On the Design, Implementation, and Use of a Volume-of-Fluid Interface Tracking Algorithm For Modeling Convection and other Processes in the Earth's Mantle

By

Jonathan Michael Robey

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Applied Math

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Elbridge Gerry Puckett, Chair

Robert Guy

Magali I. Billen

Committee in Charge

2019

To my father...

who introduced me to numerical computing,

to my mother and to my sister, Rachel.

Contents

List of Figures

List of Tables

Abstract

On the Design, Implementation, and Use of a Volume-of-Fluid Interface Tracking Algorithm For Modeling Convection and other Processes in the Earth's Mantle

In this dissertation we describe the design, implementation, and use of a two-dimensional, second-order accurate volume-of-fluid interface tracking algorithm in the open source finite element software package ASPECT, which is designed to model convection and other processes in the Earth's mantle. This involves the solution of the incompressible Stokes equations coupled to an advection-diffusion equation for the temperature, a Boussinesq approximation that governs the dependence of the density on the temperature, and an advection equation for composition or some other quantity, such as volume fraction, which is passively advected in the underlying flow field. Our volume-of-fluid method is is fully parallelized and integrated with ASPECT's adaptive mesh refinement algorithm.

To the best of our knowledge, the volume-of-fluid method has not yet been implemented in any other software, which is designed to model convection or other processes in the Earth's mantle. Furthermore we are not aware of any interface tracking methods that have been implemented in a finite element method code for use by the computational mantle convection community. In fact, we are only aware of one other interface tracking algorithm designed to model convection and other processes in the Earth's mantle and this interface tracking algorithm is implemented in a finite difference method rather than a finite element method.

We review the history of the volume-of-fluid method and then describe in detail the the design and implementation of our volume-of-fluid algorithm in ASPECT. After introducing the underlying partial differential equations we use to model mantle convection, we present the results of several interface tracking benchmarks designed to demonstrate numerically that our volume-of-fluid methodology is indeed second-order accurate on smooth flows, as it was designed to be. In addition we demonstrate that our methodology

accurately reproduces two benchmarks that are commonly used in the computational mantle convection community.

We also present the results of two more realistic computations in geodynamics. The first of these problems is a survey of the behavior of a computationally stratified fluid for varying values of a nondimensional buoyancy parameter. This model problem is intended to provide insight into how thermal plumes, which eventually reach the Earth's surface where they melt to form ocean island basalts separate from structures near the core-mantle boundary, which are denser than the surrounding mantle. These structures, known as Large Low Shear wave Velocity Provinces or "LLSVPs". LLSVPs lie in parts of the lowermost portion of the Earth's mantle and are characterized by slow shear wave velocities and higher density than the surrounding mantle. They were first discovered by seismic tomography of the deep Earth.

The second problem in computational mantle convection we present is that of a subducting slab. This computation is a basic model of, for example, the subduction of the Pacific tectonic plate beneath the South American tectonic plate. This problem involves a more complex material model than the other problems presented in this dissertation. The slab consists of an overriding crustal layer on top of a layer consisting largely of harzburgite thereby demonstrating how, with the aid of adaptive mesh refinement, one can use our volume-of-fluid methodology to track more than two materials in a single problem.

Acknowledgments

I would like to thank my advisor, Professor Elbridge Gerry Puckett, for his guidance and support during this research.

Also many thanks to Professors Magali Billen and Robert Guy for reading my thesis in its final stages and offering valuable advice for making it better.

I would like to thank Magali Billen and Donald Turcotte for their help with developing and interpreting the mantle convection application problems in Chapter 5.

Thanks to all of the core development team of Deal.II and ASPECT. Specifically, I would like to thank Wolfgang Bangerth and Timo Heister for their advice on implementation structure and especially Rene Gassmoeller for his extremely helpful and patient code review during the process of merging this volume-of-fluid methodology into the main ASPECT code repository. And to John Naliboff and Max Rudolf for their help in identifying the correct configuration for Deal.II on Peloton II.

This work was supported by the National Science Foundation's (NSF) SI2-SSE Program under Award number 1440811. The development of ASPECT was supported by the Computational Infrastructure for Geodynamics (CIG) under NSF Award numbers 0949446 and 1550901. The computations were made under the auspices of CIG on the U.C. Davis Division of Mathematical and Physical Sciences distributed computing cluster Peloton.

Chapter 1

Introduction

1.1 Motivation

Over more than the past four decades there have been many numerical methods developed to study convection and other processes in the Earth's mantle. In particular, there have a been a sequence of codes developed over this period of time that are now freely available to any individual who wishes to study mantle dynamics. They include HC (Hager and O'Connell, 1981; Hager and Clayton, 1989; Steinberger, 2000), ConMan (King et al., 1990), CitCom S (McNamara and Zhong, 2004; Tan et al., 2006; Zhong et al., 2000), Citcom CU (Moresi and Gurnis, 1996; Zhong, 2006) and ASPECT (Kronbichler et al., 2012; Heister et al., 2017). These codes, as well as others, can be downloaded from the Computational Infrastructure for Geodynamics (CIG) at U.C. Davis.[1]

