Summer 2020

Variable Compact Multi-Point Upscaling Schemes for Anisotropic Diffusion Problems in Three-Dimensions

James Quinlan

VARIABLE COMPACT MULTI-POINT UPSCALING SCHEMES FOR ANISOTROPIC DIFFUSION PROBLEMS IN THREE-DIMENSIONS

by

James Edward Quinlan

A Dissertation
Submitted to the Graduate School,
the College of Arts and Sciences
and the School of Mathematics and Natural Sciences
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

Approved by:

Dr. James Lambers, Committee Chair
Dr. C.S. Chen
Dr. Haiyan Tian
Dr. Huiqing Zhu

Dr. James Lambers	Dr. Bernd Schroeder	Dr. Karen S. Coats
Committee Chair	Director of School	Dean of the Graduate School

August 2020

ABSTRACT

Simulation is a useful tool to mitigate risk and uncertainty in subsurface flow models that contain geometrically complex features and in which the permeability field is highly heterogeneous. However, due to the level of detail in the underlying geocellular description, an upscaling procedure is needed to generate a coarsened model that is computationally feasible to perform simulations. These procedures require additional attention when coefficients in the system exhibit full-tensor anisotropy due to heterogeneity or not aligned with the computational grid. In this thesis, we generalize a multi-point finite volume scheme in several ways and benchmark it against the industry-standard routines. Specifically, we extend a local transmissibility upscaling method to three-dimensional domains and incorporate adaptive mesh refinement. Our method uses spatially varying and compact multi-point flux approximations (MPFA), based on the Variable Compact Multi-Point (VCMP) method previously introduced for two-dimensional Cartesian grids in [59]. Moreover, the optimization algorithm that selects the transmissibility weights is generalized. Numerical results show that VCMP improves upscaling accuracy compared to local TPFA upscaling methods and the local-global TPFA upscaling method.

ACKNOWLEDGMENTS

I want to thank my advisor Dr. James Lambers. I am very grateful to have worked with him on this research. He provided me with his original research code for VCMP and gave many valuable comments and suggestions to improve it. I always looked forward to our meetings to discuss concepts and results. I hope we can continue our collaboration.

I want to thank my committee members: Dr. Chen, Dr. Tian, and Dr. Zhu. I value your time serving on this committee and appreciate your dedication to all the students you serve.

I want to thank Dr. Todd Edwards of Miami University of Ohio. In addition to his editorial suggestions on improving clarity and readability, I am grateful for his time listening to me explain the details of the topic and his challenging questions.

I want to acknowledge Dr. Jui Ding from the University of Southern Mississippi. I appreciate him teaching me Functional Analysis as well as inviting me to his home for a traditional Chinese feast. I would also like to thank Dr. Joseph Kolibal, formerly from the University of Southern Mississippi, initially made it possible to pursue this degree and his mentorship over the years.

I want to acknowledge my past professors from Ohio State University that played a role in my life and career. In particular, Dr. Ed Overman, Dr. Douglas Owens (my original Ph.D. advisor), and the late Dr. Joseph Ferrar. I spent many hours in each of their offices and always enjoyed the time in their company.

Most importantly, I would like to thank my family for their love and support; my brothers Tom and Dave, sister Carol, and my partner Andrea. I would especially like to acknowledge my daughter Hannah who began this journey with me in October of 1998 and has been there every step of the way. She sacrificed far more for my accomplishment than I ever did; I am forever grateful. Lastly, although they are no longer with me, I would be remiss if I did not acknowledge my parents. They taught me the value of education and the benefits of hard work. I miss them every day.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure

LIST OF TABLES

Table

LIST OF ABBREVIATIONS

ALG	-	Adaptive Local-Global
AMR	-	Adaptive Mesh Refinement
ATP	-	Accuracy Tuning Parameter
CFD	-	Computational Fluid Dynamics
CCAR	-	Cartesian Cell-based Anisotropically Refinement
EL	-	Extended Local
FDM	-	Finite Difference Method
FEM	-	Finite Element Method
FVM	-	Finite Volume Method
L	-	Local (upscaling)
LG	-	Local-Global
MDCS	-	MATLAB Distributed Computing Server
ML	-	Multi-Level
MLLG	-	Multi-Level Local-Global
MPFA	-	Multi-Point Flux Approximation
MS	-	Multi-scale
MSFV	-	Multi-scale Finite Volume
MSTPFA	-	Multi-scale two-point flux-approximation
PCT	-	Parallel Computing Toolbox
R	-	Regional (upscaling)
SPMD	-	Single Program Multi-data
SPE	-	Society of Petroleum Engineers
TPFA	-	Two Point Flux Approximation
VCMP	-	Variable Compact Multi-Point

NOTATION AND GLOSSARY

f	flux or fractional flow
i, j, k	block indices
K	absolute permeability tensor
L_x, L_y, L_z	grid lengths
l_x, l_y, l_z	refinement levels
p, $\mathbf{p}$	pressure value, pressure vector
t	VCMP transmissibility weights
u	Darcy's velocity
q	local flow rate
Q	total flow rate at the boundary
t, $\mathbf{t}$	coarse transmissibility stencil weights, vector
T	fine-scale face transmissibility

Greek Letters

α	optimization weighting
β	optimization weighting
Δ	directional dimension
δ	regularization penalty
γ	regularization penalty
ε	tolerance value
η	refinement index
ι	isometric coefficient
μ	viscosity
ρ	density
ϕ	porosity
τ	refinement threshold
ξ	percentage of interval

Subscripts

m	iteration number

Superscripts

*	upscaled parameter
^{c}	coarse-scale variable
^{f}	fine-scale variable
^{T}	transpose operation

Chapter 1

INTRODUCTION

Modeling fluid flow through subsurface porous media occurs in a wide range of science and engineering applications, including controlling groundwater contamination, predicting escape rates of sequestered carbon dioxide (CO_2), and evaluating strategies for the placement of wells in petroleum reservoirs [84]. Flow behavior in porous media is fundamentally described by permeability [55], which is highly heterogeneous, often discontinuous, and its features differ on various length scales [48]. This extreme variability in the size and shape of the voids in the media present significant challenges in accurate models. Realistic models must also adequately represent geological complexity, such as interconnected channels, faults, fissures, and pinch-outs. Therefore, any characterization of the medium will naturally include a high degree of uncertainty.

This uncertainty can be quantified either by applying stochastic techniques [75] or by simulating flows using a large number of deterministic realizations [11]. The computational grid of a *fine-scale*[1] model is typically composed of 10^7 to 10^9 cells. However, high-resolution flow simulations are computationally prohibitive in processing time and computer memory, as thousands of runs may be required to account for significant parameter variation [33]. Consequently, the numerical simulations must be performed on coarse grids compared to the given measurement scale. It is, therefore, necessary to determine appropriate coarse-scale model parameters. This research develops and evaluates methods for estimating these parameters to perform three-dimensional flow simulations through porous media from the context of petroleum reservoirs.

1.1 Background and Motivation

Upscaling, also referred to as homogenization, is an averaging process based on Effective Medium Theory [24]. The main idea of upscaling is to replace heterogeneous values of a particular property of the medium with an equivalent homogeneous one. In the context of modeling fluid flow through porous media, upscaling determine suitable values for the permeability (or transmissibility) at the measurement scale for use on the numerical scale

[1] Fine-scale is also referred to as geological, geostatistical, geocellular, or measurement scale.

[53]. These values must be selected such that flow behavior in the two systems is similar [71]. As a result, the upscaled system can be solved with greater efficiency [39]. Although upscaled systems frequently take a different form than fully resolved models, the literature states that an analogous pressure system can be solved independently of scale [10, 73].

Several classifications of upscaling exist. For instance, upscaling can be classified by the model parameter being represented (see Section 1.1.1). In single-phase flow problems, the parameter values include porosity, absolute permeability, or transmissibility[2] [25]. Relative permeability is also an option in two-phase flow problems [33].

Upscaling is also classified based on the method by which parameters are estimated. These classification include [71]: (i) power-averaging; (ii) renormalization techniques; and (iii) flow-based (or pressure-solver) methods (see Section 1.1.2). Power-law averaging is efficient but limited in applicability, see [53, 80]. Renormalization is an iterative process that evaluates averages over small regions of the reservoir to produce a distribution with lower variance than the original [58]. This process repeats until the estimates stabilize. Although renormalization is efficient and produces robust approximations, the method can suffer from accuracy issues [53]. Flow-based methods (e.g., [17, 31, 81]) equate flux per volume at fine-scale with flux through an equal volume homogeneous medium under the same boundary conditions (see Section 1.1.2). The size of the domain in determining the parameters is a sub-classifier.

1.1.1 Parameter Upscaling

Permeability upscaling entails homogenization of the fine-scale permeability field for use in the coarse-scale model. Cardwell and Parsons [12] showed that coarse block permeability values, defined at cell centers, fall between the arithmetic and harmonic means of the fine-scale values within that block. Journel et al. [56] consequently suggested applying power averaging techniques to compute the effective permeability. This approach computes the effective coarse block permeability based on local boundary conditions that produce similar flow results at the fine-scale. Interface transmissibility values used by the finite-volume simulator are then computed. Permeability upscaling is well-documented in the literature (see [17, 20, 32, 56, 70, 72, 80] and references within). Figure 1.1 provides a conceptual illustration[3] of permeability upscaling.

Transmissibility upscaling determines coarse-scale transmissibility directly in the pro-

[2]Transmissibility is a quantity defined at cell interfaces relating flow between adjacent blocks to the pressure difference.

[3]Reproduced with permission by the Society of Petroleum Engineers. Originally published in *Petroleum Engineering Handbook*, Volume 5, page 1423 [51].

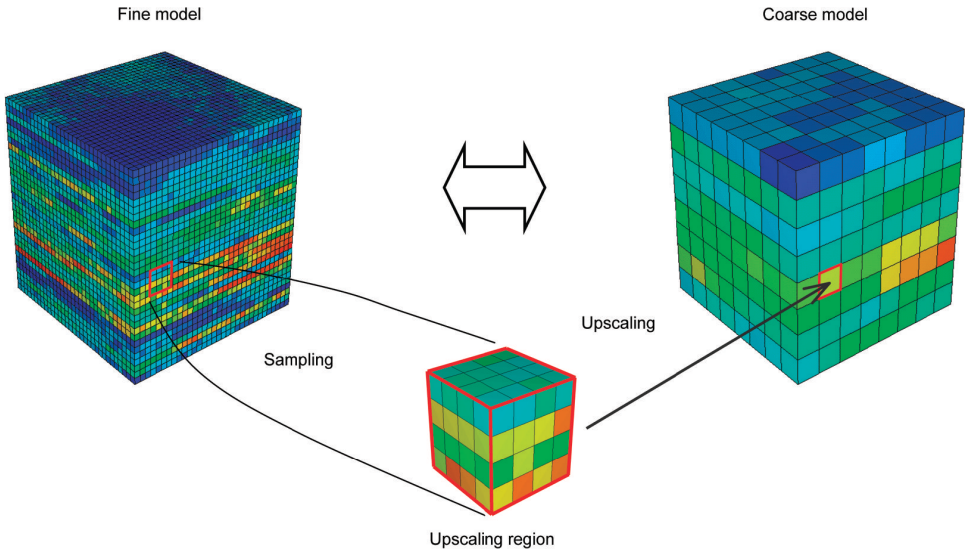


Figure 1.1: Conceptual illustration of the upscaling process.

cess, which provides more accurate coarse-scale models than permeability upscaling for highly heterogeneous systems [19, 33]. Therefore, our investigation will focused on this classification.

1.1.2 Flow-based Upscaling

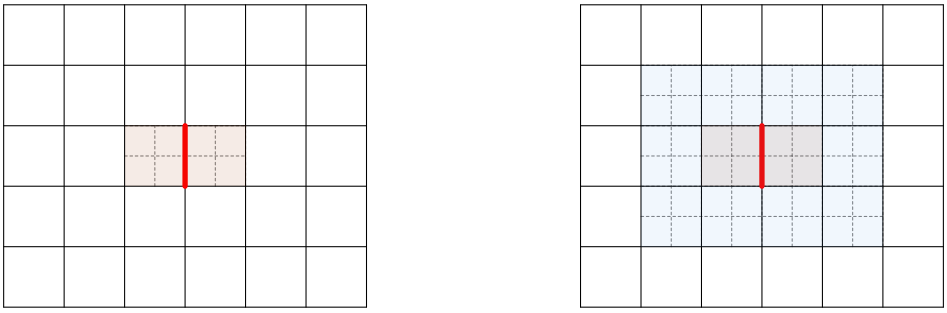
Flow-based upscaling strategies set up and solve a system of representative flow equations under some artificial set of boundary conditions [25]. Flow-based methods offer greater flexibility to work with different grid topologies and discretization schemes. The size of the computational region for which the fine-scale experiment is performed to calibrate the coarse-scale parameter often distinguishes the flow-based method. Four primary classes exist: (i) local; (ii) regional (or extended-local); (iii) global; and (iv) quasi-global.

A *local* technique estimates the model parameter by solving a pressure equation corresponding to a local fine-scale region under a given set of boundary conditions. Then coarse block pressure values are obtained through an averaging process, and these averages are used to estimate the parameter of interest (i.e., transmissibility). It is worth noting that several averaging techniques exist that warrant further investigation.

Computational efficiency and ease of implementation make local methods highly attractive. However, accuracy becomes an issue when there are large-scale connected flow paths [59]. Figure 1.2a schematically illustrates local transmissibility upscaling with coarse cells (in heavier lines) and the underlying fine Cartesian grid (in lighter, dotted lines).

Regional, or *extended local*, upscaling is an over-sampling technique that uses border regions (or *skin*) to improve parameter estimation of complicated cases by including fine-

scale information from cells neighboring the local region. In particular, the inclusion of an extended fine-scale region reduces the influence of the local boundary conditions. Additionally, it introduces the effects of larger-scale connectivity in the calculation of upscaled parameters [78]. Several investigators have applied this technique in the calculation of upscaled permeability on Cartesian grids (e.g., [43, 49, 78, 81]). Lambers et al. [59], Gerritsen et al. [42], Gerritsen and Lambers [41], and Wen et al. [79] have all applied the use of border regions in transmissibility upscaling. Figure 1.2b shows an extended fine-scale computational region in two dimensions with border ring radius $r_c = 1$ indicated with shading.



(a) Local (b) Regional

Figure 1.2: Local and regional 2D computational domain. Regional domain composed of fine-scale cells adjacent to face and a border region with radius $r_c = 1$.

Accuracy in both flow and transport calculations is directly proportional to the size of the boundary padding [78]. A radius one ($r_c = 1$) extended local region consists of one border ring. Depending on the permeability field, the use of border regions increases accuracy relative to local methods, but naturally at the cost of higher computation. Moreover, Wen et al. [78] found diminishing returns in the accuracy associated with larger ring sizes and suggest $r_c = 1$ is sufficient. They reported that the use of border regions is not necessary if the local technique adequately models permeability connectivity.

Global upscaling utilizes fine-scale flow results over the global domain, Ω, to obtain the coarse-scale parameters for the model [53] and improves accuracy by reducing large-scale connectivity errors. However, this is not true for all boundary conditions [33]. Global methods have been used to upscale both permeability and transmissibility. For instance, Holden and Nielsen [50] computed the effective permeability field by minimizing error in the solutions between the fine and coarse scales. The global upscaling approach avoids using artificial boundary conditions around the upscaling region, which, as already noted, produces more substantial upscaling errors. Unfortunately, in most cases, global fine-scale

upscaling is computationally prohibitive. Consequently, it will not be considered in this work. For further discussion and review of global methods, refer to [18, 33, 39, 50, 53].

Quasi-Global approaches substitute approximate fine-scale global information in the upscaling process to reduce errors mostly associated with connected flow paths through the medium. For instance, Chen et al. [19] developed an iterative procedure that couples nonlocal flow information in the local upscaling process. Their algorithm first performs local (or regional) upscaling with some arbitrary boundary conditions (e.g., constant pressure - no flow) to obtain model parameters used in the global coarse-scale simulations. The pressure values obtained from the solution to the global coarse-scale problem are then used as the local boundary conditions in the next iteration of local upscaling calculations. The local upscaling and global pressure solves are repeated until a self-consistent solution exists. In several test cases, the coupled local-global process provided more accurate coarse-scale results compared to existing local and regional approaches [19, 20, 30, 41] including extensions to three-dimensions [79].

The selection of the boundary conditions for the local region is of considerable importance in flow-based methods as it can impact the accuracy of the upscaling method [19]. Researchers have suggested various boundary conditions for the local flow equations. Durlofsky [31] introduced the use of periodic boundary conditions that leads to a full equivalent permeability tensor. He showed in all cases that periodic conditions produced a symmetric positive definite equivalent permeability tensors.

Open-side boundaries (e.g., linear boundary conditions, suggested by [44, 74]), allow flow to enter and leave the cell at any point along the boundary of the sides parallel to the main flow direction. Open-side conditions also produce a full-tensor for effective permeability.

Warren and Price [77] introduced sealed-sides boundary conditions such as *constant pressure and no-flow*, also referred to as *pressure-flow* [31]. These conditions employ a unidirectional constant pressure gradient by using both Dirichlet and Neumann conditions to drive flow in the principal direction. Specifically, "generic" pressure-flow conditions prescribe unit pressure on the "inlet" boundary, zero on the "outlet", and no-flow across all other sides. Sealed-side methods, however, tend to underestimate flow characteristics [57].

1.2 Statement of the Problem

Numerical simulations are necessary for the study of fluid through porous media. However, the computation in performing *fine-scale*[4] flow simulations is not feasible. Therefore,

[4]Fine-scale is also referred to as *geological*, *geostatistical*, *geocellular*, or *measurement* scale.

simulations are run on coarse grids compared to the given measurement scale. An *upscaling* process that coarsens and "scales up" the fine-scale computational grid by a factor of 10 to 1000 and then homogenizes the model parameters to perform the direct simulations efficiently is described in Section 1.1. Furthermore, the robustness of the upscaled parameter is highly desirable to be applicable under various cases and conditions. As such, upscaling is an active area of research [62]. Although our focus is on single-phase flow, accurate upscaling methods for single-phase flow is a prerequisite to multi-phase flow problems.

Several techniques exist in two-dimensions that consider highly heterogeneous permeability fields and attempt to resolve flows not aligned with the computational grid. In particular, Lambers et al. [59] developed the Variable Compact Multi-point (VCMP) method that uses a spatially varying cell-based finite volume discretization scheme. In this work, we extend that method to three-dimensions and address the following questions:

1. Is VCMP an effective method of approximating second-order elliptic equations representing flow through heterogeneous media?
2. How effective is VCMP compared to two-point flux approximations methods in three-dimensions?
3. How does the addition of border-ring affect the accuracy and efficiency of upscaling techniques?
4. Does VCMP combined with local mesh refinement effectively represent large-scale heterogeneities on computational grids for single-phase flows?
5. Does the integration of local-global iteration with VCMP improve the results by reducing the effect of local boundary conditions and include global features such as connected flow paths?
6. Do our methods successfully control the local grid adaptation during the upscaling process to reduce upscaling errors?
7. How sensitive is VCMP (and TPFA) to grid dimensions and high-aspect ratios?
8. How computationally efficient is VCMP, in particular, compared to TPFA?

1.3 Purpose and Scope

The purpose of this research is to develop and evaluate a three-dimensional compact multi-point transmissibility upscaling algorithm for Cartesian and rectilinear grids based on two-dimensional versions developed by Lambers et al. [59]. The algorithm should accurately

represent anisotropic flows and capture features, such as large-scale connected flow paths found in fluvial systems. In addition to accuracy, computational efficiency is central to our scheme. Moreover, this method, and its variants, should provide robust results on various data sets, boundary conditions, upscaling factors, and grids. These goals are accomplished through the stencil selection and compaction process.

This research fills several gaps in the current literature. First, it extends algorithms established by Lambers et al. [59] from two-dimensions to three-dimensions. The method is tested on challenging permeability fields and a variety of boundary conditions.

Second, we generalize the constrained minimization formulation used in [18, 42, 59] that yields the optimized transmissibility weights. In those works, the least-squares fitting procedure with only an L_2 penalty is applied to determine the global transmissibility weights of a given coarse-scale face. In this work, we also include the L_1 penalty to further compact and select stencils in the flow equations. The addition of the L_1 penalty yields sparse models. The inclusion of both penalties increases the number of free parameters, providing more control over the discretization process.

Third, several features from [41, 59] are included in our extensions, such as adaptive mesh refinement, the use of a border region, and local-global techniques. Although Wen et al. [79] implemented the regional and local-global procedures to standard two-point discretization schemes in three-dimensions, we implement multi-point methods, thus providing the first such coupling of local-global techniques to our method in three-dimensions. We also include adaptive grid refinement within the context of all these approaches.

Lastly, we extend several classes of two-point methods. In particular, Gerritsen et al. [41] integrated local-global and multi-level local-global with two-point methodologies on two-dimensional Cartesian and rectilinear grids. We apply those same methods to three-dimensions.

In this study, we limit our attention to steady-state, incompressible, single-phase flow in three spatial dimensions. We do not consider the presence of sources or sinks that may represent near-well upscaling. However, our methods are not limited to those constraints. The numerical algorithm presented here can employ parallel processing locally or on distributed cloud computing servers.

1.4 Document Outline

The document is organized by chapter and section as follows. Chapter 2 provides mathematical preliminaries and background information. In particular, Section 2.1 presents the governing, single-phase, Naiver-Stokes flow equation, and its weak form. Section 2.2 covers

two finite-volume discretization schemes and identifies the advantages and disadvantages of each of their use. Specifically, Subsection 2.2.1 introduces the two-point flux approximation scheme used on both the fine and coarse-scale. Subsection 2.2.2 introduces the multi-point flux approximation method that forms the basis of our method, including a brief discussion about monotonicity issues associated with multi-point methods in general. Section 2.3 gives details regarding the adaptive mesh refinement strategy employed by our upscaling algorithm.

Chapter 3 introduces the VCMP method and lists its key features. Subsection 3.1.1 reviews the two-dimensional development. We proceed to present our extension and implementation of this method to three-dimensions in Subsection 3.1.2. In this section, we discuss the necessary steps in the processes to obtain the upscaled transmissibility weights. Notably, we present and discuss our generalization of the constrained least-squares optimization problem that is the crux of our upscaling technique. Specifically, it contains a discussion of possible component scaling and regularization configurations.

Section 3.2 enumerates the steps in our local (or extended-local) and local-global algorithms including stopping criteria for the iteration. Pseudocode is also given in this section for the general upscaling process. The complete code of the upscaling algorithm can be found in Appendix A. We conclude the Chapter with Section 3.3 that details the incorporation of adaptive mesh refinement in VCMP and provides a flowchart of the overall processes.

Chapter 4 outlines our methodology. In particular, Section 4.1 identifies our permeability data sets and their sources. Subsection 4.1.1 highlights the test-suite used in our numerical experiments. Subsection 4.1.2 and 4.1.3 lists identifiers for the methods and lists the boundary conditions respectively. We include error metrics used to judge the accuracy of the methods in Section 4.2.

The results of our experiments are presented in Chapter 5. Section 5.4 provides algorithm benchmarks and charts. Chapter 6 includes conclusions and suggestions for future directions.

