881 – 909, 1999.

[32] J. J. Douglas, D. W. Peaceman, and H. H. Rachford. A method for calculating multi-dimensional immiscible displacement. *Trans. SPE AIME*, 216:297–306, 1959.

[33] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[34] H. Kazemi and J. Gilman. A pressure transient analysis of naturally fractured reservoirs with uniform fracture distribution. *SPE J*, 9(4):451–462, 1969.

[35] K. Lipnikov, G. Manzini, and M. Shashkov. Mimetic finite difference method. *Journal of Computational Physics*, 257, Part B(0):1163 – 1227, 2014. Physics-compatible numerical methods.

[36] K. Lipnikov, G. Manzini, and D. Svyatskiy. Monotonicity conditions in the mimetic finite difference method. In Jaroslav Fo**v**rt, Ji**v**rí Fürst, Jan Halama, Raphaèle Herbin, and Florence Hubert, editors, *Finite Volumes for Complex Applications VI Problems and Perspectives*, volume 4 of *Springer Proceedings in Mathematics*, pages 653–661. Springer Berlin Heidelberg, 2011.

[37] K. Lipnikov, J.D. Moulton, and D. Svyatskiy. A multilevel multiscale mimetic (m3) method for two-phase flows in porous media. *Journal of Computational Physics*, 227(14):6727 – 6753, 2008.

[38] K. Lipnikov, M. Shashkov, and I. Yotov. Local flux mimetic finite difference methods. *Numerische Mathematik*, 112:115–152, 2009.

[39] K. Lipnikov, M. J. Shashkov, and D. Svyatskiy. The mimetic finite difference discretization of diffusion problem on unstructured polyhedral meshes. *J. Comput. Phys*, 211:473–491, 2005.

[40] B. Lu, T. Alshaalan, and M. F. Wheeler. Iteratively coupled reservoir simulation for multiphase flow. In *SPE Annual Technical Conference and Exhibition*, 2007.

[41] G. Manzini. The mimetic finite difference method. In R. Eymard and Hérard J.-M., editors, *Finite Volumes for Complex Applications V, Problems and Perspectives*, pages 119–134. Wiley, 2008.

[42] R.K. Mehra, R.A. Heidemann, and K. Aziz. An accelerated successive substitution algorithm. *The Canadian Journal of Chemical Engineering*, 61, August 1983.

[43] B. Mirtich. Fast and accurate computation of polyhedral mass properties. *journal of graphics tools*, 1(2):31–50, 1996.

[44] A. Moinfar, A. Varavei, K. Sepehrnoori, and R. Johns. Development of a novel and computationally-efficient discrete-fracture model to study ior processes in naturally fractured reservoirs. In *SPE Improved Oil Recovery Symposium*, 2012.

[45] P. Ni. *Anderson Acceleration of Fixed-point Iteration with Applications to Electronic Structure Computations.* PhD thesis, Worcester Polytechnic Institute, 2009.

[46] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. Wiley. com, 2009.

[47] T. Oliphant. Python for scientific computing. *Computing in Science & Engineering*, 9(3):10–20, 2007.

[48] C. Palagi and K. Aziz. Use of voronoi grid in reservoir simluation. *SPE Advanced Techonology Series*, 2:69–77, 1994.

[49] J. Rungamornrat, M. F. Wheeler, and M. E. Mear. Coupling of fracture/non-newtonian flow for simulating nonplanar evolution of hydraulic fractures. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2005.

[50] T.F. Russell and M.F. Wheeler. Finite element and finite difference methods for continuous flows in porous media. In R.E. Ewing, editor, *Mathematics of Reservoir Simulation, Frontiers in Applied Mathematics Vol. 1*. SIAM, 1983.

[51] C. Rycroft. Voro++: A three-dimensional voronoi cell library in c++. 2009.

[52] Y. Saad. *Iterative Methods for Sparse Linear Systems*. 1996.

[53] A. A. Samarskii, V. F. Tishkin, A. P. Favorskii, and M. Yu. Shashkov. Use of the support operator method for constructing difference analogues of operations of tensor analysis. *Differentsial nye Uravneniya*, 18(7):1251–1256, 1287, 1982. English translation: Differential Equations 18 (1982), no. 7, 881–885.

[54] M. J. Shashkov. *Conservative finite-difference methods on general grids*. CRC Press, Boca Raton, FL, 1996. With 1 IBM-PC floppy disk (3.5 inch; HD).

[55] J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[56] H. Stehfest. Algorithm 368: Numerical inversion of laplace transforms [d5]. *Communications of the ACM*, 13(1):47–49, 1970.

[57] G. Vitaliy and K. Lipnikov. High-order mimetic finite difference method for diffusion problems on polygonal meshes. *J. Comput. Phys.*, 227:8841–8854, October 2008.

[58] J.E. Warren and P.J. Root. The behavior of naturally fractured reservoirs. *Old SPE Journal*, 3(3):245–255, 1963.

[59] A. Weiser and M. F. Wheeler. On convergence of block-centered finite differences for elliptic problems. *SIAM Journal on Numerical Analysis*, 25(2):351–375, 1988.

[60] J. A. Wheeler, M. F. Wheeler, and I. Yotov. Enhanced velocity mixed finite element methods for flow in multiblock domains, 2002.

[61] M. F. Wheeler, G. Xue, and I. Yotov. Accurate cell-centered discretizations for modeling multiphase flow in porous media on general hexahedral

and simplicial grids. *SPE Reservoir Simulation Symposium,Woodlands, TX*, 2011.

[62] M. F. Wheeler, Guangri Xue, and I. Yotov. Accurate cell-centered discretizations for modeling multiphase flow in porous media on general hexahedral and simplicial grids. *SPE Journal*, 2012.

Vita

Omar Al-Khattab Al-Hinai was born in Dhahran, Saudi Arabia. He finished high school in 2000 at the Dhahran Ahliyyah School also in Dhahran. Omar went to Boston Univeresity where he completed a double major in Computer Science and Mathematics and a Masters in Computer Science under the supervision of Prof. Margrit Betke. After working for Saudi Aramco in Dhahran for two years, he joined the Computational and Applied Mathematics program at the Institute for Computational Engineering and Sciences (ICES) at the University of Texas at Austin.

Permanent address: ohinai@gmail.com

This dissertation was typeset with LaTeX[†] by the author.

[†] LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.