Chapter 2

MATHEMATICAL PRELIMINARIES

2.1 Single-phase fine-scale flow equations

The flow of a single fluid (single component or a homogeneous mixture) with density ρ in the x-direction through the control volume $\Delta x \Delta y \Delta z$ representing a porous medium is shown in Figure 2.1.

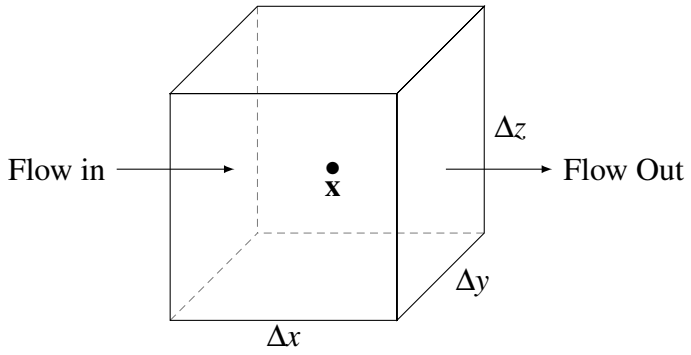


Figure 2.1: Control volume in 3D

The mass conservation equation in three-dimensions is derived by equating the difference between inflow and outflow of the control volume in all coordinate directions to the sum of mass accumulation (or depletion) due to fluid compression and sources/sinks (see [9, p. 8]). After dividing by volume and the time interval, then taking spatial limits, the mass conservation equation for three-dimensional Cartesian flow in a porous medium is given by

$$-\left(\frac{\partial}{\partial x}\rho u_x + \frac{\partial}{\partial y}\rho u_y + \frac{\partial}{\partial z}\rho u_z\right) = \frac{\partial}{\partial t}(\phi\rho) + \tilde{q}, \tag{2.1}$$

where ϕ is porosity, t is time, $\tilde{q}$ is mass per unit volume per unit time with positive indicating a source (negative a sink), and $\mathbf{u} = \langle u_x, u_y, u_z \rangle^T$ is fluid velocity of units (i.e., u_x is a volumetric velocity in the x-direction). Using the gradient operator $\nabla = \left\langle \frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z} \right\rangle$ notation, this equation can be written compactly as,

$$-\nabla \cdot (\rho \mathbf{u}) = \frac{\partial}{\partial t}(\phi\rho) + \tilde{q}. \tag{2.2}$$

For an incompressible fluid in a system with no sources or sinks, (2.2) reduces to

$$\nabla \cdot (\rho \mathbf{u}) = 0. \tag{2.3}$$

In addition to mass conservation, we will conserve momentum using Darcy's Law [28] relating the flow rate to the potential. The differential form of this relationship is

$$\mathbf{u} = -\frac{1}{\mu} K(\mathbf{x}) \left(\nabla p + \rho \mathbf{g}\right), \tag{2.4}$$

where μ is fluid viscosity, $\mathbf{x} = (x, y, z)$ represents the spatial coordinates, $K(\mathbf{x})$ is the absolute *permeability*[1] tensor, ∇p the pressure gradient, and $\mathbf{g}$ is a scaled gravitational acceleration vector. Orienting the coordinate system so that z is the positive vertical component in the downward direction, we can write

$$\rho \mathbf{g} = -\rho g \nabla z = -\gamma \nabla z.$$

Darcy's Law then becomes,

$$\mathbf{u} = -\frac{1}{\mu} K(\mathbf{x}) \left(\nabla p - \gamma \nabla z\right). \tag{2.5}$$

In general, the permeability tensor includes off-diagonal terms representing directional flow. In particular, the entry k_{xy} in (2.6) is the permeability in the x-direction given flow in the y-direction. These coefficients are typically to be highly oscillatory in space.

$$K(\mathbf{x}) = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix}. \tag{2.6}$$

At the fine-scale, $K(\mathbf{x})$ is assumed to be symmetric positive definite. Moreover, in an isotropic medium, $K(\mathbf{x}) = \alpha I$, where I denotes the identity tensor and $\alpha > 0$. When there is no ambiguity, $K(\mathbf{x})$ will be denoted as K.

The dimensionless pressure equation describing steady state, incompressible, single-phase flow through a porous media in three-dimensions is obtained by combining (2.3) with (2.5) to obtain the governing pressure equation:

$$\nabla \cdot \left(\frac{\rho}{\mu} K(\nabla p - \gamma \nabla z) \right) = 0. \tag{2.7}$$

In the absence of gravity, (2.7) becomes

$$\nabla \cdot (K \nabla p) = 0, \tag{2.8}$$

[1] Capacity of the rock to transmit fluid. Permeability values of reservoir rock ranges from approximately 0.1 millidarcy (mD) to 10 D, (1 mD $\approx 10^{-9}$ m^2) [62].

or in component form

$$\frac{\partial}{\partial x_i}\left[k_{ij}(x)\frac{\partial p}{\partial x_j}\right]=0. \tag{2.9}$$

Gauss' Divergence Theorem (2.10) that equates the volume integral to the net outward flux over grid block (i,j,k) is used to solve (2.8).

$$\iiint_{\Omega}\nabla\cdot K\nabla p\,dV=\iint_{\partial\Omega}K\nabla p\cdot\mathbf{n}\,d\omega \tag{2.10}$$

Because the fluid is incompressible, the right-hand side of (2.10) is zero, i.e.,

$$\iint_{\partial\Omega}K\nabla p\cdot\mathbf{n}\,d\omega=0. \tag{2.11}$$

Flow equations describing multi-phase (e.g., oil, water, and gas phases) flow are derived similarly. For the derivation of equations describing two-phase flow on the fine-scale, see [9, 21, 33]. Accurate upscaling of single-phase flow models is essential for the development of accurate coarse models of multi-phase flow [79].

2.2 Discretization Techniques

The governing equations will be discretized using a cell-centered finite-volume approach. Finite-element methods have also been applied to solve flow equations. For discussion on their application, see [1, 23, 37, 47, 52, 61, 74]. Finite-volume discretizations affect the quality of the solutions and are classified by the number of nodes used in the calculation as *two-point* or *multi-point*.

2.2.1 Two-point Flux Approximation

A Two-point Flux Approximation (TPFA) is a standard discretization scheme employed in control-volume formulations of elliptic operators (i.e., ∇^2) that satisfy local mass conservation [9, 83]. TPFA computes the flux at the interface between *two* adjacent cells (see Figure 2.2) using the inter-block transmissibility and the cell-centered pressure difference. On three dimensional Cartesian grids indexed by (i,j,k) in the x,y and z directions respectively, the flux f across the cell face between the neighboring cells (i,j,k) and $(i+1,j,k)$ is given by

$$f=-(T_x)_{i+\frac{1}{2},j,k}(p_{ijk}-p_{i+1,j,k}). \tag{2.12}$$

The transmissibility in the x-direction, denoted by $(T_x)_{i+\frac{1}{2},j,k}$ in Equation (2.12), is given by the formula,

$$(T_x)_{i+\frac{1}{2},j,k} = \frac{2(k_x)_{i+\frac{1}{2},j,k}\Delta y_j \Delta z_k}{\Delta x_{i+1} + \Delta x_i}, \tag{2.13}$$

where $(k_x)_{i+\frac{1}{2},j,k}$ is defined as the harmonic average of cell-block permeabilities weighted by the distances (Δx_i and Δx_{i+1}) to the neighboring cell center. Specifically, the interface permeability is

$$(k_x)_{i+\frac{1}{2},j,k} = \frac{(\Delta x_i + \Delta x_{i+1})(k_x)_{i,j,k}(k_x)_{i+1,j,k}}{\Delta x_{i+1}(k_x)_{i,j,k} + \Delta x_i (k_x)_{i+1,j,k}}. \tag{2.14}$$

Transmissibilities in the other coordinate directions are defined analogously.

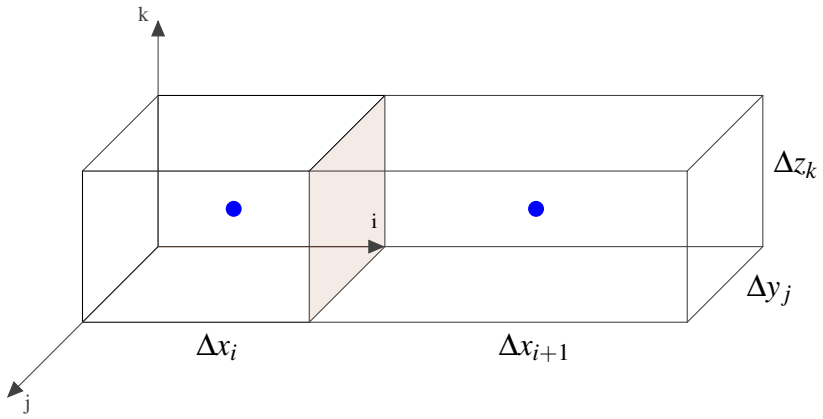


Figure 2.2: Two three-dimensional Cartesian grid blocks with indicated dimensions. Pressure values at cell-centers (blue dots) and tranmissibility (weighted harmonic average of cell permeabilities) at the interface (shaded) between cells are required in flux calculations.

In three-dimensions, a TPFA scheme leads to a 7-point stencil for the pressure equation (2.8):

$$\begin{aligned} &(T_x)_{i-1/2,j,k}(p_{i-1,j,k} - p_{i,j,k}) + (T_x)_{i+1/2,j,k}(p_{i+1,j,k} - p_{i,j,k}) \\ &+(T_y)_{i,j-1/2,k}(p_{i,j-1,k} - p_{i,j,k}) + (T_y)_{i,j+1/2,k}(p_{i,j+1,k} - p_{i,j,k}) \\ &+(T_z)_{i,j,k-1/2}(p_{i,j,k-1} - p_{i,j,k}) + (T_z)_{i,j,k+1/2}(p_{i,j,k+1} - p_{i,j,k}) = 0, \end{aligned} \tag{2.15}$$

where $p_{i,j,k}$ represents pressure in cell (i,j,k). The resulting discretized system yields a sparse M-matrix that satisfies the monotonicity property [68] (see Section 2.2.2 for further discussion on monotonicity).

In local transmissibility upscaling, the discretization (2.15) is applied to each fine-scale cell composing the region outlined by the two coarse grid blocks to solve the pressure equation (2.8) under certain boundary conditions, typically constant pressure no-flow. The solution is then used to compute the upscaled transmissility T^*,

$$T^* = \frac{f^c}{p_i^c - p_j^c}, \tag{2.16}$$

where p_i^c and p_j^c are volume averages of the fine-scale pressure values over coarse cells i and j, respectively. The flux across the interface, f^c, is obtained by integrating the fine-scale fluxes along the target face.

TPFA is easy to implement and fairly robust. It provides accurate results when the computational grid is in alignment with the principal components of the local permeability field (i.e., K-orthogonal). Therefore, TPFA is always applicable at the measurement scale. However, if the media is highly heterogeneous anisotropic, then K is most likely a non-symmetric full coefficient tensor. In this case, generalized techniques are needed that provide more accurate discretization of the flow equations [5, 6, 36, 60].

2.2.2 Multi-point Flux Approximation

TPFA schemes are simple to implement, produce monotone solutions, and generate sparse linear systems due to the compact nature of their stencil size. However, TPFA yields inconsistent approximations to the pressure equation (2.7) in the presence of full-tensor anisotropy. Full-tensor effects can arise on the coarse-scale as a result of upscaling even when the underlying fine-scale permeability is isotropic.

Several researchers have investigated procedures to improve the accuracy of TPFA methods. Combining local-global iteration with adaptive gridding [41] leads to further improvements making TPFA more competitive. Chen et al. [22] developed a nonlinear two-point flux approximation (NTPFA) algorithm for systems with full-tensor effects that provided accurate flow predictions over broad parameter ranges. Further, TPFA techniques coupled with quasi-local procedures have been shown to provide good upscaled transmissibilities in some challenging cases [19, 20, 22]. It remains an open question whether TPFA can be improved in terms of accuracy by applying *local-global* iteration, using multi-scale grid adaptation techniques, etc. in the process to remain a viable approach to flow simulation.

Multi-point flux approximation (MPFA) methods were developed to capture non-orthogonal grid effects, particularly from upscaling heterogeneous permeability tensors or flow non-alignment where TPFA tend to fail. In general, MPFA provides a more accurate representation of full-tensor anisotropy. Aavatsmark et al. developed several classes of MPFA

discretization methods (O-method, U-method, [6]) for both two-dimensional and three dimensional quadrilateral grids [8]. MPFA techniques exist for two and three-dimensional structured [2, 3, 4, 5, 42, 59] and unstructured grids [7, 35, 38], as well as general grids [16, 83]. MPFA methods have also been applied for general polygonal and polyhedral grids [16] and rectilinear grids [59].

MPFA methods express flux between adjacent grid blocks as a linear combination of pressure values from those two blocks as well as several surrounding blocks. In particular, MPFA approximates the flux by using the *multi-point flux expression*

$$f_i \approx \sum_{j \in \mathcal{J}} t_{i,j} p_j. \tag{2.17}$$

The coefficients t_{ij} are called the *transmissibility weights* and sum to zero in order to represent a constant solution exactly (i.e., $\sum_{j \in J} t_j = 0$). The quantity p_j is the cell-centered pressure value of the jth grid block. For a three-dimensional Cartesian grid, the index set $\mathcal{J}$ consists of a subset of cell numbers surrounding the target cell interface. In vector notation, the multi-point flux expression (2.17) is represented as,

$$f_i \approx -\mathbf{t}^T \mathbf{p}, \tag{2.18}$$

with

$$\mathbf{t} = [t_{i,1}\, t_{i,2} \cdots t_{i,n}]^T \quad \text{and} \quad \mathbf{p} = [p_1\, p_2 \cdots p_n]^T.$$

Consequently, MPFA methods lead to larger stencil sizes and therefore increase computational complexity. Stencil sizes typically vary between nine-point in two-dimensions and 18 to 27-point in three-dimensions [36]. Aavatsmark et al. [3] introduce the L-method for quadrilateral grids minimized the number of entries in the flux stencils while honoring uniform flow fields. A discussion of the convergence rates for MPFA methods can be found in [3].

Monotonicity

MPFA methods can produce non-physical oscillations in the pressure solution. The resulting solutions violate the discrete maximum principle [76], a consequence of a non-montone matrix used to discretize the second-order elliptic boundary value problem. MPFA methods that attempt to produce compact flux stencils such as the L-method [2] and VCMP [59] increase the domain of monotonicity. Gerritsen et al. [42] developed a predictor-corrector method, called the *M-fix*, to the elements of the matrix to ensure an M-matrix[2], and hence

[2] A nonsingular matrix $A = (a_{ij})$ with nonpositive off-diagonal elements ($a_{ij} \leq 0$ for $i \neq j$) and whose inverse is nonnegative ($A^{-1} \geq 0$).

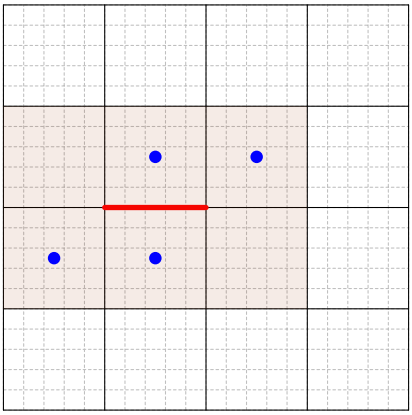


Figure 2.3: Compact MPFA stencil in 2D with an underlying fine grid structure. Six coarse cells shaded represents the local domain–four of the six adjacent coarse cells used in the flux stencil.

guarantee monotonicity. For further discussions on the monotonicity of MPFA methods see [3, 65, 69, 64].

2.3 Cartesian Cell-based Anisotropic Refinement

Local mesh refinement is another effective strategy of reducing upscaling errors while controlling excessive computation associated with high-resolution models [41, 66]. In the context of flow simulation, these methodologies typically introduce larger grid densities in high flow locales. In subsurface flow scenarios, refinement can also capture geological features such as channels, faults, or pinch-outs and better resolve permeability in regions with considerable heterogeneity. In particular, local flow-based mesh refinement combined with an accurate upscaling process provides realistic coarse-scale models of the fine-scale description [33]. Adaptive mesh techniques produce computational grids that selectively refine the local domain while controlling computation.

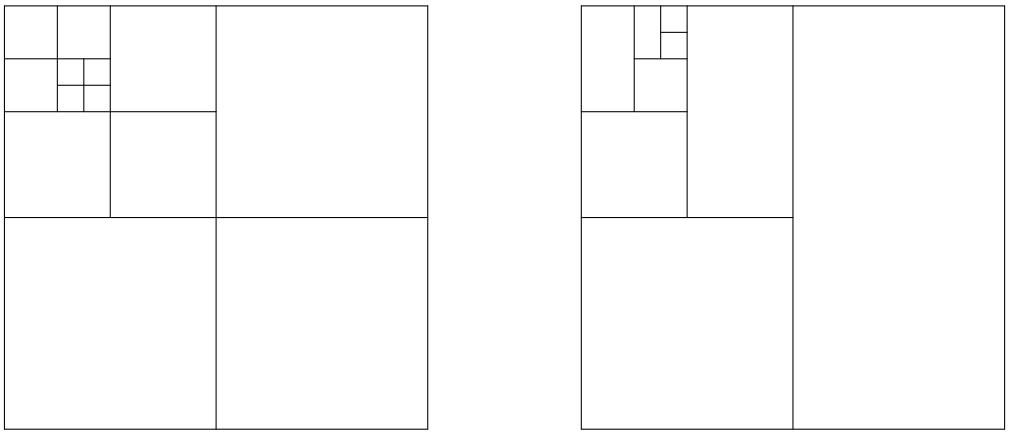
There are several flow-based grid strategies and refinement techniques for both structured and unstructured grids used in upscaling. For example, see [15, 46] for procedures on curvilinear grid generation in three-dimensional systems, [40] for the elastic grid framework that refines coarse cells in such a way to minimize permeability heterogeneity, [34] for nonuniform coarsening in both two and three dimensions on stratigraphic grids. See [63] and the references within for a comprehensive review of general grid generation techniques used in solving partial differential equations.

Cartesian cell geometry is frequently used in the literature as the base grid model in flow simulations and applied to local, regional, and global upscaling procedures (i.e.,

[19, 22, 33, 46, 59, 78, 81]). A Cartesian grid structure is easy to generate, represent in software, and maintain. Refinement is achieved by recursive cell division. However, uniform refinement of a Cartesian grid is discouraged due to insufficient capture of geological features, flow alignment, and increased computation [66].

Cartesian Cell-Based Anisotropic Refinement (CCAR) [45, 66] technique for regular Cartesian grids improves accuracy in regions of interest by increasing cells while controlling computation. CCAR grid topology allows aggressive grid generation with anisotropic refinement through automatic adaptation. Compared to hierarchical parent-child tree structures [27] primarily supporting isotropic refinements, CCAR imposes larger memory requirements as it is necessary to store coordinates, dimensions, neighbors, and refinement for each cell. However, CCAR is more computationally efficient because no tree traversals are necessary [66].

Anisotropic refinement is achieved by iteratively dividing rectangular cells in half along a single coordinate direction producing noncongruent parallelepipeds (rectangles in two-dimensions). Figure 2.4 illustrates isotropic (2.4a) and anisotropic (2.4b) refinement in two-dimensions.



(a) Isotropic Refinement (b) Anisotropic Refinement

Figure 2.4: Isotropic and anisotopic rectilinear Cartesian mesh refinements in two-dimensions.

Within the CCAR framework, it is common (e.g., [45, 41, 59, 66]) to limit the number of iterations so that no cell can have more than four neighbors in each of the three coordinate directions. Figure 2.5 shows a three-dimensional CCAR cell with four neighbors on the right, two above, and one in all other directions.

Using similar notation as [45, 66] for two-dimensions, we index three-dimensional anisotropic cells by two sets: (i) standard logical indexing $\{(i, j, k)\}$, with i, j, k, starting

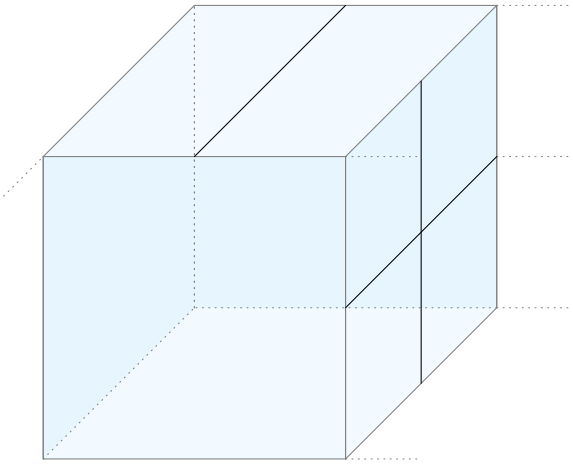


Figure 2.5: Three-dimensional CCAR cell with two neighbors above, four to the right, and one in the other four directions. CCAR cells in three-dimensions are limited to four neighbors in any direction.

at one, and (ii) a set to indicate the level of refinement, $\{[l_i, l_j, l_k]\}$, with coarsest cell level (i.e., no refinement) set equal to zero. For a given grid, G, with domain $L_x \times L_y \times L_z$ and origin $(0,0,0)$, the centroid (x_c, y_c, z_c), and dimensions $(\Delta x_i, \Delta y_j, \Delta z_k)$ of each cell can be calculated using the indexing scheme above, in particular,

$$(x_c, y_c, z_c) = \left(\frac{L_x\,(i-\frac{1}{2})}{2^{l_i}}, \frac{L_y\,(j-\frac{1}{2})}{2^{l_j}}, \frac{L_z\,(k-\frac{1}{2})}{2^{l_k}} \right) \tag{2.19}$$

$$(\Delta x_i, \Delta y_j, \Delta z_k) = \left(\frac{L_x}{2^{l_i}}, \frac{L_y}{2^{l_j}}, \frac{L_z}{2^{l_k}} \right). \tag{2.20}$$

These values are recorded in the cell structure of the MATLAB grid class.

Chapter 3

VARIABLE COMPACT MULTI-POINT METHOD

The Variable Compact Multi-point method (VCMP) developed in two-dimensions (see Gerritsen et al., [42]; Lambers et al. [59]) is a single-phase local multi-point transmissibility upscaling method that allows stencils to vary spatially. It constructs a flux approximation that adapts to the dominating principal direction of flow while minimizing the number of weights in the flux stencil that increases efficiency of linear solvers.

VCMP has been tested on multiple realizations of two-dimensional fine-scale permeability distributions that included both oriented layers and channelized paths for coarse models defined on Cartesian and adaptive Cartesian grids [59]. Gerritsen et al. [41] integrated VCMP with the iterative local-global method. Chen et al. [18] extended VCMP to irregular quadrilateral grids as well as developing a global VCMP technique.

VCMP is a multi-point flux continuous, locally conservative, finite volume scheme whose main features are:

- VCMP produces compact stencils while accommodating full-tensor anisotropy by remaining close to a two-point scheme. Two-point schemes are desirable because of consistency, efficiency (maximize the sparsity of the coefficients matrix), and robustness (as it likely generates an M-matrix). In particular, in cases involving homogeneous permeability, K-orthogonality, or flows aligned with the grid, VCMP would default to a two-point flux approximation method.

- VCMP performs well on grids with high aspect ratios, commonly used in reservoir simulation;

- VCMP can be applied to both regular and irregular grid topology;

- VCMP allows integration of adaptive mesh refinement techniques, such as the Cartesian Cell-based Adaptive Refinement (CCAR) strategy developed in [66] to effectively reduce upscaling errors (see Section 3.3).

- VCMP is extremely accurate for smooth pressure fields. If the pressure field is not smooth, accuracy can be improved by local grid refinement; and

- VCMP can include techniques that ensure an M-matrix (see [42]), and consequently monotone solutions.

3.1 VCMP in Two- and Three-dimensions

A detailed explanation of two-dimensional VCMP is presented in Subsection 3.1.1 and the extension to three-dimensions in Subsection 3.1.2.

3.1.1 Two-dimensional VCMP

VCMP is a multi-point method that allows the stencil nodes to vary per cell face as stated above. In two-dimensions (see [59]), it uses a subset of the six pressure values p_j, for $j = 1,\ldots,6$ that surround a target face as depicted in Figure 3.1. The stencil always includes the nodes on either side of the face (indicated by the larger blue dots). Two generic flow problems are solved on a local region around each face, by driving flow in each of the coordinate directions, represented by the arrows in Figure 3.1. The multi-point flux approximations are required to match the flow for these two problems at the face.

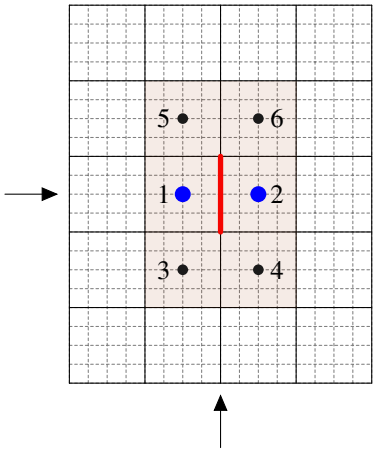


Figure 3.1: Interior target face (red) in a Cartesian grid surrounded by six pressure values. Arrows indicate direction of pressure driving flow.

For each $j = 1,\ldots,n$, ($n = 6$ in two-dimensions), let t_j denote the weight assigned to point j in the flux approximation, which has the general form

$$f = -\mathbf{t}^T\mathbf{p}, \tag{3.1}$$

where the vectors

$$\mathbf{t} = [t_1\, t_2 \cdots t_n]^T \quad \text{and} \quad \mathbf{p} = [p_1\, p_2 \cdots p_n]^T.$$

Using local (or regional) upscaling (see Section 1.1.2), we solve the pressure equation on a fine-scale local region containing the six points with two artificial boundary conditions. We let $p^1(x,y)$ and $p^2(x,y)$ be the solutions of these local problems, and $p^i{}_j$ denote the value of $p^i(x,y)$ at the point j. The boundary conditions can be arbitrarily chosen. Common examples include:

1. Constant pressure one at the inlet and zero on the outlet with no-flow on all other sides;

2. Linear pressure along the boundary;

3. Variable pressure between 0 and 1, and the gradients for the two problems are parallel to each of the coordinate axes.

The pressure field p^1 is computed using boundary values chosen so that flow is across the face, and p^2 is obtained from boundary values chosen to drive flow parallel to the face.

For $i = 1,2$, let f_i denote the coarse-scale flux (sum of fine-scale fluxes) across the face obtained from the local solution $p^i(x,y)$. To compute the transmissibility weights t_j for use in (2.18), we solve constrained optimization problem,

$$\underset{t\in S}{\text{minimize}} \sum_{i=1}^{2} \alpha_i^2 (\mathbf{t}^T \mathbf{p}^i + f_i)^2 + \sum_{j=3}^{6} \beta_j^2 t_j^2 \tag{3.2}$$

subject to the essential linear constraints

$$\begin{cases} \sum_{j=1}^{6} t_j = 0 \\ t_{2j-1} \leq 0 \ \text{ for } j = 1,\ldots,3 \\ t_{2j} \geq 0 \ \text{ for } j = 1,\ldots,3\,. \end{cases} \tag{3.3}$$

The weights α_i and β_j are free to be chosen. For example, in [59], $\alpha_i = |f_i|$ for $i = 1,2$ and $\beta_j = (|f_1| + |f_2|)/M$ for all j, where M is a tuning parameter controlling the relative importance of accuracy and robustness. The larger the value of M, yields higher accuracy. For small M, a two-point flux results (i.e., $t_j = 0$ for $j = 3,4,5,6$. Other weightings are possible.

The optimization problem (3.2) is solved using `lsqlin` function from MATLAB's Optimization Toolbox. This function employs an interior-point method, for details of active set method see [67, p. 480]. It finds an initial feasible solution by first solving a linear programming problem. If a minimizing set of weights t_j for $i = 1,\ldots 6$ is found, then we examine the weights and determine whether any of them are negligibly small.
Next, we present our three-dimensional extension.

3.1.2 Three-dimensional VCMP

Three dimensional VCMP, like with two-dimensions, uses a dynamic subset of pressure values p_j, for $j = 1, \ldots, n$ surrounding the target coarse interface (see Figure 3.2). Possible choices for the selection of the maximum number of nodes is $n = 10$ or $n = 18$ as the maximum stencil size.

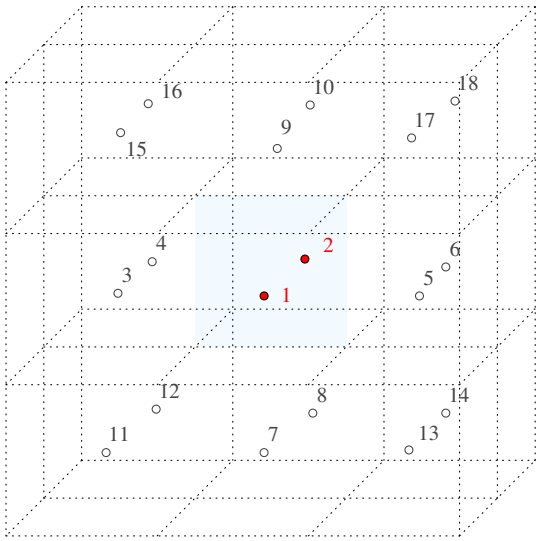


Figure 3.2: The 18 nodes surrounding target face (in blue).

The pressure equation (2.8) is solved on the local region of the fine grid containing these eighteen points using constant pressure, no-flow boundary conditions that drive flow in each of the three coordinate directions. For $i = 1, 2, 3$, we denoted the local pressure solutions by $\mathbf{p}^{(i)}(x, y, z)$ and p_j^i the value of $\mathbf{p}^{(i)}$ at node j. In addition, we denote the coarse-scale flux (sum of fine-scale fluxes) across the target face obtained from the local solution $p^{(i)}(x, y, z)$ by f_i for $i = 1, 2, 3$.

To compute upscaled transmissiblity weights, we generalized the optimization method found in [41, 59] that uses only an L_2 Tikhonov penalty [54]. In this work, we use elastic-net regularization (3.7) with zero-sum and sign constraints in (3.8) to overcome limitations in ridge regression [54] and aid in stencil selection. In particular, although the L_2 penalty shrinks coefficient weights towards zero, none are exactly zero, thus not compacting the stencil. In [59] for example, weights are deemed negligible if less than an arbitrary threshold value (e.g., less than 1% of largest weight) and removed. The optimization process is then repeated with those weights removed.

In general, the an L_1 penalty is expected to perform better in settings where there is

a small number of possible weights of significance (i.e., $n = 10$) [54]. It provides both continuous shrinkage and variable selection simultaneously [85]. The L_2 penalty is better when there is a possibility for larger stencils (i.e., $N = 18$). Elastic-net [85], however, combines both L_1 and L_2 penalties associated with Least Absolute Shrinkage and Selection Operator (LASSO) and Tikhonov regularization respectively. Using (3.1) as our context, for fixed $\lambda_1, \lambda_2 > 0$, we define the naive elastic-net loss function

$$L(\lambda_1, \lambda_2, \mathbf{t}) = |\mathbf{t}^T \mathbf{p} + f|_2^2 + \lambda_2 |\mathbf{t}|_2^2 + \lambda_1 |\mathbf{t}|_1 \tag{3.4}$$

where

$$|\mathbf{t}|^2 = \sum_{j=3}^{n} t_j^2 \quad \text{and} \quad |\mathbf{t}|_1 = \sum_{j=3}^{n} |t_j|.$$

If we let $\delta = \frac{\lambda_2}{\lambda_1 + \lambda_2}$, then $0 < \delta < 1$ is a tuning parameter responsible for mixing the two penalties L_1 and L_2. The loss function (3.4) can then be written in Lagrangian form,

$$L(\delta, \mathbf{t}) = \sum_{i=1}^{3} (\mathbf{t}^T \mathbf{p} + f_i)^2 + \delta \sum_{j=3}^{n} t_j^2 + (1 - \delta) \sum_{j=3}^{n} |t_j|. \tag{3.5}$$

Equation (3.5) can be further generalize by including additional parameters $\alpha_i, \beta_j, \gamma_j, \lambda$ for $i = 1, 2, 3$ and $j = 3, \ldots, n$ to produce,

$$L(\alpha_i, \beta_j, \gamma_j, \delta, \lambda, \mathbf{t}) = \sum_{i=1}^{3} \alpha_i^2 (\mathbf{t}^T \mathbf{p} + f_i)^2 + \lambda \left(\delta \sum_{j=3}^{n} \beta_j^2 t_j^2 + (1 - \delta) \sum_{j=3}^{n} \gamma_j |t_j| \right). \tag{3.6}$$

Scaling, Selection, and Regularization

The degrees of freedom in (3.6) provide ultimate flexibility in stencil selection. In our implementation, we assign $\beta_j = 1$ and $\gamma_j = 1$ for all j for simplicity, although these values are also free to be selected and affect the stencil selection process. We take $\lambda \geq 0$ and note that when $\lambda = 0$, (3.6) becomes an ordinary weighted least-squares problem, producing the largest stencils. If $\lambda \to \infty$, the large penalty greatly reduces the weights $t_3, \ldots, t_n$, but does not reduce the number of variables since it never leads to a zero value. Hence, this penalty does not technically compact the stencil.

The parameter $\delta \in (0, 1)$ mixes the penalty types, L_1 and L_2. When $\delta \to 1$, the method is Tikhonov regularization as applied in [18, 41, 59]. When $\delta \to 0$, the method becomes LASSO regression including only L_1 penalty. We set the penalty mixing parameter $\delta = 0.5$ by default. In this implementation, we assign $\beta_j = 1$ and $\gamma_j = 1$ for all j, although these values are also free to be selected and affect the stencil selection process.

There are several selection strategies for α_i. Lambers et al. [59] selected $\alpha_i = |f_i|$ and $\beta_j = \sum |f_i|/M$ to place emphasis on local flows with the largest flux. If the tuning parameter M is large, larger stencils are produced and higher flow accuracy results. For smaller M values, $t_j \approx 0$ for $j \geq 3$, driving closer to a two-point flux stencil.

For the α_i's, we adopt a gradient-based weighting strategy suggested by [41] in which we compute the average pressure gradient of the corresponding local solution. For each i, we set $\alpha_i = |\cos\theta_i|$, where θ_i is the angle between $\nabla p^{(i)}$ and the unit face norm. This method reduces the effect of strong flows parallel or nearly parallel to the face.

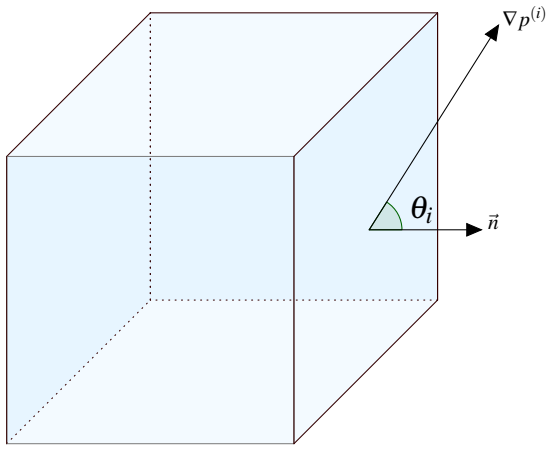


Figure 3.3: Angle θ_i between pressure gradient $\nabla p^{(i)}$ and face norm $\vec{n}$. Optimization weights in (3.7) that honor flow are calculated as $\alpha_i = |\cos\theta_i|$.

Therefore, the constrained optimization formulation used in this investigation is,

$$\underset{t\in S}{\text{minimize}} \left\{ \sum_{i=1}^{3} \alpha_i^2 (\mathbf{t}^T \mathbf{p} + f_i)^2 + \lambda \left(\delta \sum_{j=3}^{n} t_j^2 + (1-\delta) \sum_{j=3}^{n} |t_j| \right) \right\} \tag{3.7}$$

subject to the linear constraints:

$$\begin{cases} \sum_{j=1}^{n} t_j = 0 \\ t_{2j-1} \leq 0 \ \text{ for } j = 1, \ldots, n/2 \\ t_{2j} \geq 0 \ \text{ for } j = 1, \ldots, n/2\,. \end{cases} \tag{3.8}$$

3.2 VCMP Algorithm

An enumeration of the sequence of steps used in the VCMP Local (and regional) upscaling algorithm is given below, followed by pseudocode.

1. For each coarse-scale face, define a local (or extended-local) fine-scale region surrounding the face (see Figure 3.2). A border ring of radius one is included in regional upscaling.

2. Solve fine-scale flow equation (2.8) three times, once for each of the three sets of generic boundary conditions corresponding to the coordinate directions.

3. Average the pressure values in each of the coarse cells involved in the local domain, $\bar{p}^{(i)}$, and integrate the fine-scale flux across this face, $f_i = -\sum T\Delta p$.

4. Upscale the face transmissibility by solving the optimization problem (3.7) with constraints (3.8) for the stencil weight vector $\mathbf{t} = [t_1, t_2, \ldots, t_n]$. A two-point flux is the default value. There are several degrees of freedom in the optimization process. Selecting different regularization parameters has a significant effect on accuracy and robustness.

5. Solve the global coarse-scale pressure equation and compute relative errors in total flow rate, pressure, and flux.

   (a). If the local-global method is applied, then these pressure values are interpolated and used as local boundary conditions of Step 2 in the next iteration until consistency in the solutions is achieved. We use (3.9) as a termination criterion where the subscript indicates the iteration number and ε is a tolerance level.

$$\frac{|Q^c_{m+1} - Q^c_m|}{|Q^c_m|} \leq \varepsilon \tag{3.9}$$

Algorithm 1 Local (or regional) VCMP Transmissibility Upscaling

```
for each coarse grid face do
  create local (or regional) fine-scale domain
  for each coordinate direction do
    - solve local pressure equation (2.8)
    - compute average pressure per coarse block in local domain
    - compute fine-scale face flux f_i
  end for
  - optimize transmissibility t_i using Elastic net with given constraints
  if |t_i| ≈ 0 for i ≥ 3 then
    remove t_i
  end if
end for
```

3.3 VCMP with Adaptive Mesh Refinement

Extending the 2D cases, [41, 59, 66], we develop a version of VCMP upscaling that includes three-dimensional Cartesian Cell-based Anisotropic Refinement (CCAR), see Section 2.3. In addition to allowing aggressive adaptation near regions of high flow, the anisotropic refinement quality of CCAR strategies aligns the grid to the primary flow direction and with geological features to improve accuracy while minimizing computational cost of the addition of new cells. We restrict cells to have no more than four neighbors in three dimensions to maintain solution accuracy.

To obtain improved accuracy in the pressure solution and capture geological features, we apply a pre-processing step of isotropic refinement in regions near high permeability contrasts to a uniform Cartesian grid. These refinements are based on an approach developed by [82] that cluster fine cells with similar permeability. Then refinement flags are obtained by boundary detection. Our algorithm sets a limit on maximum number of cells in the base CCAR grid and refinement is repeated while this constraint is satisfied.

The mesh is then further refined during the upscaling process in high flow regions. In [41], an adaptive refinement condition was introduced in which cells surrounding a face are flagged for refinement if, in global coarse-scale flow simulations, a sufficiently large portion of the total flow passes through the face. Specifically, let A_i be the area of the ith face F_i and A the area of the boundary face of the global domain with the same orientation. If q_i is the flux across F_i obtained from a local pressure solve with flow driven in the direction norm to the face and Q represents the global flow at the inlet. Refinement occurs when inequality (3.10) is satisfied. Inequality (3.10) is called the *fair share* criteria.

Fair share Criteria. *Let A be the area of the inlet boundary of the global domain. Given any face F_i oriented Let A_i represent the area of the i-th face oriented*

$$|q_i| > \tau \left| \frac{A_i}{A} Q \right|. \tag{3.10}$$

In (3.10), $\left|\frac{A_i}{A}Q\right|$ can be interpreted as the face's "fair share" of the flow. The proportionality constant τ in (3.10) is a dynamically set threshold value to further control refinement by including the refinement level and the flow direction; larger values lead to less refinement. Let $\tau = \tau(\eta, \iota)$ be a function of the refinement level η and isotropic index ι, both defined below.

To slow refinement as the refinement increases (i.e., face area decreases), define

$$\eta = \frac{\log(A_c)}{\log(A_i)}, \tag{3.11}$$

where A_c is the area of the coarsest face in the mesh parallel to F_i. Note, as $A_i \to 0$, η increases, and hence reducing refinement by increasing the threshold. For example, with $A_c = 64$ and $A_i = 8$, $\eta = 2$, thus requiring double the fair share amount to trigger refinement.

To reduce the amount of refinement when the flow is aligned with the grid, we introduce an isotropic index (3.12) involving the transmissibility weights.

$$\iota = \left(\frac{\sum_{i=3}^{n} t_i^2}{\sum_{j=1}^{n} t_j^2} \right)^{1/2} \tag{3.12}$$

If flow is nearly orthogonal to the face, then all t_i are small and $\iota \approx 0$, not triggering refinement. However, when at least one of the weights t_i for $i = 3, \ldots, n$ is significant, $\iota \gg 0$, effectively lowering the refinement threshold in (3.10).

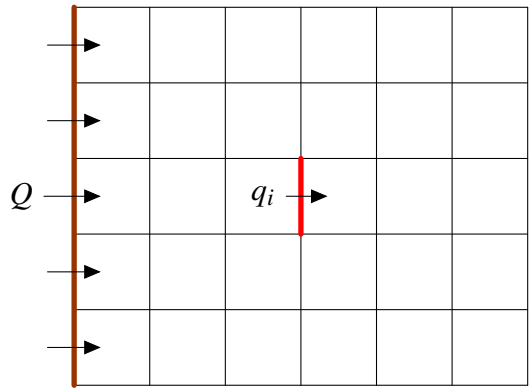


Figure 3.4: Fairshare Principle in 2D. The face flux, q_i, should not exceed an amount relative to the global flow amount Q.

In our software implementation, a CCAR grid is an instance of a Grid class object that assigns cell centers, dimensions, refinement levels, and connectivity data as properties. Refinement procedures are handled as methods of the class.

Figure 3.5 displays a flowchart of all processes in our implementation including upscaling, mesh refinement, and local-global iteration.

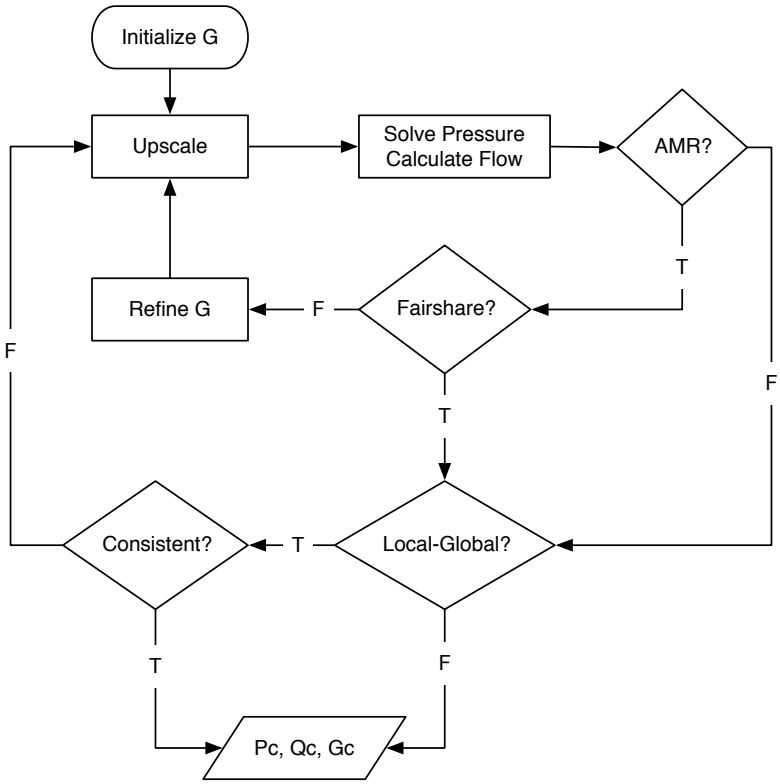


Figure 3.5: Flowchart of the Local-Global upscaling process with adaptive mesh refinement. Given an initial grid, G_0, algorithm ends after consistency in flow results yielding, P_c, Q_c, and G_c, the coarse-scale pressure solution, total flow, and final grid respectively.

Chapter 4

NUMERICAL EXPERIMENTS AND METHODS

This chapter discusses the methods and test cases implemented. Section 4.1 provides the permeability data used in our tests, including the data source. The test cases are presented in Subsection 4.1.1. Section 4.1.2 identifies the upscaling methods applied in our test cases. The global boundary conditions are specified in Section 4.1.3. Error measurements used to compare the quality of each method are described in Section 4.2.

Software to execute all tests was written in MATLAB (R2020a). The Optimization Toolbox was used to solve the constrained least-squares problem. The Parallel Computing Toolbox can be utilized as the algorithms are designed to running on parallel processors due to the independence in the upscaling procedure. Therefore, utilization of computing clusters are appropriate under these conditions.

4.1 Permeability Data

The data in our experiments are described on regular three-dimensional Cartesian grids and consist of geostatistical realizations of fine-scale permeability fields from two different sources. The first source contains realizations of permeability fabricated using a basic geostatistical method. First, porosity values ϕ are generated by a convolution of normally distributed data values with a generalized Gaussian kernel, $Ae^{-(ax^2+2bxy+cy^2)}$, where a, b, and c are defined by (4.1 - 4.3). We let $A = 1$ for simplicity.

$$a = \frac{\cos^2\theta}{2\sigma_x^2} + \frac{\sin^2\theta}{2\sigma_y^2} \tag{4.1}$$

$$b = -\frac{\sin 2\theta}{4\sigma_x^2} + \frac{\sin 2\theta}{4\sigma_y^2} \tag{4.2}$$

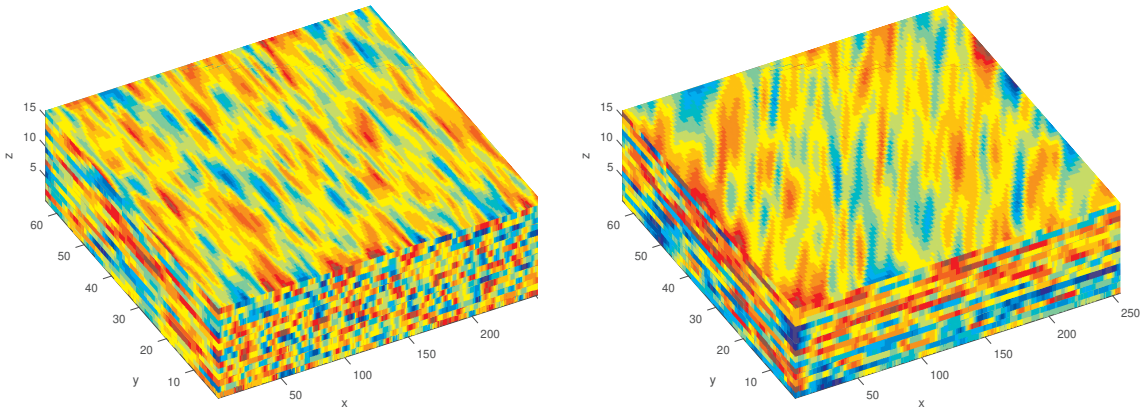
$$c = \frac{\sin^2\theta}{2\sigma_x^2} + \frac{\cos^2\theta}{2\sigma_y^2} \tag{4.3}$$

The angle of rotation θ and the standard deviations σ_x, σ_y are degrees of freedom that control grid alignment and correlation lengths. In our test cases, we set $\theta = \pi/4$ for non-K orthononal systems and and $\theta = 0$ for permeability aligned in the x directions. We use $\sigma_x = 1$ and $\sigma_y = 12$. In this simplified formulation, we assume the media particles have

uniform spherical diameter $d = 10^{-6}$ meters and set $\tau = 1$, then apply the Kozeny-Carman (see [13, 14]) relation (4.4) to calculate an isotropic permeability distribution K. Figure 4.1 shows two realizations of $256 \times 64 \times 16$ permeability fields generated by (4.4) with long correlation length aligned (Figure 4.1a) and not aligned (Figure 4.1b) with the grid that we include in our test cases.

$$K = \frac{d^2 \phi^3}{72\tau(1-\phi)^2} \tag{4.4}$$

We transform these values by standardizing to obtain log-normal permeability values for examination. For more realistic geostatistical models, use GSLIB [29].



(a) K-orthogonal

(b) Non K-orthogonal

Figure 4.1: Permeability fields generated using Kozeny-Carman formula (4.4).

The second data source comes from the 10th SPE Comparative Solution Project[1] [26], a standard benchmark widely used for evaluating upscaling techniques (see [78, 59]). In particular, this set contains relative permeability measurements of two formations from the Jurassic Upper Brent in the North Sea $(60.3272086, 4.399465)$: (i) the first 35 layers are representative of a nearshore Tarbert formation, and (ii) the lower 50 fluvial layers of the Upper Ness. The grid dimensions of the original model are $220 \times 60 \times 85$ cells, with the fine-scale cell size of 10 ft $\times$20 ft $\times$2 ft respectively. The overall physical domain is $2200 \times 1200 \times 170$ feet. Figure 4.2 shows the log-permeability distribution of a $256 \times 64 \times 16$ channelized region from the fluvial Upper Ness formation.

[1] Publicly available at `https://www.spe.org/web/csp/datasets/set02.htm`.

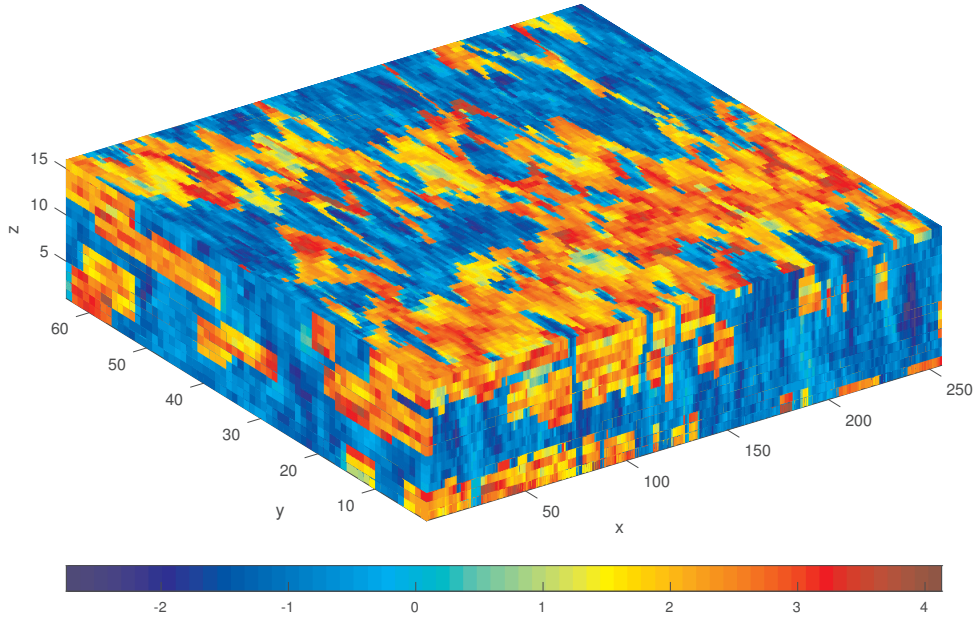


Figure 4.2: A $256 \times 64 \times 16$ channelized permeability field from the Upper Ness fluvial formation contained in the 10th SPE benchmark data.

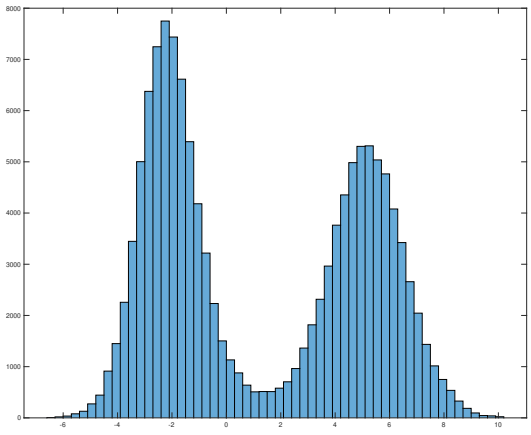


Figure 4.3: Bimodal log-normal distribution of permeability field from the Upper Ness Formation (SPE-10 Model).

4.1.1 Test Cases

In our implementation, the dimensions of the permeability field $L_x \times L_y \times L_z$ are powers of two (e.g., $256 \times 64 \times 16$) for computational convenience. Fine-scale block dimensions,

denoted as $\Delta x, \Delta y$, and Δz, are assumed to be unit size for all spatial dimensions, that is $\Delta x = \Delta y = \Delta z = 1$. Data used in our experiments consists of permeability fields from the sources listed in Section 4.1. Specifically, the experimental cases include:

1. Data Source 1: Generated using geostatistical method of convolution with a Gaussian kernel.

   (a). Twenty-five realizations of log-normally distributed permeability fields with long correlation lengths aligned with the grid in x direction $\theta = 0$.

   (b). Twenty-five realizations of log-normally distributed permeability fields with long correlation lengths not aligned with the grid, $\theta = \pi/4$.

2. Data Source 2: SPE-10

   (a). Twenty-five realizations from Near Shore Environment Layers 1 – 35 with domain $L_x = 64, L_y = 64$, and $L_z = 32$.

   (b). Twenty-five realizations from Near Shore Environment Layers 1 – 35 with domain $L_x = 256, L_y = 64$, and $L_z = 16$.

   (c). Twenty-five realizations from fluvial Upper Ness Layers 36 – 85 with domain $L_x = 64, L_y = 64$, and $L_z = 32$.

   (d). Twenty-five realizations from fluvial Upper Ness Layers 36 – 85 with domain $L_x = 256, L_y = 64$, and $L_z = 16$.

Numerical simulations of single-phase incompressible flow with given permeability fields were conducted to test and validate a cell-centered finite-volume scheme in three-dimensional using spatial variable and compact discretization stencils. For each of the five boundary conditions (see Table 4.2), combinations of local, regional, multi-level, and local-global flow-based transmissibility upscaling methods will be examined on high and low aspect ratios over two levels of heterogeneity (Layers 1-50 and Layers 51-85).

4.1.2 Experimental Upscaling Algorithms

In this study, we compare several upscaling algorithms in the test-suite. Table 4.1 lists the identifier, an abbreviation, and a short description of each algorithm. There are two finite-volume discretization schemes (TPFA and VCMP) and three flow-based upscaling methods (local, regional, and quasi-global). Adaptive mesh refinement (AMR) using fairshare principle is applied as indicated.

Algorithm	Short Description
TPFA-L	Local two-point flux approximation (TPFA)
TPFA-R	Regional (extended-local) TPFA
TPFA-ML	Multi-level (i.e., CCAR gridding) TPFA
TPFA-LG	Local-global TPFA
TPFA-MLLG	Multi-level Local-global TPFA
VCMP-L	Local VCMP
VCMP-R	Regional VCMP
VCMP-ML	Multi-level VCMP
VCMP-LG	Local-global VCMP
VCMP-MLLG	Multi-level Local-global VCMP

Table 4.1: Algorithm identifier and brief description of the upscaling algorithms used in this study.

4.1.3 Boundary Conditions

For each permeability field, we solve the pressure equation (2.8) with the following five sets of boundary conditions while testing the effectiveness of each algorithm in reducing process dependency, resolving the fine-scale velocity. Table 4.2 lists the boundary conditions used in this investigation. These sets are grouped into the two types enumerated below.

1. Constant pressure/no-flow. For each of the three coordinate directions x, y, and z, we prescribe Dirichlet conditions on "inlet" and "outlet" boundaries and Neumann no-flow conditions on the four other sides (e.g., see Figure 4.4a). These generic boundary conditions are referred to as *sealed-side conditions* and are also used in the upscaling process when computing local fine-scale solutions (see Section 1.1.2).

   (a). In the x-direction, we prescribe that $p(0,y,z) = 1$, $p(L_x,y,z) = 0$ and $u_y(x,0,z) = u_y(x,L_y,z) = u_z(x,y,0) = u_z(x,y,L_z) = 0$.

   (b). In the y-direction, these conditions are stipulated by $p(x,0,z) = 1$, $p(x,L_y,z) = 0$, and $u_x(0,y,z) = u_x(L_x,y,z) = u_z(x,y,0) = u_z(x,y,L_z) = 0$.

   (c). In the z-direction, the conditions are specified as $p(x,y,0) = 1$, $p(x,y,L_z) = 0$, and $u_x(0,y,z) = u_x(L_x,y,z) = u_y(x,0,z) = u_y(x,L_y,z) = 0$.

2. Corner-to-corner flows. These conditions fix pressure on the cross-diagonal coarse-scale corner regions instead of over continuous ranges of a particular dimensions as in the three constant pressure conditions above. To drive flow from the front-bottom-left

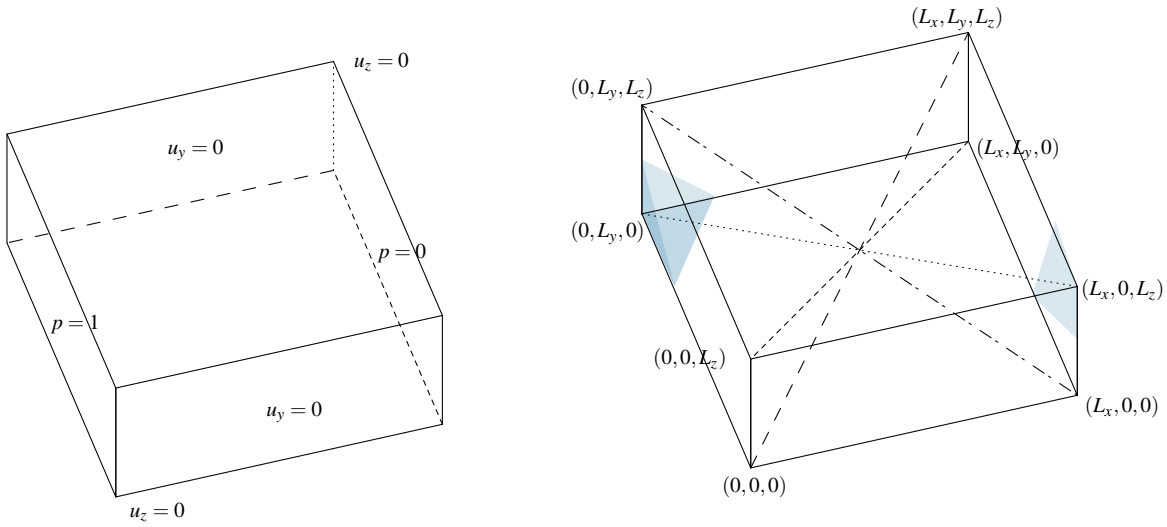
corner $\mathbf{x} = (0,0,0)$, to the back-upper-right corner $\mathbf{x} = (L_x, L_y, L_z)$ of the grid, define the six regions,

$$
\begin{aligned}
R_1 &= \{(x,y,z) \mid x = 0,\ 0 \le y \le \xi L_y, \text{ and } 0 \le z \le \xi L_z\} \\
R_2 &= \{(x,y,z) \mid 0 \le x \le \xi L_x,\ y = 0, \text{ and } 0 \le z \le \xi L_z\} \\
R_3 &= \{(x,y,z) \mid 0 \le x \le \xi L_x,\ 0 \le y \le \xi L_y, \text{ and } z = 0\} \\
R_4 &= \{(x,y,z) \mid x = L_x,\ 0 \le y \le (1-\xi) L_y, \text{ and } 0 \le z \le (1-\xi) L_z\} \\
R_5 &= \{(x,y,z) \mid 0 \le x \le (1-\xi) L_x,\ y = L_y, \text{ and } 0 \le z \le (1-\xi) L_z\} \\
R_6 &= \{(x,y,z) \mid 0 \le x \le (1-\xi) L_x,\ 0 \le y \le (1-\xi) L_y, \text{ and } z = L_z\},
\end{aligned}
$$

then prescribe

$$
p(x,y,z) = \begin{cases} 1, & \text{if } (x,y,z) \in R_1 \cup R_2 \cup R_3 \\ 0, & \text{if } (x,y,z) \in R_4 \cup R_5 \cup R_6 \end{cases} \tag{4.5}
$$

and no-flow conditions otherwise. Boundary condition 5 representing the corner-to-corner flow $(0, L_y, 0)$ to $(L_x, 0, L_z)$, see Figure 4.4b is defined similarly. The size of the corner regions is variable and controlled by the parameter $\xi \in (0, 0.5)$.



(a) Constant Pressure/No-flow

(b) Corner-to-corner

Figure 4.4: Experimental boundary conditions under consideration in this study: (a) Constant pressure/no-flow and (b) corner-to-corner flow, see (4.5).

4.2 Error Analysis

Errors for total flow through the system E_Q and L_2 norms for the pressure, E_p, are reported as percentages relative to fine-scale solutions. In particular, the relative errors for flow and

Boundary Condition	Description
1	1-0 Pressure-flow (in x)
2	1-0 Pressure-flow (in y)
3	1-0 Pressure-flow (in z)
4	Corner-to-corner flow, $(0,0,0) \rightarrow (L_x, L_y, L_z)$
5	Corner-to-corner flow, $(0, L_y, 0) \rightarrow (L_x, 0, L_z)$

Table 4.2: Experimental Boundary Conditions

pressure are defined respectively by:

$$E_Q = \frac{|Q_c - Q_f|}{|Q_f|} \quad \text{and} \quad E_p = \frac{||p - p_c||_2}{||p||_2}. \tag{4.6}$$

Q_c and Q_f, denote the coarse-scale and fine-scale flow calculations, respectively. The fine-scale pressure values p_f corresponding to each coarse cell block are computed using a volume-weighted average. Coarse flux is obtained by integrating fine-scale fluxes along the face of each coarse cell. Because unit pressure values are prescribed only on a small region of the domain, depending on the permeability field, corner to corner flows rates may be extremely low. In these cases, absolute errors ($|Q_f - Q_c|$) may be reported if relative errors are unreasonably large.

Chapter 5

RESULTS & ANALYSIS

Our numerical experiments demonstrate the effectiveness of upscaling methods on a variety of three-dimensional fine-scale permeability fields with boundary conditions. Our primary interest is on VCMP algorithm and its variants. However, we include TPFA as a baseline for comparisons as TPFA is commonly used in reservoir simulators. Table 4.1 in Section 4.1.2 lists an algorithm identifier as an abbreviation of the upscaling method referred to throughout this section and a short description. Table 4.2 in Section 4.1.3 lists the prescribed boundary conditions that the drive flow. Regularization parameters for all experiments, unless otherwise stated, are set at $\lambda = 10^{-3}$ and $\delta = 0.5$. The gradient weighting method is used in the assignment of the parameters α_i. For multi-level approaches, maximum number of cells is initially set to 1024 for large domains (e.g., $256 \times 64 \times 16$) and 512 for domains size $64 \times 64 \times 32$.

5.1 Results of Normal Permeability

Twenty-five realizations of the normal permeability field with dimensions $L_x = 64$, $L_y = 64$, and $L_z = 32$ were generated by (4.4) with $\sigma_x = 12$ and $\sigma_y = 1$ producing long correlation length. We conduct the following experiments applying methods TPFA-L, TPFA-R, TPFA-ML, VCMP-L, VCMP-R, and VCMP-ML:

1. K-orthogonal, $\theta = 0$, with coarse block dimensions $\Delta x = \Delta y = \Delta z = 8$;
2. K-orthogonal, $\theta = 0$, with coarse block dimensions $\Delta x = 16, \Delta y = 8, \Delta z = 4$;
3. Non K-orthogonal, $\theta = \pi/4$, with coarse block dimensions $\Delta x = \Delta y = \Delta z = 8$;
4. Non K-orthogonal, $\theta = \pi/4$, with coarse block dimensions $\Delta x = 16, \Delta y = 8, \Delta z = 4$.

Long correlation aligned with grid

Our first experiment examines twenty-five realizations of the permeability field described in the introduction of this section. This permeability field has long correlation length aligned with the grid (K-orthogonal) and uniform block dimensions identified in the outline above, (#1). Figure 5.1 shows one such realization.

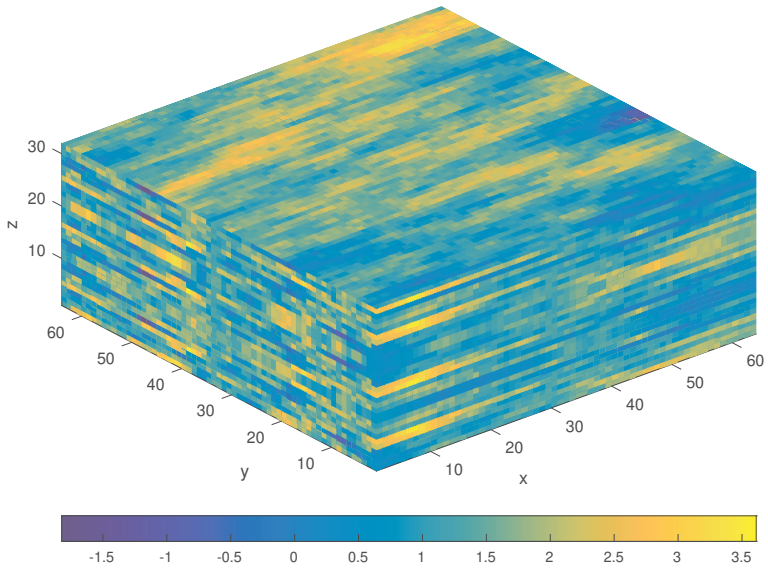


Figure 5.1: A $64 \times 64 \times 32$ permeability field with long correlation length aligned with grid at $\theta = 0$ with x axis.

Table 5.1 lists average percent relative error in the total flow rate using four upscaling methods on all five boundary conditions with maximal stencil width of $N = 10$. With the exception of the third boundary condition (flow in the vertical direction), TPFA is competitive with VCMP. This is to be expected when permeability is aligned with the grid. Table 5.2 lists the corresponding mean relative L_2 pressure errors. We note a reduction in pressure errors in corner flows for K-orthogonal permeability fields on uniform blocks.

We observe comparatively large errors in corner flows (Conditions 4 – 5) as opposed to pressure-flow conditions. In corner flows, unit pressure is applied to proportionately small areas, producing significantly lower flow through the system. In contrast, unit pressure imposed on the inlet boundary, which has a larger total area (e.g., 64×32 for Condition 1), produces relatively high pressure. The high-pressure system thus creates a higher flow. Therefore, we observe more substantial errors in corner flow conditions. For this reason, the phenomenon is seen throughout our results on corner flow conditions.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	0.53	1.16	1.16	0.96
2	1.57	1.16	1.05	1.33
3	3.04	0.29	1.38	0.22
4	11.21	9.33	10.10	6.77
5	19.94	18.05	12.02	10.66

Table 5.1: Mean relative error percentage in total flow with uniform block size on a long correlation $K-$orthogonal permeability field.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	1.87	1.52	1.34	0.93
2	2.12	2.27	1.46	1.07
3	2.36	2.63	2.15	1.44
4	7.28	7.55	6.54	6.02
5	5.33	6.02	4.69	5.08

Table 5.2: Relative error in pressure values from $K-$orthogonal permeability generated by Kozeny-Carman formula (4.4).

Extending the local domain to regional upscaling produces mixed results on this test case (see Table 5.3).

Flow Direction	TPFA-R	VCMP-R	TPFA-L	VCMP-L
1	0.41	0.07	0.53	1.16
2	0.30	0.17	1.57	1.16
3	0.80	1.04	3.04	0.29
4	11.46	11.73	11.21	9.33
5	12.50	12.11	19.94	18.05

Table 5.3: Mean relative error percentage in total flow on k-orthogonal uniform grid using regional upscaling with border radius $r = 1$.

In Table 5.4, mean percent relative flow error of twenty-five realizations of long correlation $K-$orthogonal permeability fields with high-aspect-ratio coarse blocks are shown. In all cases, VCMP outperforms TPFA. Adding mesh refinement further improves flow accuracy in all TPFA cases. However, the addition of refinement to VCMP yields mixed results on boundary conditions 1 - 3 and a significant improvement in corner conditions. Similar results hold for relative mean pressure values (see Table 5.5). Table 5.6 presents absolute flow errors instead of relative errors.

Flow Direction	Flow	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	2273	4.71	0.89	0.34	0.78
2	1173	5.9	0.16	4.44	0.64
3	1957	2.13	1.78	1.18	2.44
4	311	17.17	14.32	10.51	7.85
5	194	19.76	18.48	14.05	11.63

Table 5.4: Mean flow and relative mean flow rate error (in percent) on a long correlation $K-$orthogonal permeability field with high-aspect-ratio coarse blocks.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	1.5	1.55	1.82	0.62
2	1.49	1.51	1.43	0.67
3	1.67	1.8	1.02	0.68
4	6.08	6.33	3.31	3.63
5	4.91	5.47	3.32	3.62

Table 5.5: Mean pressure error (in percent) on a long correlation $K-$orthogonal permeability field with high-aspect-ratio coarse blocks.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	107.32	20.27	7.77	17.91
2	69.41	1.96	52.16	7.62
3	41.66	35.00	23.24	47.76
4	53.11	44.14	33.01	25.14
5	39.44	36.56	27.46	22.85

Table 5.6: Mean absolute flow errors, $|Q_f - Q_c|$, in $K-$orthogonal permeability with high aspect ratio coarse grid blocks, $\Delta x = 16, \Delta y = 8$, and $\Delta z = 4$.

Long correlation not aligned with grid

Figure 5.2 depicts one realization of a permeability field with long correlation length not aligned with the grid used in this experiment. We present detailed upscaling results for both uniform and nonuniform coarse grid block sizes as outlined above. We include results for TPFA methods for completeness and comparisons, though our emphasis is on VCMP strategies.

Tables 5.7 and 5.8 show results for the mean flow rate and pressure errors for uniform coarse grid blocks respectively. First, we note that flow errors are smaller for VCMP-L compared to TPFA-L. All differences are significant except for boundary condition 5. VCMP with mesh refinement, VCMP-ML, improved accuracy in flow errors both along and across

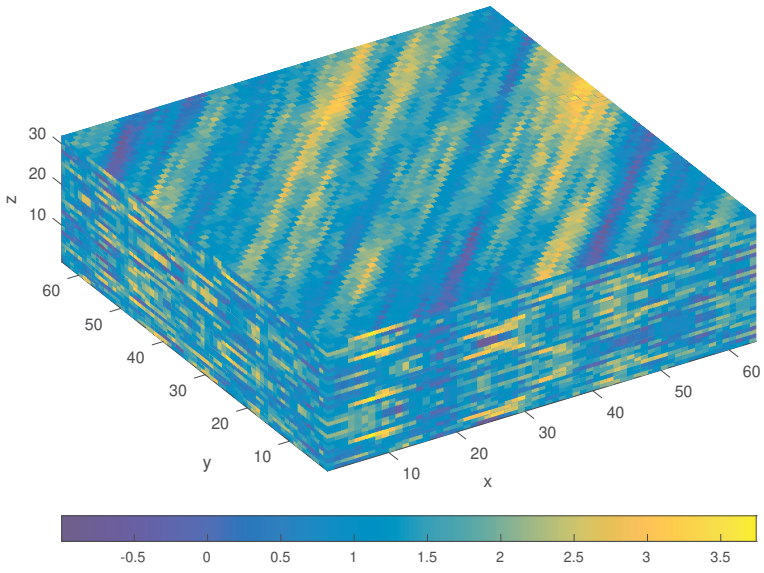


Figure 5.2: A $64 \times 64 \times 32$ log-normal permeability field with long correlation length non-aligned with grid at $\theta = \pi/4$ with x direction.

layering compared to TPFA-ML. However, there was no significant difference in the mean flow errors for boundary conditions 3 and 5.

Two-sample t-test with pooled variance shows a significant difference in relative mean errors between VCMP-L and VCMP-ML for boundary conditions 3 and 4 with $p < 0.05$. We further note significant improvement when refinement is applied in the TPFA method with all $p < 0.05$. Pressure error results are similar, except for a small difference in boundary conditions 3 and 5 when comparing TPFA with VCMP.

Flow Direction	**Flow**	**TPFA-L**	**VCMP-L**	**TPFA-ML**	**VCMP-ML**
1	1512	12.03	2.35	9.81	2.34
2	1568	15.95	1.07	13.81	1.15
3	2313	3.66	0.32	0.92	0.63
4	344	30.67	15.89	22.85	8.55
5	115	25.20	23.06	16.47	15.60

Table 5.7: Flow and mean relative error (in percent) in total flow for long correlation length, non $K-$orthogonal permeability coarse grid blocks, $\Delta x = 8, \Delta y = 8$, and $\Delta z = 8$. Conditions 4 and represent along layering and across layering flow respectively.

Table 5.9 presents stencil statistics of VCMP-L and VCMP-ML for the long correlation

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	9.14	2.59	7.15	2.08
2	9.32	2.31	7.66	2.69
3	2.36	2.65	2.13	1.45
4	10.72	8.27	6.19	5.10
5	7.52	7.67	6.23	6.03

Table 5.8: Mean relative error percentage in pressure for long correlation length, non $K-$orthogonal permeability with high aspect ratio coarse grid blocks, $\Delta x = 8, \Delta y = 8$, and $\Delta z = 8$.

length, not aligned with the grid using uniform block dimensions. VCMP-ML produces slightly larger stencils; however, the mode statistic is the same. Figure 5.3 shows that nodes 3 and 6 are more frequently involved in the computation of the flux. Referring back to Figure 3.2, this highlights the diagonal flow feature in this case.

Method	Max	Mode	Mean	St.Deviation
VCMP-L	6	4	2.84	1.02
VCMP-ML	8	4	3.03	1.33

Table 5.9: Stencil statistics for VCMP-L and VCMP-R for long correlation length, not aligned with the grid using uniform block dimensions.

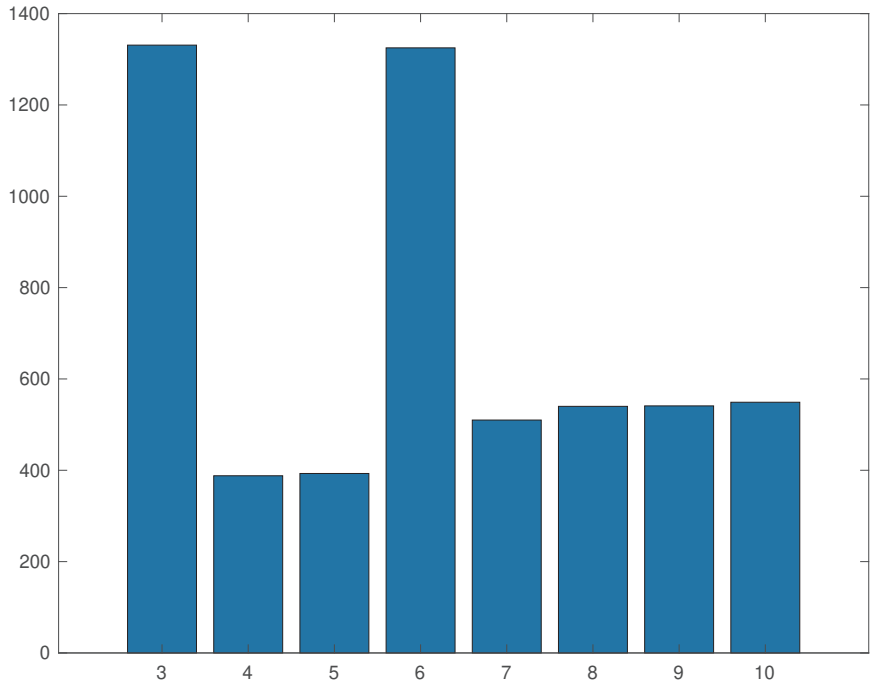


Figure 5.3: Stencil distribution of nodes 3 - 10 for long correlation length, not aligned with the grid using uniform block dimensions.

High-aspect-ratio long correlation non-alignment

Table 5.10 presents mean relative error percent in total flow for long correlations not aligned with grid with coarse grid block dimensions $\Delta x = 16, \Delta y = 8$, and $\Delta z = 4$. Absolute flow errors are listed in Table 5.11. With high-aspect-ratios, as expected, VCMP outperforms TPFA methods in general. VCMP-L and VCMP-ML yield smaller percent errors except for boundary conditions 3 and 5. There differences are significant between VCMP-L and VCMP-ML ($p < 0.05$) for boundary conditions 1 - 4. Refinement appears to reduced accuracy for boundary conditions 5 between VCMP-L and VCMP-ML, however, there was no significant difference ($p = 0.38$).

Table 5.12 shows the L_2 pressure errors for all five boundary conditions on four methods (TPFA-L, TPFA-ML, VCMP-L, and VCMP-ML). Results show that grid refinement does not improve flow or pressure errors across all boundary conditions. Unlike flow errors, VCMP yields only slight improvement in pressure errors, except in the pressure-flow conditions 1 - 3.

Table 5.13 displays the stencil statistics, including mean, standard deviation, max, and mode of the twenty-five realizations for the simulations run on the log-normal permeability

Flow Direction	Flow	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	1510	22.04	2.65	21.23	2.37
2	1567	16.10	2.61	15.53	2.38
3	2306	0.37	1.98	3.16	5.53
4	346	40.83	22.5	34.40	8.72
5	104	11.34	11.86	14.20	14.59

Table 5.10: Mean relative error percentage in total flow for long correlation length, non $K-$orthogonal permeability with high aspect ratio coarse grid blocks, $\Delta x = 16, \Delta y = 8$, and $\Delta z = 4$.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	332.91	40.15	320.71	35.87
2	252.54	40.93	243.56	37.36
3	8.78	45.80	72.87	127.58
4	142.56	77.11	121.44	30.26
5	13.84	15.17	16.35	17.04

Table 5.11: Absolute flow errors, $|Q_f - Q_c|$, in non $K-$orthogonal permeability with high aspect ratio coarse grid blocks, $\Delta x = 16, \Delta y = 8$, and $\Delta z = 4$.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	10.38	4.37	8.99	2.74
2	11.83	2.58	9.76	2.00
3	1.96	2.34	1.22	1.03
4	11.10	8.82	12.33	4.92
5	6.73	7.79	5.11	5.02

Table 5.12: Mean relative error percentage in pressure for long correlation length, non $K-$orthogonal permeability with high aspect ratio coarse grid blocks, $\Delta x = 16, \Delta y = 8$, and $\Delta z = 4$.

field having long correlation length not aligned with the grid and high aspect ratio blocks. Compared to the uniform block size (see Table 5.9), the high-aspect-ratio stencils are slightly smaller with similar variance. The distribution of nonzero weights is nearly identical to Figure 5.3.

5.2 Results of SPE Nearshore Tarbert Formation

Twenty-five realizations of a permeability field from the SPE-10 Tarbert formation representing a Near Shore Environment with dimensions $L_x = 64$ $L_y = 64$ and $L_z = 32$ are

Method	Max	Mode	Mean	St.Deviation
VCMP-L	6	2	3.22	1.26
VCMP-ML	7	3	3.92	1.41

Table 5.13: Stencil statistics ($n = 10$) for VCMP-L and VCMP-ML for long correla-tion length, not aligned with the grid and high-aspect-ratio blocks.

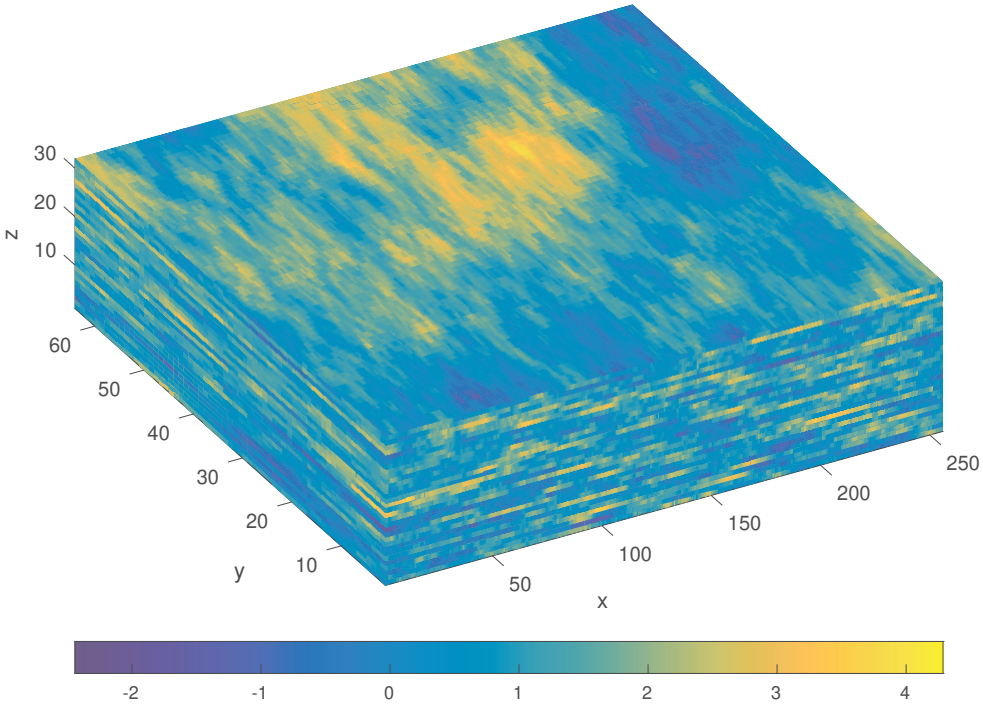


Figure 5.4: Log-normal permeability field from the SPE-10 Tarbert formation (Layers 1 – 32).

examined using our upscaling methods (see Figure 5.4). The uniform coarse grid block sizes in this experiment are $\Delta x = \Delta y = \Delta z = 8$. We conduct the following experiments by applying methods: TPFA-L, TPFA-R, TPFA-ML, VCMP-L, VCMP-R, and VCMP-ML. We do not apply local-global methods to these cases as that method is designed to handle channelize flows encountered in the next Upper Ness Formation included in Section 5.3.

Table 5.14 presents flow errors using the specified upscaling methods listed. Results are reported on six sets of boundary conditions that include corner flow conditions. As noted in Section 5.1, and mentioned throughout, corner-flows often yield exceptionally large errors due to low flow. The absolute errors for these cases (Conditions 4 and 5) are 13.66

and 44.21 respectively. Both TPFA-L and TPFA-ML produce larger relative errors when compared to VCMP-L and VCMP-ML. VCMP with adaptive mesh refinement (VCMP-ML) has less than one-percent relative error when flow is driven in the three coordinate directions. These results agree with the results reported in the literature for two-dimensions.

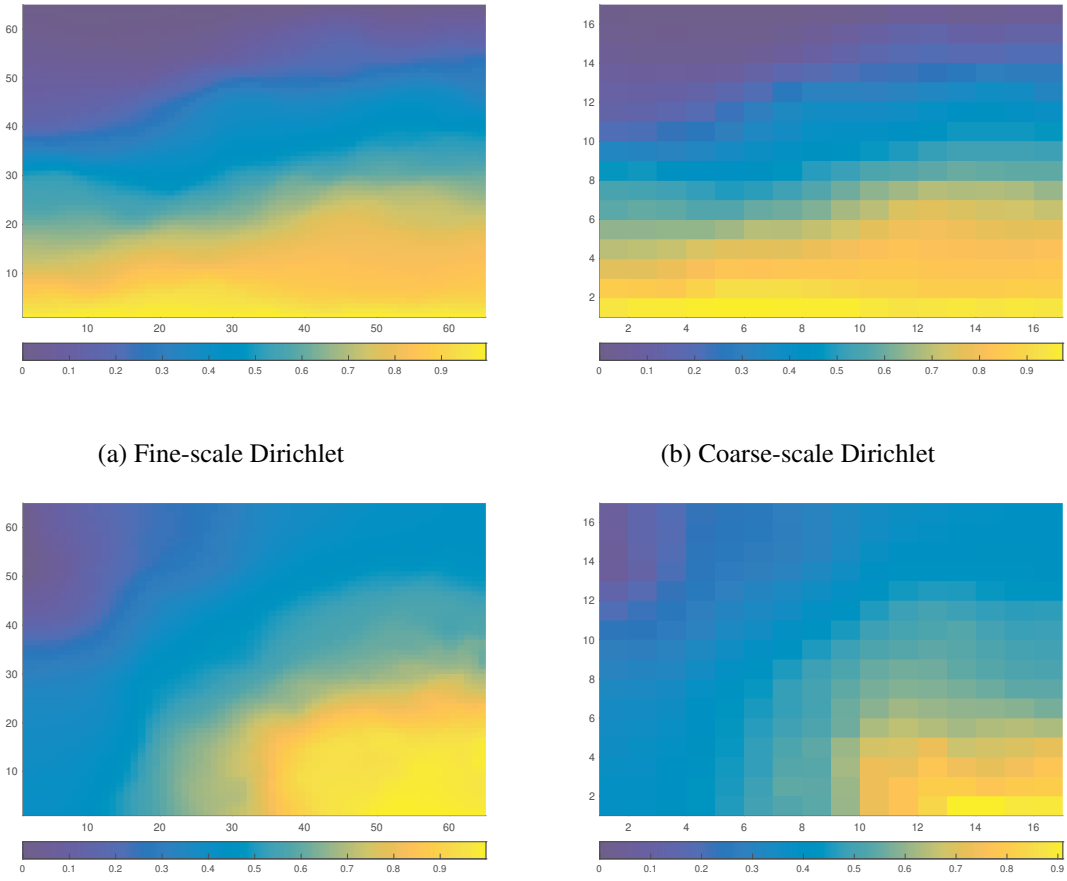
Flow Direction	**TPFA-L**	**VCMP-L**	**TPFA-ML**	**VCMP-ML**
1	2.59	2.10	3.42	0.53
2	1.45	2.27	5.35	0.66
3	4.97	1.07	4.46	0.62
4	20.98	21.01	9.98	8.87
5	47.04	43.85	38.67	34.79

Table 5.14: Mean relative error percentage in total flow in global coarse-scale solutions of 25 realizations from Tarbert Formation (SPE Layers 1 – 35) on selective boundary conditions including corner-to-corner flow and linear Dirichlet conditions. (Uniform block size $\Delta x = \Delta y = \Delta z = 8$)

Average relative pressure errors for this set of realizations and boundary conditions are given in Table 5.15. The pressure errors are in line to the flow errors, with a few notable exceptions. In particular, TPFA with adaptive refinement (TPFA-ML) matches the fine-scale pressure fields better than VCMP-L. However, with grid refinement, we observe superior agreement with the fine-scale pressure field using VCMP-ML, with the exception of Condition 5. Figure 5.5 depicts both fine-scale and coarse-scale pressure fields for generic Dirichlet and corner-to-corner flow boundary conditions.

Flow Direction	**TPFA-L**	**VCMP-L**	**TPFA-ML**	**VCMP-ML**
1	3.17	2.85	2.25	2.08
2	3.71	3.11	2.25	1.70
3	4.38	5.53	2.81	2.54
4	10.51	10.71	3.04	2.08
5	19.46	20.82	9.25	9.53

Table 5.15: Mean relative percentage in pressure error in global coarse-scale solutions of 25 realizations from Tarbert Formation (SPE Layers 1 – 35) on selective boundary conditions including corner-to-corner flow and linear Dirichlet conditions. (Uniform block size $\Delta x = \Delta y = \Delta z = 8$)



(a) Fine-scale Dirichlet

(b) Coarse-scale Dirichlet

(c) Fine-scale corner flow

(d) Coarse-scale corner flow

Figure 5.5: Fine-scale and coarse scale pressure fields for generic Dirichlet and corner flow conditions.

Table 5.16 presents the results comparing local and regional upscaling for TPFA and VCMP methods. A radius $r = 1$ border ring shows improvement on both methods. VCMP-R shows significant improvement in flow accuracy over TPFA-R on nearly all boundary conditions. Improvement is more substantial when TPFA-L is compared to VCMP-R. VCMP-R was approximately five times greater CPU time over VCMP-L as the size of the border regions directly impacts computational requirements.

In Table 5.17, relative mean flow error in TPFA and VCMP local and multi-level methods are compared on high-aspect-ratio coarse grid blocks. Overall, TPFA is competitive with VCMP in these cases. Results show $E_Q < 0.5$ using VCMP-L and $E_Q < 1.1$ for VCMP-ML on the first three boundary conditions. Mesh refinement improves accuracy on both methods except for corner flow Condition 5, where VCMP-L edges VCMP-ML.

Flow Direction	TPFA-L	TPFA-R	VCMP-L	VCMP-R
1	2.59	1.26	2.10	0.22
2	1.45	0.72	2.27	0.31
3	4.97	1.65	1.07	0.39
4	20.98	16.32	21.01	16.70
5	47.04	29.50	43.83	48.40

Table 5.16: Mean percent relative error in total flow from global coarse-scale solutions TPFA vs. VCMP using local and regional ($r = 1$) upscaling. ($\Delta x = \Delta y = \Delta z = 8$)

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	1.57	0.41	2.65	0.90
2	0.91	0.21	4.34	1.07
3	1.48	0.29	0.43	0.49
4	24.62	23.49	8.38	8.92
5	40.05	40.21	18.85	22.58

Table 5.17: Comparison of total flow in global coarse-scale solutions of 25 realizations from Tarbert Formation (SPE Layers 1 – 35) with high-aspect-ratio coarse blocks of size ($\Delta x = 16, \Delta y = 8, \Delta z = 4$).

5.3 Results of SPE Fluival Ness Formation

The uniform coarse grid block sizes in the first set of experiments are $\Delta x = \Delta y = \Delta z = 8$. We conduct the apply the following methods: TPFA-L, TPFA-R, TPFA-ML, VCMP-L, VCMP-R, and VCMP-ML. We also apply local-global methods to these cases as that method is designed to handle channelize flows.

Table 5.18 displays the mean percent relative errors in the total flow rate on uniform coarse grid block ($\Delta x = \Delta y = \Delta z = 8$) for four upscaling methods and five boundary conditions that drive flow (see Table 4.2). VCMP-L and VCMP-ML both outperform corresponding TPFA methods (TPFA-L, TPFA-ML). These are consistent with results reported by Lambers et al. [59] in two-dimensions. As noted above, substantial flow errors are present due to overall low flow through the system based on imposing unit pressure on small corner regions.

We see from these results, that grid refinement generally improves accuracy in both TPFA and VCMP upscaling methods. However, in cases when errors are already low (e.g., Flow direction 1), refinement can decrease accuracy.

VCMP produces compact stencils. The mean stencil width for VCMP-L and VCMP-ML

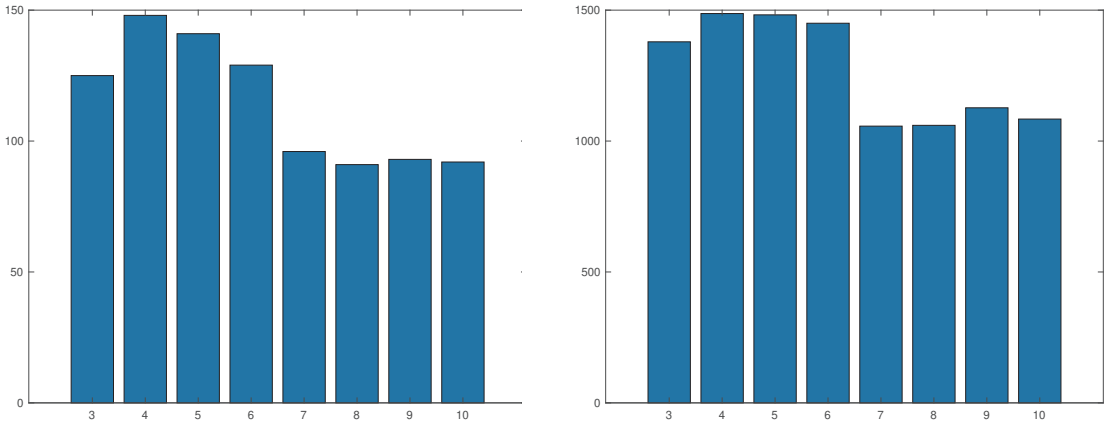
Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	14.17	1.71	19.09	3.88
2	30.39	1.73	22.42	1.42
3	38.53	3.60	19.95	0.65
4	63.51	46.33	55.08	40.41
5	25.36	13.76	10.37	2.97

Table 5.18: Mean percent relative error in total flow obtained from global coarse-scale solution on fluival Upper Ness in SPE Layers 36 – 85 with coarse grid block size $\Delta x = \Delta y = \Delta z = 8$.

is approximately 3.4 (see Table 5.19). Stencil distributions are shown in Figure 5.6. Both distributions are right-skewed, possibly due to natural vertical layering in the permeability field.

Method	Max	Mode	Mean	St.Deviation
VCMP-L	10	3	3.43	1.15
VCMP-ML	10	3	3.40	1.28

Table 5.19: Stencil statistics for VCMP-L and VCMP-ML on SPE Layers 36 – 85 with uniform block size using $n = 10$.



(a) VCMP-L

(b) VCMP-ML

Figure 5.6: Distribution of $N = 10$ stencil weights for channelized layers of SPE.

We repeat the experiment above on the same data set but allow a maximal stencil size of $N = 18$ instead of $N = 10$. Table 5.20 shows slightly improvement in flow errors at the cost of wider stencils (see Table 5.21 and Figure 5.7).

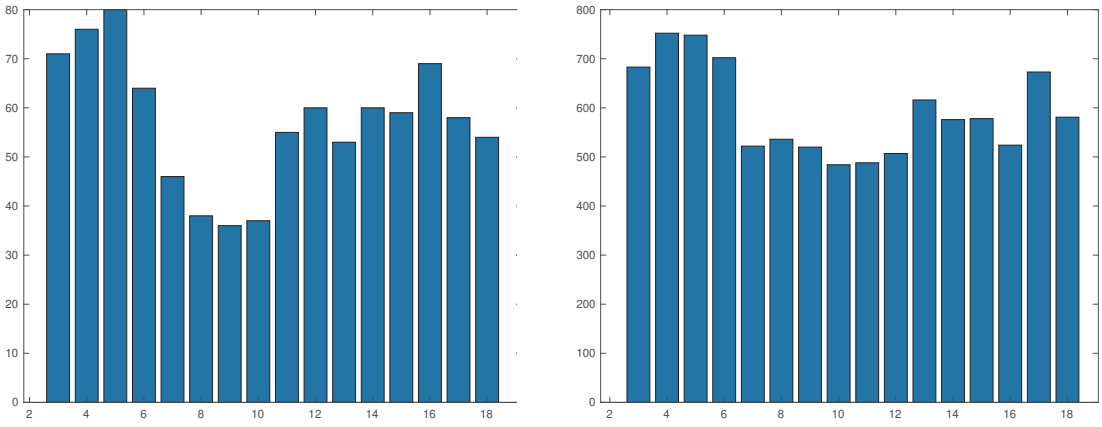
Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	14.17	1.91	19.09	3.42
2	30.39	0.47	22.42	0.31
3	38.53	2.71	19.95	0.05
4	63.51	48.54	55.08	39.57
5	25.36	13.66	10.37	2.79

Table 5.20: Mean flow errors using $N = 18$ node stencils on SPE Layers 36 – 85 with uniform block size $\Delta x = \Delta y = \Delta z = 8$.

When $N = 18$, maximum stencil width increased on VCMP-ML method, but width decreased in VCMP-L from 10 to 9 (see Table 5.21). The mean stencil size for both methods remains around 3.4, however, the mode stencil size decreased in both methods from 3 to 2. Additionally, there was a slight increase in the standard deviation. Figure 5.7 shows the distribution of stencil weights. We observe a bimodal distribution in VCMP-L compared to VCMP-ML, where there is a trend toward uniformity in the distribution with refinement.

Method	Max	Mode	Mean	St.Deviation
VCMP-L	9	2	3.41	1.61
VCMP-ML	13	2	3.43	1.71

Table 5.21: Stencil statistics for VCMP-L and VCMP-R on SPE Layers 36 – 85 with uniform block size using $n = 18$.



(a) VCMP-L

(b) VCMP-ML

Figure 5.7: Distribution of $N = 18$ stencil weights for channelized layers of SPE.

Table 5.22 lists relative errors in the total flow of the realizations comparing local and regional upscaling. The results show that regional upscaling using a radius $r = 1$ border ring greatly improves accuracy when applied to TPFA. These results agree with the literature [78, 79]. However, the comparison between local and regional VCMP produce mixed results.

The results also show VCMP-R yields greater flow accuracy when compared to TPFA-R on all boundary conditions. However, VCMP-L produced better results than TPFA-R. Because regional upscaling uses a larger local domain in the upscaling process, computation time increases. In particular, VCMP-R was approximately ten times greater processing time compared to VCMP-L.

Flow Direction	**TPFA-L**	**TPFA-R**	**VCMP-L**	**VCMP-R**
1	14.17	6.77	1.91	3.07
2	30.39	8.63	0.47	1.82
3	38.53	7.38	2.71	0.61
4	63.51	45.37	48.54	43.42
5	25.36	15.06	13.66	11.89

Table 5.22: Flow errors comparing local and regional ($r = 1$) upscaling methods on channelized layers of SPE-10 with coarse block sizes $\Delta x = \Delta y = \Delta z = 8$.

Pressure errors for this experiment are provided in Table 5.23 for local and regional upscaling. Although regional upscaling improves flow accuracy, pressure solutions are fairly consistent among methods. Table 5.24 displays average total flow and absolute errors in TPFA-L and VCMP-L upscaling on the five boundary conditions. Total flow in corner-to-corner boundary conditions are exceptionally low and account for corresponding flow errors in Table 5.22.

Flow Direction	**TPFA-L**	**TPFA-R**	**VCMP-L**	**VCMP-R**
1	7.80	9.26	9.14	8.26
2	12.71	10.15	19.56	7.99
3	12.43	12.11	11.23	12.67
4	17.59	25.92	18.98	22.36
5	12.80	35.87	17.81	35.89

Table 5.23: Pressure error using local and regional ($r = 1$) upscaling methods on channelized layers of SPE-10 with uniform coarse block sizes $\Delta x = \Delta y = \Delta z = 8$.

Flow Direction	Total Flow	TPFA-L	VCMP-L
1	2758	396	17
2	656	206	119
3	762	294	30
4	121	75	58
5	119	30	16

Table 5.24: Average total flow and absolute errors using local upscaling methods on channelized layers of SPE-10 with uniform coarse block sizes $\Delta x = \Delta y = \Delta z = 8$.

High-aspect-ratio

The results from twenty-five realizations of SPE - 10 fluvial Ness formation with domain $L_x = 256, L_y = 64, L_z = 16$ and high-aspect-ratio coarse grid blocks of size $\Delta_x = 32, \Delta_y = 8$, and $\Delta_z = 4$ are presented in this section. Table 5.25 displays the flow and flow errors from local and multi-level local upscaling methods (TPFA-L, TPFA-ML, VCMP-L, and VCMP-ML). When comparing VCMP-L to TPFA-L, results are mixed. We note general improvement among the results of VCMP to TPFA when refinement is included, except for corner-to-corner conditions where the flow is low. A significant difference exists between the mean flow errors for VCMP-L and VCMP-ML at $p < 0.05$ on boundary conditions 1 - 3. Additionally, we observe VCMP outperforms TPFA in high flow situations and vice-versa. These results are in contrast to the results on uniform grid block sizes found in Table 5.18, where VCMP dramatically increases accuracy. Average pressure errors also show mixed results, as seen in Table 5.26.

Flow Direction	Flow	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	334	42.52	5.36	24.45	10.77
2	2395	13.81	7.32	12.71	3.23
3	10701	5.57	16.13	13.34	1.79
4	104	42.82	20.11	42.93	30.89
5	154	26.07	51.02	24.10	58.15

Table 5.25: Total flow and average percent flow error in channelized layers of SPE-10 with high-aspect-ratio coarse block sizes ($\Delta x = 32, \Delta y = 8, \Delta z = 4$).

Table 5.27 presents results comparing local and regional upscaling methods on a large domain channelized permeability fields with high-aspect-ratio coarse grid blocks. Extending the local domain by adding a border ring has a significant effect on the accuracy of flow results except in condition 5.

Flow Direction	TPFA-L	VCMP-L	TPFA-ML	VCMP-ML
1	5.68	4.29	6.61	4.77
2	11.11	8.67	7.37	10.26
3	12.78	14.21	9.85	21.24
4	33.74	16.09	33.95	52.60
5	15.54	12.24	13.43	9.71

Table 5.26: Average relative pressure error (in percent) in channelized layers of SPE-10 with high-aspect-ratio coarse block sizes ($\Delta x = 32, \Delta y = 8, \Delta z = 4$).

Flow Direction	TPFA-L	TPFA-R	VCMP-L	VCMP-R
1	42.52	4.34	5.36	1.77
2	13.81	4.25	7.32	1.25
3	5.57	1.65	16.13	0.45
4	42.82	20.81	20.11	11.91
5	26.07	67.62	51.02	69.27

Table 5.27: Comparison between local and regional ($r = 1$) upscaling methods on flow error in channelized layers of SPE-10 with high-aspect-ratio coarse block sizes ($\Delta x = 32, \Delta y = 8, \Delta z = 4$).

Local-global on Large domain with High-aspect-ratio

In this section, we present results (see Table 5.28) from the application of a modified local-global approach that incorporated global pressure values at the Dirichlet boundaries but kept no-flow conditions on the other four sides. In high-aspect-ratio blocks on channelized systems, we observe that local-global with adaptive grid refinement (CCAR) is needed to overcome significant flow errors with local-global methods alone. Improvement is most significant in corner-to-corner flows, where we have seen low flow leads to significant inaccuracies. To the best of our knowledge, this is the first implementation of local-global and multi-level local-global techniques with TPFA in three-dimensions.

Table 5.29 provide stencil statistics for VCMP-L, VCMP-R, VCMP-LG and VCMP-MLLG methods. We observe that VCMP produces compact stencils for all four methods. VCMP-R and VCMP-MLLG have the largest maximum stencil width overall; however, the mean stencil widths across all methods are nearly identical, with mean approximately 3.5 and mode 3.

Flow Direction	TPFA-LG	VCMP-LG	TPFA-MLLG	VCMP-MLLG
1	39.97	5.44	25.07	5.77
2	14.43	6.53	12.40	5.96
3	3.38	15.12	14.06	2.18
4	19.26	19.53	28.12	6.32
5	51.86	17.57	39.22	15.92

Table 5.28: Average relative percent flow error in channelized layers of SPE-10 with high-aspect-ratio coarse block sizes ($\Delta x = 32, \Delta y = 8, \Delta z = 4$) using modified local-global procedure.

Method	Max	Mode	Mean	St.Deviation
VCMP-L	7	3	3.55	1.15
VCMP-R	10	3	3.58	1.18
VCMP-LG	8	3	3.58	1.16
VCMP-MLLG	10	3	3.36	1.20

Table 5.29: Stencil statistics for VCMP-L, VCMP-R, VCMP-LG, and VCMP-MLLG on high-aspect-ratio large domain.

5.4 Algorithm Benchmarks

Benchmarks were computed against TPFA-L to test individual algorithm performance. All algorithms were run in parallel on a Mac computer running macOS 10.13.6. The hardware consists of a 2.8 GHz Intel Core i7 with four cores and 16GB RAM. All methods are written in MATLAB and tested on Version 9.8.0.1323502 (R2020a) with Parallel Computing Toolbox (Versions 7.2). The MATLAB Optimization Toolbox (Version 8.5) is used in the upscaling process. Relative times, compared to TPFA-L, for 10 realizations of domain size $64 \times 64 \times 32$ over three boundary conditions are displayed in Figure 5.8.

For reference, the average parallel run time for TPFA-L was 1.72 minutes for ten realizations on a domain $64 \times 64 \times 32$. Run time for VCMP-R is 185.53 minutes and that VCMP-ML requires approximately 12 times longer to execute than TPFA-L. Table 5.30 provides actual times in minutes for all algorithms on two domains with sizes $32 \times 32 \times 32$ and $64 \times 64 \times 32$.

Figure 5.9 depicts weighted errors versus time on several permeability fields. Specifically, for boundary conditions 1 – 3, the weighted errors are computed by an inner product of relative errors associated with each method and the normalized fine-scale flow, $F_{1:3}/||F_{1:3}||$. VCMP methods are in red and TPFA in blue. We see in general, VCMP methods are more accurate, however (naturally) require more computational time. In particular, VCMP-R (red

open circles) requires significant computation whereas similar accuracy can be achieved with mesh refinement, VCMP-ML (red squares) or by using an iterative local-global procedure, VCMP-LG (red closed circle).

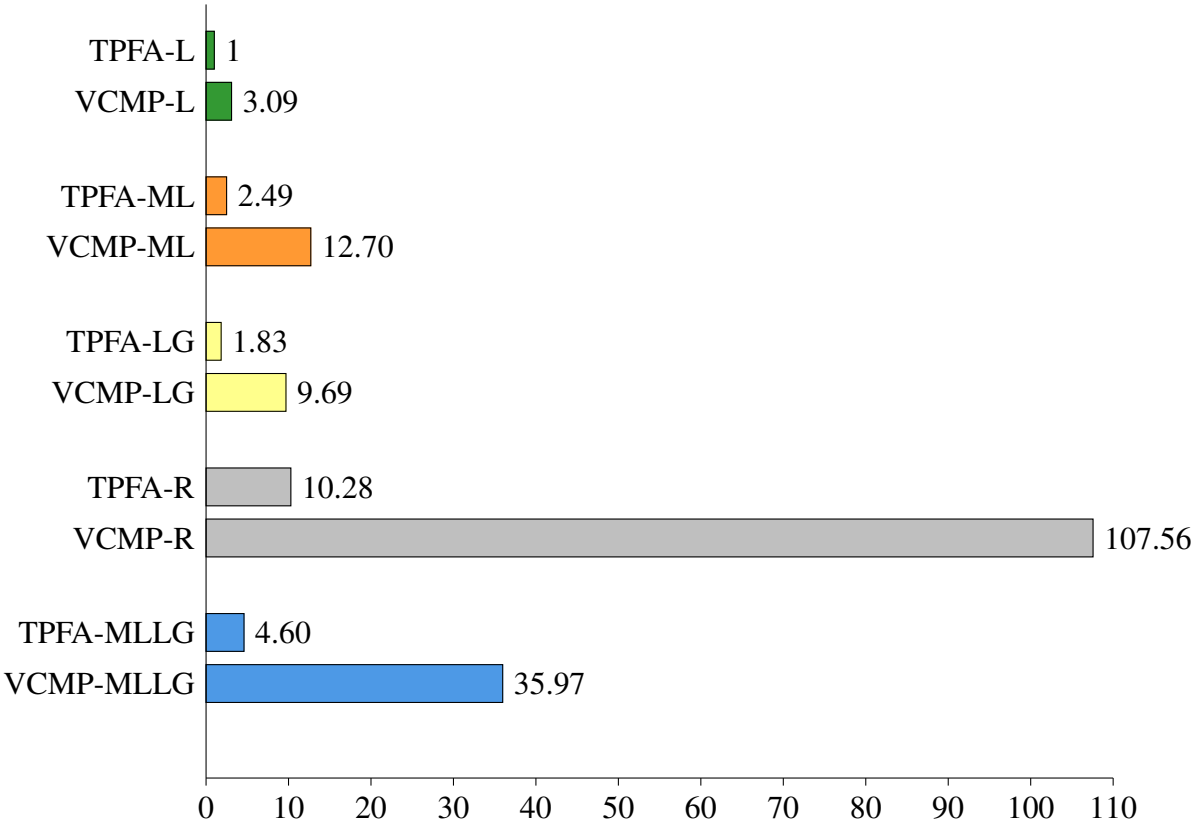
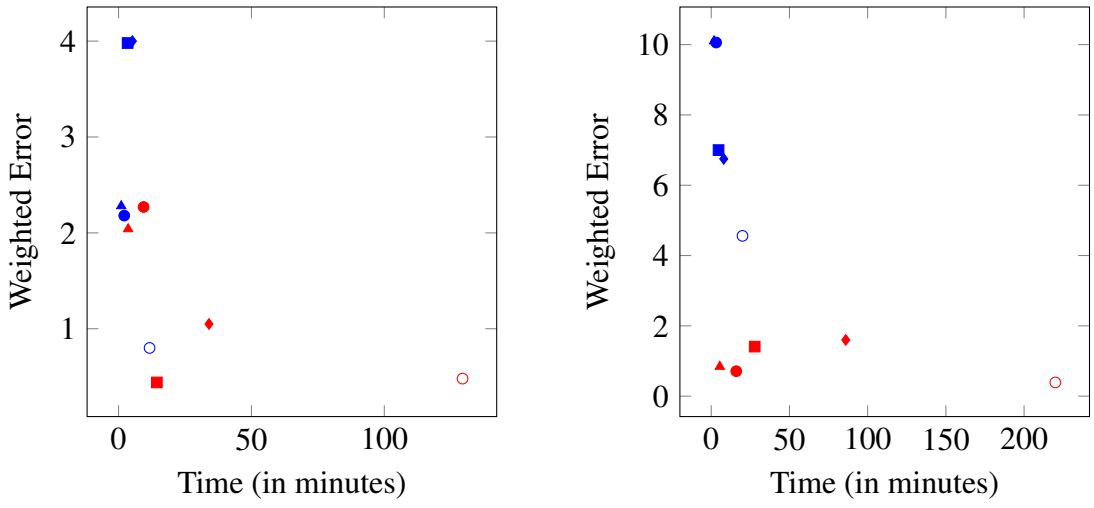


Figure 5.8: Benchmark data relative to TPFA-L (1.72 minutes) using 4 workers for 10 realizations and three boundary conditions. Domain size: $64 \times 64 \times 32$. All algorithms run in parallel.

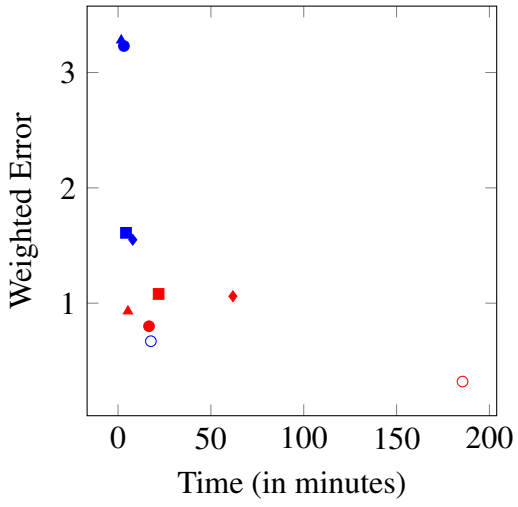
Algorithm	Domain 1	Domain 2
TPFA-L	0.51	1.72
VCMP-L	1.51	5.32
TPFA-ML	2.27	4.29
VCMP-ML	7.48	21.85
TPFA-LG	1.19	3.15
VCMP-LG	4.67	16.67
TPFA-R	2.41	17.68
VCMP-R	21.98	185.53
TPFA-MLLG	5.83	7.92
VCMP-MLLG	23.04	61.86

Table 5.30: Time (in minutes) of each method running 10 realizations on 4 workers in parallel on multiple sized domains. Domain 1 and Domain 2 are $32 \times 32 \times 32$ and $64 \times 64 \times 32$, respectively.

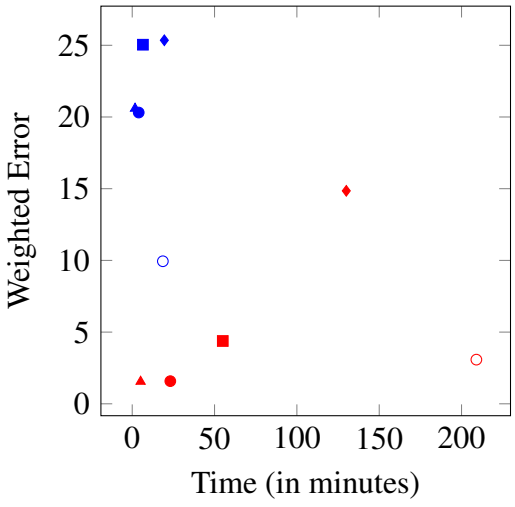


(a) K-orthogonal

(b) Non-K-orthogonal



(c) SPE Tarbert



(d) SPE Upper Ness

Figure 5.9: Weighted error versus time. Legend: TPFA (blue) and VCMP (red); Local methods (triangles); Regional methods (open circle); Multi-level (square); Local-global (closed circle); Multi-level Local-global (diamond).

Chapter 6

CONCLUSIONS AND FUTURE DIRECTIONS

We have generalized VCMP to three-dimensional Cartesian and Cartesian Cell-based Anisotropically Refined (CCAR) grids to perform coarse-scale simulations. We have applied a modified local-global procedure to several methods to incorporate global flow information into the upscaled models to represent large-scale permeability connectivity. In doing so, this represents the first time a local-global technique has been applied to either TPFA or VCMP in three-dimensions. We found that in general, VCMP accurately captures full-tensor anisotropy, which makes coarse-scale flow modeling difficult. It accomplishes this by adapting the stencil to the orientation of the flow instead of the underlying permeability distribution. The stencils can also be used as an additional indicator of grid refinement.

In addition to local VCMP, we included regional upscaling method with our investigations. We combined adaptive mesh refinement and local-global methods to develop a three-dimensional multi-level local-global technique.

The following main conclusions can be drawn from this work:

1. VCMP can accurately resolve three-dimensional flows for a wide variety of boundary conditions, including those not used in the construction of its stencils. The addition of mesh refinement can further increase the quality of the flow approximation. Grid refinement also impacts accuracy of high-aspect-ratio grid blocks.

2. TPFA is competitive with VCMP on K-orthogonal permeability fields in three-dimensions in terms of both accuracy and computational efficiency. However, VCMP outperforms TPFA for non-K-orthogonality. VCMP is more accurate and efficient overall in the other three-dimensional test cases as well.

3. VCMP regional upscaling techniques are substantially better than TPFA methods, and local VCMP, especially in a channelized system as the more extensive region includes more global information. However, computational requirements increase significantly.

4. Multi-level local-global upscaling that couples an iterative local-global technique with adaptive mesh refinement improves accuracy in flow approximations in the most challenging and difficult cases.

5. Although stencils can attain maximum width, on average, VCMP produces highly compact stencils.

6. We have increased the ability to fine-tune VCMP. Specifically, our generalized optimization scheme provides increased flexibility through the addition of multiple regularization parameters. Various assignments of values to these parameters in the objective function can require local flows to be honored more closely or to drive compactification of the stencil by identifying and selecting individual components. The performance of VCMP can be sensitive to the choice of these weights for domains with complicated features such as channels.

7. VCMP can be used to guide adaptivity in such a way as to prevent unnecessary refinement in high-flow areas, thereby achieving high accuracy with fewer cells than in the Cartesian case.

6.1 Future Directions

The three-dimension VCMP upscaling techniques develop in this work can be extended or improved in several ways. Some recommendations for future work are listed below.

1. Examine different weighting schemes. By including multiple weight parameters, our generalized loss equation is highly flexible. In order to fine-tune the solution, a large number of parameters sweeps are necessary. For instance, instead of assigning the pressure gradient to α_i in (3.7), fine-scale face flux from each local solve can be specified instead.

2. Include machine-learning techniques into the upscaling process. For example, using k-fold cross-validation can reduce errors in the optimized transmissibility weights. This approach would involve randomly partitioning the fine-scale pressure values into k groups, or *folds*, of equal size. Then set aside one of the k folds as a test set. Upscale transmissibility by training the model using the pressure values from the remaining $k-1$ sets. This method is repeated k times, and the mean square error is calculated on the held-out set. The process will result in k estimates of the test error, which can be averaged to yield a cross-validation error. Depending on the number k, this process can be computationally expensive; however, it may provide an effective method to reduce error while increasing robustness.

3. Evaluate alternative averaging techniques. In flow-based upscaling, pressure solutions are found on a local (or regional) domain. Averages of these solutions in each coarse

block are then used in the least-squares optimization problem. Different averaging techniques (e.g., arithmetic cell-averaging, point averaging) or other measures of central tendencies (e.g., median) produce different results. Our results are found using the median pressure value in each coarse cell block to reduce the effect of outliers.

Additionally, alternative means should be tested in computing fine-scale transmissibility during local solves. In particular, current methods use a weighted harmonic mean to compute transmissibility in all flow directions; however, it may be more physical to apply this average to vertical flows while the arithmetic average to horizontal flow directions.

4. Implement and test the M-fix (or a modified M-fix) to three-dimensional VCMP as we did not implement the M-fix in this work. It is well-known that multi-point methods, in general, suffer from non-monotonicity. In particular, monotonicity problems occur mostly in cases where permeability is misaligned with the grid and is highly heterogeneous. Ensuring the pressure matrix, A, is an M-matrix is a sufficient (but not necessary) condition to guarantee monotonicity. Lambers et al. [59] developed the M-fix for two-dimensional VCMP methods. The M-fix is a predictor-corrector method that calculates transmissibilities using VCMP and solves the coarse-scale problem. The monotonicity of the solutions is then determined, and if needed, additional constraints are imposed. At that point, the transmissibilities are recomputed.

5. Increase the domain of monotonicity by developing necessary conditions on the matrix A, without requiring and M-matrix, that guarantees monotone solutions. Large scale experiments may provide insights into such conditions.

6. Extend three-dimensional VCMP to unstructured grids. Our implementation of three-dimensional VCMP was limited to rectilinear grids. An obvious extension would be to apply the work done by Chen et al. [18] that implemented two-dimensional VCMP on irregular grids. Because their study was limited to quadrilateral grids and corner-point grids, implementing three-dimensional VCMP to general unstructured grids represents another avenue of research.

7. Develop global three-dimensional VCMP techniques. In Chen et al. [18], a global VCMP method was developed, in which global fine-scale flow information is used in the upscaling process in two-dimensions.

8. Include gravity in the model and consider different implementations in horizontal and vertical directions as the flow will be affected due to sedimentary layering.

9. Examine alternative local boundary conditions. As reported by Chen et al., [19, p. 8], the choice of local boundary conditions is an essential issue in numerical upscaling techniques. For instance, in this study, we modified the local boundary conditions used in the local-global iteration process by prescribing interpolated global pressure values on the inlet and outlet boundaries but maintained no flow on all other boundaries in the upscaling region. Guerillot et al. [44] suggested the use of linear boundary conditions.

10. Generalize VCMP in three-dimensions to multi-phase flow. Accurate single-phase flow results are necessary for a method to apply to multi-phase flow.

Appendix A

MATLAB CODE

VCMP-3D was implemented using MATLAB with the following specifications:
A significant portion of the code to perform the simulations and upscaling process is listed below.
The complete code is available at the url: `https://github.com/jamesquinlan/vcmp3d`.

A.1 Software License Agreement

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

The software is provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or the use or other dealings in the software.

```
% ------------------------------------------------------------------------ %
MATLAB Version: 9.8.0.1323502 (R2020a)
Operating System: Mac OS X  Version: 10.13.6 Build: 17G65
Java Version: Java 1.8.0-202-b08
Optimization Toolbox Version 8.5 (R2020a)
Parallel Computing Toolbox Version 7.2 (R2020a)

% ------------------------------------------------------------------------ %
% REZSIM performs 3D reservoir simulation on given permeability field
%        using specified methods (TPFA vs. MPFA/VCMP) and configurations
%        found in the config.m file.
% ------------------------------------------------------------------------ %
% Configurations
```

```
config;

% Local Variables
nbc = uint8(numel(bcs));
num_methods = uint8(numel(method));

num_mpfa = 0;
for ii = method
    if (ii == 2) || (ii == 4)
        num_mpfa = num_mpfa + 1;
    end
end

% Gross Flow
Q.coarse = zeros(num_methods,nbc,realizations);
Q.fine = zeros(realizations, nbc);

% Flow errors
Eq_abs = zeros(num_methods,nbc,realizations);
Eq = zeros(num_methods,nbc,realizations);
mean_rel_flow_error = zeros(num_methods,nbc);
median_rel_flow_error  = zeros(num_methods,nbc);

% Cell Pressure
Ep = zeros(num_methods,nbc,realizations);
mean_pressure_error = zeros(num_methods,nbc);

% Face Flux
Ef = zeros(num_methods,nbc,realizations);

for r = 1:realizations
    % Load Permeability Data
    K = loadperm(layers, whichPerm, r);
    K = K(1:nx,1:ny,1:nz);
    K_stats{r,1} = permstats(K);

    % Global Fine
    [Pf,Qf,Gf,Tf,Fx,Fy,Fz] = solve_global_finescale(K,bcs,dx,dy,dz);
    Q.fine(r,:) = Qf;

    % Global Coarse
    for m = 1:num_methods
       [Pc,Qc,Gc,Tc,Fc,As,bs,badelements] = solve_global_coarse(K,bcs,method(m),prams);

       % Flow Error bc x method x realization
       Q.coarse(m,:,r) = Qc(3,:);
       Eq_abs(m,:,r) = abs(Qf-Qc(3,:));
       Eq(m,:,r) = 100*abs(Qf-Qc(3,:))./abs(Qf);

       % Pressure Errors
       Pf_avg = zeros(Gc.nicells,nbc);
       for ii = 1:Gc.nicells
          [cx, cy, cz] = find_fine_cells(Gc,Gc.icells(ii),1,1,1);
          for jj=1:nbc
            Pf_avg(ii,jj) = mean(Pf(cx+1,cy+1,cz+1,jj),'all');
          end
       end
       Ep(m,:,r) = 100*vecnorm(Pc(Gc.icells,:) - Pf_avg,2)./vecnorm(Pf_avg);
```

```
        % Face Flux Errors
        F = zeros(Gc.nfaces, nbc);
        for fid = 1:Gc.nfaces
            orientation = Gc.faces(fid,3);
            switch orientation
                case 1
                    F(fid,:) = faceflux(fid,Gc,Fx);
                case 2
                    F(fid,:) = faceflux(fid,Gc,Fy);
                case 3
                    F(fid,:) = faceflux(fid,Gc,Fz);
            end
        end
        Ef(m,:,r) = 100*vecnorm(F - Fc)./vecnorm(F);
    end
end

for m = 1:num_methods
    if realizations > 1
        mean_rel_flow_error(m,:) = mean(squeeze(Eq(m,:,:)),2);
        median_rel_flow_error(m,:) = median(squeeze(Eq(m,:,:)),2);
        mean_pressure_error(m,:) = mean(squeeze(Ep(m,:,:)),2);
    else
        mean_rel_flow_error(m,:) = Eq(m,:);
        median_rel_flow_error(m,:) =  Eq(m,:);
        mean_pressure_error(m,:) = Ep(m,:);
    end
end

% ------------------------------------------------------------------------ %
function [P,Q,G,T,Fx,Fy,Fz] = solve_global_finescale(K,bcs,dx,dy,dz)
% solve_global_finescale = solves global fine scale problem, including:
%                          pressure, (total) flow(s), and grid
% INPUT:
%   K = fine-scale permability
%   bcs = vector of boundary condition specifiers
%   dx, dy, dz = size of fine cell in x, y, z direction, default=[1,1,1]
%
% OUTPUT:
%   P - array of pressure fields, including boundary values, 4th dim is bc
%   Q - array of total flow values, indexed by bc
%   G - Grid structure (fine scale grid structure)
%   Fx = Flux in x
%   Fy = Flux in y
%   Fz = Flux in z

% Local Variables
[xspan,yspan,zspan] = size(K);
nbc = length(bcs);                      % number of bdry conditions
bcs = reshape(bcs,1,nbc);               % reshape to be a row vector
P = zeros(xspan+2,yspan+2,zspan+2,nbc); % initialize pressure per BC
Fx = zeros(xspan+1,yspan,zspan,nbc);    % Fluxes in x direction per bc
Fy = zeros(xspan,yspan+1,zspan,nbc);    % Fluxes in y direction per bc
Fz = zeros(xspan,yspan,zspan+1,nbc);    % Fluxes in z direction per bc
Q = zeros(size(bcs));                   % Q = total flow;

G = Grid(xspan,yspan,zspan,dx,dy,dz);
```

```
T = compute_fine_fluxes(G,K,dx,dy,dz);  %
parfor i = 1:nbc
      [bvals,btype] = bdry_data(G,bcs(i));
      [A,b] = build_system(G,T,bvals,btype);
      [P0,F0] = pressure_solve(A,b,G,T,bvals,btype);
      P(:,:,:,i) = pressure_to_array(P0,G);
      [Fxi,Fyi,Fzi] = flux_to_arrays(F0,G);
      Qx = sum(sum(sum(Fxi([1 end],:,:))))/2;
      Qy = sum(sum(sum(Fyi(:,[1 end],:))))/2;
      Qz = sum(sum(sum(Fzi(:,:,[1 end]))))/2;
      Q(i) = Qx+Qy+Qz;
end

% ------------------------------------------------------------------ %
function [P,Q,G,T,F,As,bs,badelements]=solve_global_coarse(K,bcs,method,parameters)
%
% solve_global_coarse: solves coarse pressure equation with given conditions
%   [P,Q,G]=solve_global_coarse(K,bc) returns coarse pressure, flow, & grid
% INPUT:
%       K =  fine scale permeability data (isotropic, symmetric pos.def)
%       bcs = vector of boundary condition specifiers
%           1: constant-pressure in x, no-flow in y and z
%           2: constant-pressure in y, no-flow in x and z
%           3: constant-pressure in z, no-flow in x and y
%           4-11: pressure=1 at a corner, and decays to zero away from corner
%       method  =1 is TPFA with Cartesian grid
%               =2 is VCMP with Cartesian grid
%               =3 is TPFA with CCAR grid
%               =4 is VCMP with CCAR grid
%       parameters = structure containing basic information
%           maxcells: allow max grid refinement based on permeability
%           dx, dy, dz: dimensions of fine cell in x, y, z dimensions
% OUTPUT:
%       P = array of pressure fields, indexed by cells in G.cells and bc
%       Q = array of total flow values, indexed by bc
%       G = coarse grid structure
%       T = Upscaled Transmissibility Stencils (cells, weights, fineflux)
%       As= System of equations (Matrix A)
%       bs= RHS of system of equations (Columns vector b)
%       badelements = Bad elements in matrix (e.g., a_{ii} < 0)

%  Settings (Replace this with prams, configs/settings)
    switch method %
        case 1
            tpfa = true;
            ccar = false;
        case 2
            tpfa = false;
            ccar = false;
        case 3
            tpfa = true;
            ccar = true;
        case 4
            tpfa = false;
            ccar = true;
        otherwise
            tpfa = true;
            ccar = false;
```

```
    end

% Load parameters
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
dx = parameters.fine(1);
dy = parameters.fine(2);
dz = parameters.fine(3);

% Construct grid (refine based on high permeability)
if ccar
    G=initial_grid(K,dx,dy,dz,parameters);
else
    G=Grid(parameters.coarsegrid(1),...
            parameters.coarsegrid(2),...
            parameters.coarsegrid(3),...
            parameters.coarsegrid(4),...
            parameters.coarsegrid(5),...
            parameters.coarsegrid(6));
end

%  Initialize local variables
nbc=length(bcs);
Q = zeros(3,nbc);
As = cell(nbc,1);
bs = cell(nbc,1);
badelements = zeros(nbc,4);
Q0 = [0 0 0];
consistent_solution = false;
ref_percent = 100;

% Local-Global iteration
while ~consistent_solution
    % UPSCALE
    % ++++++++++++++++++++++++++++++++++++++++++++++++++++
    if ccar  % Cartesian Cell-based Anisotropic Refinement
        while true  % Until NO more refinements
            [T,refinements] = upscale(G,K,tpfa,marked,parameters);

            % Calculuate flows (for each bc)
            F0=zeros(G.nfaces,3);  % Initialize Face Flux
            for i=1:3
                [bvals,btype]=bdry_data(G,bcs(i));
                [A,b]=build_system(G,T,bvals,btype); % 1
                [~,F0(:,i)]=pressure_solve(A,b,G,T,bvals,btype); % 2
            end

            % Integrate Flow in each direction
            for i=1:3
                leftfaces=G.faces(G.cells(G.faces(:,1),3+i)==0,4);
                Q1=sum(F0(leftfaces,i));
                rightfaces=G.faces(G.cells(G.faces(:,2),3+i)==0,4);
                Q2=sum(F0(rightfaces,i));
                Q0(i)=(Q1+Q2)/2;
            end

            % Calculate FAIRSHARE per face, refine accordingly
            cnt_more_fair_share = 0;
            for i = 1:G.nfaces
```

```
            more_than_fair = fairshare(G,T,F0,Q0,i);
            reason = 1;
            if more_than_fair
                refinements = update_refines(refinements,G,i,reason);
                cnt_more_fair_share = cnt_more_fair_share + 1;
            end
        end

        % Refine Grid
        noldcells = G.nicells; % BREAK criteria
        if ref_percent > 1
            refine_cells = refinements(refinements(:,3) > 0,2);
            refine_dims = refinements(refinements(:,3) > 0,3);
            G = refine_grid(G,refine_cells,refine_dims,dx,dy,dz,ccar);
        end

        % Stop Refinement
        if G.nicells==noldcells
            break;
        end
    end
else
    [T , ~ ] = upscale(G,K,tpfa,marked,parameters);
end

P = zeros(G.ncells,max(3,nbc));
Fc = zeros(G.nfaces,max(3,nbc));

% Solve GLOBAL COARSE Pressure and Flow / principle component
for i=1:3

    % Get boundary values and type
    [bvals,btype]=bdry_data(G,bcs(i));

    % Build system
    [A,b]=build_system(G,T,bvals,btype);

    % Solve PRESSURE and FLOW
    [P(:,i),F]=pressure_solve(A,b,G,T,bvals,btype);
    G.cells(:,15+i)=P(:,i);
    Fc(:,i) = F;

    % Calculate (global COARSE) flow per i-th boundary condition
    Q(3,i) = sum(F(G.faces(G.faces(:,5)>0,4)))/2;

    % M-matrix ==> monotone, (monotone =/=> M-matrix)
    monotone=checkmono(G.cells(:,15+i),G);
end

% If Local-Global interation, then check for consistent flows
if parameters.local_global == 1
    Q(1,:) = Q(3,:);
    Q(2,:) = Q(3,:);
    parameters.local_global = parameters.local_global + 1;
elseif (parameters.local_global > 1) && (parameters.local_global < 5)
    dQ=flowcon(Q);

    % Determine if solution is consistent
```

```
        if (dQ < 0.001)
            consistent_solution=true;
        else
            Q(2,:) = Q(3,:);
            Q(3,:) = 0;
            parameters.local_global = parameters.local_global + 1;
        end
    else % = 0
        consistent_solution = true;
    end
end

% Solve All Other Boundary Conditions
other_bcs = setdiff(bcs,[1,2,3]);

for i = 1:numel(other_bcs)
    [bvals,btype] = bdry_data(G,other_bcs(i));
    [A, b] = build_system(G,T,bvals,btype);
    [P(:,i + 3),F] = pressure_solve(A,b,G,T,bvals,btype);
    Fc(:,i+3) = F;
    Q(3,i + 3) = sum(F(G.faces(G.faces(:,5)>0,4)))/2;
    As{i+3}=A;bs{i+3}=b;
    badelements(i+3,:) = bad_elements(A,G,T);
end
F = Fc;
end


% ------------------------------------------------------------------------ %
function [T,refinements]=upscale(G,K,tpfa,marked,prams)
% upscale(G,K) computes upscaled transmissibility stencils for fluxes
%         across all faces of grid G and return a list of cells flagged for refinement.
%
% T = upscale(G,K) - Generalize Stencils used in all simulations.
%
% Input (Summary):
%    G class (cells(centers, dimesnions,..), faces, etc.
%    K = fine permeability field (nx x ny x nz)
%    tpfa = true to use two-point flux or false for VCMP
%    marked = faces that are redone after applying the mfix
%    prams = upscaling parameters set in config.m
%
% Output:
%       1. T = structure containing stencils (Flux Stencils)
%            T.cells = G.nfaces-by-10 matrix
%               ith row lists cell indices from G.cells
%               involved in stencil for ith face.  The indices of T.cells
%               indicate relative position in stencil of cell indices.
%
%            T.weights - G.nfaces-by-10 matrix
%               ith row lists weights of cells involved in stencil for ith face
%               10 = # cells involved in ith face
%
%       2. refinements = List of cell refinements [  RefineID   CellID   Yes/No   ]
%                        refine_dims = bitmask to indicate dimensions which cells are refined
%
% Upscaling Workflow:
% For each coarse scale face:
```

```
%     1. Create a local fine region that includes the
%        cells surrounding the selected face
%     2. For each element in standard basis
%        a. Solve for fine flux across the face using TPFA, f^(i)
%        b. Solve for local fine scale pressure values, then
%           average to use as the coarse pressure vector p^(i)
%     3. OPTIMIZE the transmissibility weights t_i (i=1, 2,...,10)
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

% Declare Local (Broadcast and Slice Variables)
ntotcells = prams.ntotcells;
cells = zeros(G.nfaces,ntotcells);
weights = zeros(G.nfaces,ntotcells);
refinements = [(1:G.nicells)', G.icells, zeros(G.nicells,1)];
markedFaces = [(1:G.nfaces)' zeros(G.nfaces,1)];
signs = kron(ones(1,ntotcells/2),[-1 1]);
Aleq0 = diag(-signs);
bleq0 = -eps*ones(ntotcells,1);
dx = G.fine(1);
dy = G.fine(2);
dz = G.fine(3);
tikhonov = prams.tikhonov;
delta = prams.delta;
tau = prams.tau;
padding = prams.padding;
weighting = prams.weighting;
local_global = prams.local_global;
significance_level = prams.negligible;
cell_averaging = prams.cell_averaging;
S = uint8(1:ntotcells);
n_dims = uint8(ndims(K));
coarsefluxes = zeros(G.nfaces,3);
x=uint8(1);y = uint8(2);z=uint8(3);
vcmp = ~tpfa;

% Local-Global Interpolation
if local_global > 1 % if > 0
    Sx = scatteredInterpolant(G.cells(G.realcells,1:3),G.cells(G.realcells,16));
    Sy = scatteredInterpolant(G.cells(G.realcells,1:3),G.cells(G.realcells,17));
    Sz = scatteredInterpolant(G.cells(G.realcells,1:3),G.cells(G.realcells,18));
end

% ++++ UPSCALE ++++
% ------------------------------------------------
parfor fid = 1:G.nfaces
    % Reset Local Variables
    facebounds = [Inf eps repmat([Inf 0],1,ntotcells-1)];
    LB0 = facebounds(1:ntotcells).*signs;
    UB0 = facebounds(2:ntotcells+1).*signs;
    volumes = zeros(ntotcells,1);
    pgrads = zeros(3,2,3);
    psi = zeros(3,ntotcells);
    grads = zeros(3);
    faceflux = zeros(3,1);
    cellflux = zeros(3,ntotcells);
    alphas = zeros(1,3);
```

```
% Cell and Face Information
% -------------------------------------------------
% Get cells numbers on either side of face
i1 = uint32(G.faces(fid,1));
i2 = uint32(G.faces(fid,2));

% Get orientation of face & dimensions perp _|_ to face
orientation = uint8(G.faces(fid,3));
otherinds = uint8(1:3);otherinds(orientation)=[];

% Determine WHICHCELL a face (mostly) belong
faceareas = prod(G.cells([ i1,i2 ],otherinds+3),2);
[minarea,whichcell] = min(faceareas);

% Stencil Cell (identification and pruning)
if tpfa || G.faces(fid,5) > 0 % G.cells(i1,14)~=0 || G.cells(i2,14)~=0
    fluxcells = [G.faces(fid,1), G.faces(fid,2), zeros(1,ntotcells-2)];
    validcells = [ 1 2 ];
else % MPFA and Interior Cell
    fluxcells = getfluxcells(G, fid, whichcell, otherinds, ntotcells);
    fluxcells = fluxcells(1:ntotcells);
    fluxcells(S(UB0<LB0+1e-10)) = 0;

    validcells = find(fluxcells>0);
    validcells = setdiff(validcells,ntotcells+1:18);   %
end
num_valid_cells = length(validcells);

% Volume of each cell involved
volumes(validcells) = prod(G.cells(fluxcells(validcells),4:6),2);
total_volume = sum(volumes(validcells)); % Volume of each validcell

% Build local (or extended local) region, make Grid, compute fluxes
Di = mklocaldomain(G,fluxcells,orientation,dx,dy,dz,padding);
Ki = K(Di.xfinemin:Di.xfinemax,Di.yfinemin:Di.yfinemax,Di.zfinemin:Di.zfinemax);
Gi = Grid(Di.xspan,Di.yspan,Di.zspan,dx,dy,dz); % Make local fine grid
Ti = compute_fine_fluxes(Gi,Ki,dx,dy,dz);

% Solve Pressure & Calculate Flow ( Local solves )
for bc = randperm(3)   % [x,y,z];
    if local_global < 2
        [bvals, btype] = Gi.bvals(bc);
    else
        % Dirichlet (Local-global)
        [default_bvals, btype] = Gi.bvals(bc);

        % Values (ordered by face)
        bcenters = Gi.cells(Gi.bcells,1:3);
        bcenters = bcenters + [Di.xfinemin-1, Di.yfinemin-1, Di.zfinemin-1];

        switch bc % Interpolate
            case 1
                bvals = Sx(bcenters);
            case 2
                bvals = Sy(bcenters);
            case 3
                bvals = Sz(bcenters);
        end
```

```
    bvals = bvals.*(default_bvals.*btype);
end %-local_global < 2

% Setup and Solve Local Pressure Equations
[Ai,bi] = build_system(Gi,Ti,bvals,btype);
[P0,F0] = pressure_solve(Ai,bi,Gi,Ti,bvals,btype);
[Fx,Fy,Fz] = flux_to_arrays(F0,Gi);
P = pressure_to_array(P0,Gi);

% Calculate Avg. Cell Pressures & Pressure Gradient
for j = 1:num_valid_cells
    [xfinej,yfinej,zfinej] = find_fine_cells(G,fluxcells(validcells(j)),dx,dy,dz);

    % Localize Cell:  Fine-scale local grid of coarse cell j
    xfinej = xfinej-Di.xfinemin+1;
    yfinej = yfinej-Di.yfinemin+1;
    zfinej = zfinej-Di.zfinemin+1;

    % Avg. Pressure of course cell j
    psi(bc,validcells(j)) = pressure_average(P,xfinej,yfinej,zfinej,cell_averaging);

    % Cell Flux of course cell j
    if G.faces(fid,5)==0
        switch bc % Flux per coarse cell
            case 1
                cellflux(1,validcells(j)) = mean(Fx(xfinej,yfinej,zfinej),'all');
            case 2
                cellflux(2,validcells(j)) = mean(Fy(xfinej,yfinej,zfinej),'all');
            case 3
                cellflux(3,validcells(j)) = mean(Fz(xfinej,yfinej,zfinej),'all');
        end
    end

    % Avg (relative) pressure gradient / cell [px,py,pz]'
    pgrads(bc,j,1) = mean(diff(P(xfinej+1,yfinej+1,zfinej+1),1,1)/G.fine(1),'all')...
        *volumes(validcells(j));
    pgrads(bc,j,2) = mean(diff(P(xfinej+1,yfinej+1,zfinej+1),1,2)/G.fine(2),'all')...
        *volumes(validcells(j));
    pgrads(bc,j,3) = mean(diff(P(xfinej+1,yfinej+1,zfinej+1),1,3)/G.fine(3),'all')...
        *volumes(validcells(j));
    pgrads(isnan(pgrads)) = 0;
end

% Weighted average of pressure gradient (normalized)
grads(bc,[x,y,z]) = sum(pgrads(bc,1:num_valid_cells,[x y z]),2)./total_volume;

% (Local) fine-scale face flux, f^(i)
switch orientation
    case 1
        if whichcell == 1
            faceflux(bc) = sum(Fx(Di.xfine1(end)+1,Di.yfine1,Di.zfine1),'all');
        else
            faceflux(bc) = sum(Fx(Di.xfine2(1),Di.yfine2,Di.zfine2), 'all');
        end
    case 2
        if whichcell == 1
            faceflux(bc) = sum(Fy(Di.xfine1,Di.yfine1(end)+1,Di.zfine1),'all');
        else
```

```
                faceflux(bc) = sum(Fy(Di.xfine2,Di.yfine2(1),Di.zfine2),'all');
            end
        case 3
            if whichcell == 1
                faceflux(bc) = sum(Fz(Di.xfine1,Di.yfine1,Di.zfine1(end)+1),'all');
            else
                faceflux(bc) = sum(Fz(Di.xfine2,Di.yfine2,Di.zfine2(1)),'all');
            end
    end
end

% ************************************************
% Results: Pressure and flux
% ************************************************
psi = psi(:,validcells);
coarsefluxes(fid,:) = faceflux;
% ------------------------------------------------

% ************************************************
% Upscale transmissibilities
% ************************************************
t = (faceflux./(psi(:,1) - psi(:,2)))';
t = t(G.faces(fid,3));
t = [-abs(t), abs(t)];

if vcmp && G.faces(fid,5)==0
    % Weighting Method (alpha or beta)
   norm_of_gradients=sqrt(sum(grads.^2,2));
   pGradient=grads./norm_of_gradients;
   facenorm = diff(G.cells(fluxcells(1:2),1:3));
   facenorm = facenorm'/norm(facenorm);
   alphas(1:3)=abs(pGradient*facenorm);
end

    betas = eye(num_valid_cells); % diag(ones(1,num_valid_cells));
    betas(1,1) = 0; betas(2,2) = 0;

    % Bounds on the Solution
    LB0([false false LB0(3:end) == -Inf]) = 1.5 * min(Ti.weights(:,1));
    UB0([false false UB0(3:end) == Inf])  = 1.5 * max(Ti.weights(:,2));

    X0=t;

    % Constraint Equations: t1+t2+... = 0 && t_{2k+1} =< 0, t_{2k} >= 0
    [A,b,Aeq,beq,LB,UB] = lsq_constraints(Aleq0, bleq0, LB0, UB0, validcells);

    [C,d] = lsq_objective(alphas, betas, psi, faceflux, num_valid_cells, tikhonov);
    A(end,:) = delta.*signs(validcells);
    A(end,1:2) = 0;

    b(end) = tau*mean(abs(t));

    [t,~,~,exitflag,~] = lsqlin(C,d,A,b,Aeq,beq,LB,UB,X0,lsqoptions);

    % Remove cells with negligible weights
    significant_weights = abs(t(1:end)) > significance_level*max(abs(t)); %
    significant_weights(1:2)=true;
```

```
    validcells = validcells(significant_weights);
    t = t(significant_weights);% try commenting out
    psi = psi(:,significant_weights);

    % Update
    if num_valid_cells ~=length(validcells)
        num_valid_cells = length(validcells);
        betas = betas(significant_weights,significant_weights);
        % changed next line from 0 to tikhonov
        [C,d] = lsq_objective(alphas, betas, psi, faceflux, num_valid_cells, tikhonov);
        [A, b, Aeq, beq, LB, UB, ~] = lsq_constraints(Aleq0, bleq0, LB0, UB0, validcells);
        [t,~,~,exitflag, ~] = lsqlin(C,d,A,b,Aeq,beq,LB,UB,t,lsqoptions);

        significant_weights = abs(t)>0.1;
        significant_weights(1:2) = true;
        validcells = validcells(significant_weights);
        t = t(significant_weights);
        epsilon = sum(t);
        if epsilon > 0
           t(1) = t(1) - epsilon;
        else
           t(2) = t(2) - epsilon;
        end
    end
end %-if vcmp

% Save (local) Stencil Cells and Weights
newvec=zeros(1,ntotcells);
for ii = validcells
    newvec(ii) = fluxcells(ii);
end
cells(fid,:) = newvec;
weights(fid,:)=weightsAroundFace(ntotcells,validcells, t);

% Check for Negative Transmissibility
if (t(1)>0 || t(2)<0) || exitflag ~= 1
    if whichcell==1
        switch orientation
            case 1  % Left cell x-face
                k1=Ki(Di.xfine1(end),Di.yfine1,Di.zfine1);
                if Di.xfine1(end)<size(Ki,1)
                    k2=Ki(Di.xfine1(end)+1,Di.yfine1,Di.zfine1);
                else
                     k2=k1;
                end
            case 2  % Left cell x-face
                k1=Ki(Di.xfine1,Di.yfine1(end),Di.zfine1);
                if Di.yfine1(end)<size(Ki,2)
                    k2=Ki(Di.xfine1,Di.yfine1(end)+1,Di.zfine1);
                else
                    k2=k1;
                end
            case 3
                k1=Ki(Di.xfine1,Di.yfine1,Di.zfine1(end));
                if Di.zfine1(end)<size(Ki,3)
                    k2=Ki(Di.xfine1,Di.yfine1,Di.zfine1(end)+1);
                else
                    k2=k1;
```

```
                end
            end
        else
            switch orientation
                case 1
                    k2=Ki(Di.xfine2(1),Di.yfine2,Di.zfine2);
                    if Di.xfine2(1)>1
                            k1=Ki(Di.xfine2(1)-1,Di.yfine2,Di.zfine2);
                    else
                            k1=k2;
                    end
                case 2
                    k2=Ki(Di.xfine2,Di.yfine2(1),Di.zfine2);
                    if Di.yfine2(1)>1
                        k1=Ki(Di.xfine2,Di.yfine2(1)-1,Di.zfine2);
                    else
                        k1=k2;
                    end
                case 3
                    k2=Ki(Di.xfine2,Di.yfine2,Di.zfine2(1));
                    if Di.zfine2(1)>1
                        k1=Ki(Di.xfine2,Di.yfine2,Di.zfine2(1)-1);
                    else
                        k1=k2;
                    end
            end
        end
        gap=abs(diff(G.cells(fluxcells(1:2),orientation)));
        Tback=mean(mean(2*k1.*k2./(k1+k2)))*min(faceareas)/gap;
         Tback = max(Tback,eps);

        % Update Cells and Weights for Neg. Flux
        weights(fid,:)=[ -Tback Tback zeros(1,ntotcells-2) ];
        cells(fid,:)=[fluxcells(1:2) zeros(1,ntotcells-2) ];

        % Mark face for potential refinment
        markedFaces(fid,2)=1

    end
end

% Update refinement array
updateThese=markedFaces(markedFaces(:,2)>0, 1);
if ~isempty(updateThese)
    for i=updateThese'
        refinements=update_refines(refinements,G,i,3); %
    end
end

if tpfa
    cells=cells(:,1:2);
    weights=weights(:,1:2);
end
T=Stencil(cells,weights,coarsefluxes);
```

BIBLIOGRAPHY

[1] Jorg E Aarnes. On the use of a mixed multiscale finite element method for greater flexibility and increased speed or improved accuracy in reservoir simulation. *Multiscale Modeling & Simulation*, 2(3):421–439, 2004.

[2] I Aavatsmark and GT Eigestad. Numerical convergence of the MPFA O-method and U-method for general quadrilateral grids. *International journal for numerical methods in fluids*, 51(9-10):939–961, 2006.

[3] I Aavatsmark, GT Eigestad, BT Mallison, and JM Nordbotten. A compact multipoint flux approximation method with improved robustness. *Numerical Methods for Partial Differential Equations*, 24(5):1329–1360, 2008.

[4] Ivar Aavatsmark. An introduction to multipoint flux approximations for quadrilateral grids. *Computational Geosciences*, 6(3-4):405–432, 2002.

[5] Ivar Aavatsmark. Multipoint flux approximation methods for quadrilateral grids. In *9th International forum on reservoir simulation, Abu Dhabi*, pages 9–13, 2007.

[6] Ivar Aavatsmark, T Barkve, Ø Bøe, and T Mannseth. Discretization on non-orthogonal, quadrilateral grids for inhomogeneous anisotropic media. *Journal of computational physics*, 127(1):2–14, 1996.

[7] Ivar Aavatsmark, Tor Barkve, O Bøe, and Trond Mannseth. Discretization on unstructured grids for inhomogeneous, anisotropic media. part i: Derivation of the methods. *SIAM Journal on Scientific Computing*, 19(5):1700–1716, 1998.

[8] Ivar Aavatsmark, Tor Barkve, Trond Mannseth, et al. Control-volume discretization methods for 3d quadrilateral grids in inhomogeneous, anisotropic reservoirs. *SPE Journal*, 3(02):146–154, 1998.

[9] Khalid Aziz and Antonin Settari. *Petroleum reservoir simulation*. Chapman & Hall, 1979.

[10] Alain Bourgeat. Homogenized behavior of two-phase flows in naturally fractured reservoirs with uniform fractures distribution. *Computer Methods in Applied Mechanics and Engineering*, 47(1):205–216, 1984.

[11] Jef Caers. *Petroleum geostatistics*. Society of Petroleum Engineers Richardson, 2005.

[12] WT Cardwell Jr, RL Parsons, et al. Average permeabilities of heterogeneous oil sands. *Transactions of the AIME*, 160(01):34–42, 1945.

[13] Philip Crosbie Carman. Fluid flow through granular beds. *Trans. Inst. Chem. Eng.*, 15:150–166, 1937.

[14] Philip Crosbie Carman. Flow of gases through porous media. 1956.

[15] A Castellini, Michael G Edwards, and Louis J Durlofsky. Flow based modules for grid generation in two and three dimensions. In *ECMOR VII-7th European Conference on the Mathematics of Oil Recovery*, 2000.

[16] Qian-Yong Chen, Jing Wan, Yahan Yang, and Rick T Mifflin. Enriched multi-point flux approximation for general grids. *Journal of Computational Physics*, 227(3):1701–1721, 2008.

[17] Tao Chen, Christoph Clauser, Gabriele Marquart, Karen Willbrand, and Darius Mottaghy. A new upscaling method for fractured porous media. *Advances in water resources*, 80:60–68, 2015.

[18] Tianhong Chen, Margot G Gerritsen, James V Lambers, and Louis J Durlofsky. Global variable compact multipoint methods for accurate upscaling with full-tensor effects. *Computational Geosciences*, 14(1):65–81, 2010.

[19] Y. Chen, Louis J. Durlofsky, M. Gerritsen, and Xian-Huan-H. Wen. A coupled local–global upscaling approach for simulating flow in highly heterogeneous formations. *Advances in Water Resources Advances in Water Resources*, 26(10):1041–1060, 2003.

[20] Yuguang Chen and Louis J Durlofsky. Adaptive local–global upscaling for general flow scenarios in heterogeneous formations. *Transport in porous Media*, 62(2):157–185, 2006.

[21] Yuguang Chen, Yan Li, and Yalchin Efendiev. Time-of-flight (tof)-based two-phase upscaling for subsurface flow and transport. *Advances in water resources*, 54:119–132, 2013.

[22] Yuguang Chen, Bradley T Mallison, and Louis J Durlofsky. Nonlinear two-point flux approximation for modeling full-tensor effects in subsurface flow simulations. *Computational Geosciences*, 12(3):317–335, 2008.

[23] Zhiming Chen and Thomas Hou. A mixed multiscale finite element method for elliptic problems with oscillating coefficients. *Mathematics of Computation*, 72(242):541–576, 2003.

[24] Tuck C Choy. *Effective medium theory: principles and applications*, volume 165. Oxford University Press, 2015.

[25] M. A. Christie. Upscaling for reservoir simulation. *Journal of Petroleum Technology*, 48(11):1004–1010, 1996.

[26] MA Christie, MJ Blunt, et al. Tenth SPE comparative solution project: A comparison of upscaling techniques. In *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers, 2001.

[27] William J Coirier and Kenneth G Powell. Solution-adaptive cartesian cell approach for viscous and inviscid flows. *AIAA journal*, 34(5):938–945, 1996.

[28] Henry Darcy. Les fontaines publiques de la ville de dijon, dalmont. *Paris: Dalmont*, 1856.

[29] Clayton V Deutsch and Andre G Journel. Geostatistical software library and user's guide. *Oxford University Press, New York*, 1998.

[30] LJ Durlofsky, Y Efendiev, and V Ginting. An adaptive local–global multiscale finite volume element method for two-phase flow simulations. *Advances in Water Resources*, 30(3):576–588, 2007.

[31] Louis J. Durlofsky. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water resources research*, 27(5):699–708, 1991.

[32] Louis J Durlofsky. Representation of grid block permeability in coarse scale models of randomly heterogeneous porous media. *Water Resources Research*, 28(7):1791–1800, 1992.

[33] Louis J Durlofsky. Upscaling and gridding of fine scale geological models for flow simulation. In *8th International Forum on Reservoir Simulation Iles Borromees, Stresa, Italy*, volume 2024, 2005.

[34] Louis J Durlofsky, Richard C Jones, and William J Milliken. A nonuniform coarsening approach for the scale-up of displacement processes in heterogeneous porous media. *Advances in Water Resources*, 20(5-6):335–347, 1997.

[35] Michael G Edwards. Unstructured, control-volume distributed, full-tensor finite-volume schemes with flow based grids. *Computational Geosciences*, 6(3-4):433–452, 2002.

[36] Michael G Edwards and Clive F Rogers. Finite volume discretization with imposed flux continuity for the general tensor pressure equation. *Computational Geosciences*, 2(4):259–290, 1998.

[37] Yalchin Efendiev and Thomas Y Hou. Multiscale finite element methods, volume 4 of surveys and tutorials in the applied mathematical sciences, 2009.

[38] GT Eigestad, I Aavatsmark, and M Espedal. Symmetry and m-matrix issues for the O-method on an unstructured grid. *Computational Geosciences*, 6(3-4):381–404, 2002.

[39] C. L. Farmer. Upscaling: a review. *International journal for numerical methods in fluids*, 40(1-2):63–78, 2002.

[40] Michel Garcia, Andre G Journel, Khalid Aziz, et al. Automatic grid generation for modeling reservoir heterogeneities. *SPE Reservoir Engineering*, 7(02):278–284, 1992.

[41] M. Gerritsen and J.V. Lambers. Integration of local–global upscaling and grid adaptivity for simulation of subsurface flow in heterogeneous formations. *Computational Geosciences*, 12(2):193–208, 2008.

[42] MG Gerritsen, JV Lambers, and BT Mallison. A variable and compact mpfa for transmissibility upscaling with guaranteed monotonicity. In *Proceedings of the 10th European Conference on the Mathematics of Oil Recovery, Amsterdam*, volume 47, 2006.

[43] J Jaime Gomez-Hernandez and Andre Journel. Stochastic characterization of gridblock permeabilities. *SPE Formation Evaluation*, 9(02):93–99, 1994.

[44] D Guerillot, JL Rudkiewicz, Ch Ravenne, G Renard, and A Galli. An integrated model for computer aided reservoir description: from outcrop study to fluid flow simulations. *Revue de l'Institut français du pétrole*, 45(1):71–77, 1990.

[45] FE Ham, FS Lien, and AB Strong. A cartesian grid method with transient anisotropic adaptation. *Journal of Computational Physics*, 179(2):469–494, 2002.

[46] C He and LJ Durlofsky. Structured flow-based gridding and upscaling for modeling subsurface flow. *Advances in water resources*, 29(12):1876–1892, 2006.

[47] Chuanping He, Michael G Edwards, and Louis J Durlofsky. Numerical calculation of equivalent cell permeability tensors for general quadrilateral control volumes. *Computational Geosciences*, 6(1):29–47, 2002.

[48] Marc A Hesse, Bradley T Mallison, and Hamdi A Tchelepi. Compact multiscale finite volume method for heterogeneous anisotropic elliptic equations. *Multiscale Modeling & Simulation*, 7(2):934–962, 2008.

[49] Lars Holden and Oddvar Lia. A tensor estimator for the homogenization of absolute permeability. *Transport in Porous Media*, 8(1):37–46, 1992.

[50] Lars Holden and Bjørn Fredrik Nielsen. Global upscaling of permeability in heterogeneous reservoirs: The output least squares (OLS) method. *Transport in Porous Media*, 40(2):115–143, 2000.

[51] Edward D Holstein. Volume V – Reservoir Engineering and Petrophysics. *Petroleum Engineering Handbook; Lake, LW, Ed.; Society of Petroleum Engineers: Richardson, TX*, 2007.

[52] Thomas Y Hou and Xiao-Hui Wu. A multiscale finite element method for elliptic problems in composite materials and porous media. *Journal of computational physics*, 134(1):169–189, 1997.

[53] M Rafuqul Islam, M Enamul Hossain, S Hossien Mousavizadegan, Shabbir Mustafiz, and Jamal H Abou-Kassem. *Advanced Petroleum Reservoir Simulation: Towards Developing Reservoir Emulators*. John Wiley & Sons, 2016.

[54] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[55] Patrick Jenny, SH Lee, and Hamdi A Tchelepi. Multi-scale finite-volume method for elliptic problems in subsurface flow simulation. *Journal of Computational Physics*, 187(1):47–67, 2003.

[56] AG Journel. Geostatistics: models and tools for the earth sciences. *Mathematical Geology*, 18(1):119–140, 1986.

[57] Michael J King, Mark Mansfield, et al. Flow simulation of geologic models. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1997.

[58] PR King. The use of renormalization for calculating effective permeability. *Transport in porous media*, 4(1):37–58, 1989.

[59] James V. Lambers, Margot G. Gerritsen, and Bradley T. Mallison. Accurate local upscaling with variable compact multipoint transmissibility calculations. *Computational Geosciences*, 12(3):399–416, 2008.

[60] Seong H Lee, Hamdi A Tchelepi, Patrick Jenny, Larry J DeChant, et al. Implementation of a flux-continuous finite-difference method for stratigraphic, hexahedron grids. *SPE Journal*, 7(03):267–277, 2002.

[61] Xianping Li and Weizhang Huang. An anisotropic mesh adaptation method for the finite element solution of heterogeneous anisotropic diffusion problems. *Journal of Computational Physics*, 229(21):8072–8094, 2010.

[62] Knut-Andreas Lie. *An Introduction to Reservoir Simulation Using MATLAB/GNU Octave: User Guide for the MATLAB Reservoir Simulation Toolbox (MRST)*. Cambridge University Press, 2019.

[63] Vladimir D Liseikin. *Grid generation methods*. Springer, 2017.

[64] Martin J Mlacnik and Louis J Durlofsky. Unstructured grid optimization for improved monotonicity of discrete solutions of elliptic equations with highly anisotropic coefficients. *Journal of Computational Physics*, 216(1):337–361, 2006.

[65] Reza Mohammadnia, Albert C Reynolds, and Fahim Forouzanfar. Monotonicity conditions for MPFA methods for a numerical well testing reservoir simulator. *Journal of Petroleum Science and Engineering*, 158:707–728, 2017.

[66] J. Nilsson, M. Gerritsen, R. Younis, et al. A novel adaptive anisotropic grid framework for efficient reservoir simulation. In *SPE reservoir simulation symposium*. Society of Petroleum Engineers, 2005.

[67] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[68] Jan M Nordbotten, Ivar Aavatsmark, and GT Eigestad. Monotonicity of control volume methods. *Numerische Mathematik*, 106(2):255–288, 2007.

[69] JM Nordbotten and Ivar Aavatsmark. Monotonicity conditions for control volume methods on uniform parallelogram grids in homogeneous media. *Computational Geosciences*, 9(1):61–72, 2005.

[70] G. E. Pickup, P. S. Ringrose, J. L. Jensen, and K. S. Sorbie. Permeability tensors for sedimentary structures. *Mathematical Geology*, 26(2):227–250, 1994.

[71] Dasheng Qi and Tim Hesketh. An analysis of upscaling techniques for reservoir simulation. *Petroleum science and technology*, 23(7-8):827–842, 2005.

[72] Ph Renard and G De Marsily. permeability upscaling: a review. *Advances in water resources*, 20(5-6):253–278, 1997.

[73] Avelino E Saez, Carlos J Otero, and Isak Rusinek. The effective homogeneous behavior of heterogeneous porous media. *Transport in porous media*, 4(3):213–238, 1989.

[74] P Samier. A finite element method for calculating transmissibilities in n-point difference equations using a non-diagonal permeability tensor. In *ECMOR II-2nd European Conference on the Mathematics of Oil Recovery*, 1990.

[75] R Mohan Srivastava. An overview of stochastic methods for reservoir characterization. 1994.

[76] Richard S Varga. On a discrete maximum principle. *SIAM Journal on Numerical Analysis*, 3(2):355–359, 1966.

[77] JE Warren, HS Price, et al. Flow in heterogeneous porous media. *Society of Petroleum Engineers Journal*, 1(03):153–169, 1961.

[78] XH Wen, LJ Durlofsky, and MG Edwards. Use of border regions for improved permeability upscaling. *Mathematical Geology*, 35(5):521–547, 2003.

[79] Xian-Huan Wen, Louis J Durlofsky, Yuguang Chen, et al. Efficient 3d implementation of local-global upscaling for reservoir simulation. *Spe Journal*, 11(04):443–453, 2006.

[80] Xian-Huan Wen and J Jaime Gómez-Hernández. Upscaling hydraulic conductivities in heterogeneous media: An overview. *Journal of Hydrology*, 183(1-2):ix–xxxii, 1996.

[81] Xiao-Hui Wu, Y Efendiev, and Thomas Y Hou. Analysis of upscaling absolute permeability. *Discrete and Continuous Dynamical Systems Series B*, 2(2):185–204, 2002.

[82] Rami M Younis and Jef Caers. A method for static-based up-gridding. In *ECMOR VIII-8th European Conference on the Mathematics of Oil Recovery*, 2002.

[83] Wenjuan Zhang and Mohammed Al Kobaisi. A simplified enhanced MPFA formulation for the elliptic equation on general grids. *Computational Geosciences*, 21(4):621–643, 2017.

[84] Wenjuan Zhang and Mohammed Al Kobaisi. A two-step finite volume method to discretize heterogeneous and anisotropic pressure equation on general grids. *Advances in Water Resources*, 108:231–248, 2017.

[85] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.