There are a large number of problems associated with the Earth's mantle that contain one or more interfaces in some form or another. Although there have been some very specialized computational models of interfaces in the mantle; e.g., the dynamics of bubbles and plumes (Manga, 1996; Manga and Stone, 1994; Manga et al., 1993), to the best of our knowledge it is only now that researchers have begun to consider using interface tracking algorithms in codes designed to model convection and other processes in the entirety of the Earth's mantle. In fact, to the best of our knowledge, the Volume-of-Fluid (VOF) method, or any other interface tracking method, have not previously been implemented in

[1]The CIG is an NSF funded, community driven organization that advances Earth science by developing and disseminating software for geophysics and related fields.

a finite-element code designed to model convection in the Earth's mantle. A particle level set interface tracking algorithm (Enright et al., 2002) has been used in a code intended to study mantle convection(Samuel and Evonuk, 2010), but that code is both finite difference and makes use of a significantly different approach.

1.2 Review of Volume of Fluid Methods

There are a wide variety of VOF interface reconstruction and advection algorithms; e.g., see Pilliod and Puckett (2004); Rider and Kothe (1998); Scardovelli and Zaleski (2003) and the references therein. The VOF method was first developed at the U.S. National Labs in the 1970s (Noh and Woodward, 1976) and have continued to be used and developed by researchers at the National Labs (Hirt and Nichols, 1981; Nichols et al., 1980; Torrey et al., 1985, 1987) as well as around the world (e.g., Parker and Youngs (1992))

VOF methods can and have been used effectively to model a wide variety of moving interface problems, including interfaces in compressible flow with shock waves (Henderson et al., 1991), interfaces with shock waves in materials in the limit of no strength effects (Miller and Puckett, 1994, 1996), jetting in meteorite impacts (Puckett and Miller, 1996), *nonconservative* interface motion such as photolithography (Helmsen et al., 1997, 1996), the transition from deflagration to detonation (Pilliod and Puckett, 1998) and more than two materials; i.e., more than one interface in a cell (Anbarlooei and Mazaheri, 2011; Hill and Shashkov, 2013).

An advantage that VOF methods have over other interface tracking methods is that they can readily (or naturally) be designed to approximate solutions of a conservation equation such as equation (3.26) for the composition. Thus, materials that should be conserved as they move with the flow are (*theoretically*) conserved. However, in practice, i.e., when the algorithm is implemented on a computer, some VOF advection algorithms —including the one we use here —will only conserve the volume of the composition or fluid that is being tracked up to some numerical error that typically depends on the grid size, h. This will depend on the design of the VOF advection algorithm. There has been recent research into how to design VOF advection algorithms that conserve volume to machine

precision; e.g., see Scardovelli and Zaleski (2003); Aulisa et al. (2007); Weymouth and Yue (2010).

1.3 Interface Tracking in Finite Element Methods

In this section we present a brief survey of the implementation of interface tracking algorithms in finite element methods. To begin, we emphasize that, to the best of our knowledge, this is the first implementation of a VOF interface tracking method in any code —finite element, finite difference, or any other, that has been specifically designed to model convection and other processes in the Earth's mantle. Furthermore, we know of only one other interface tracking algorithm that has been implemented in such a code; namely the particle level set method (Samuel and Evonuk, 2010), which has been implemented in a finite difference code. Several of the most commonly used interface tracking methods such as VOF, level set, etc. have been implemented in finite element methods designed for computing different problems. Here we only survey implementations of VOF, level set, and composition field approaches. A survey of particular cases for other problems may be found in Benson (1998). For a review of interface tacking algorithms in general we refer the reader to Tryggvason et al. (2011) and Elgeti and Sauerland (2015).

The most basic approach to implementing an interface tracking in an FEM method is the use of a so-called 'compositional field', such as described in Shin and Lee (2000), to track the boundary between two distinct compositions. These types of algorithms, which do not maintain a sharp boundary or interface between the two compositions, are typically referred to as *interface capturing methods* in the interface tracking community in order to distinguish them from methods that do maintain a sharp boundary. See Puckett et al. (2018) for a comparison of the VOF method described in this dissertation with three interface capturing methods that have been implemented in ASPECT.

However, using an interface capturing method leads to a number of limitations and numerical assumptions that reduce the accuracy of compositional advection algorithms in a FEM. In particular, the imposition of a continuous approximation to the two distinct compositions, in each cell and at cell edges / faces, where there should be sharp discontinuity

between the compositions, will 'smear' the interface, as opposed to maintaining a sharp interface.

In our implementation of the VOF method in ASPECT the volume fractions $f_{i,j}$ are stored as Discontinuous Galerkin elements that carry a constant value on each cell; i.e, DGQ0 elements. In the terminology used in the ASPECT community these elements technically constitute a compositional field. However, since on the volume fraction is constant each element this is essential equivalent to VOF implementations in finite difference / finite volume methods on arbitrary meshes. However, since ASPECT, which requires a smooth function for accuracy in the assembly of the Stokes matrix, which is done with Gaussian quadrature, we present the FEM code with a DGQ1 approximation to the reconstructed interface These implementation issues are discussed in detail in Section 2.7.

In cases where the volume of fluid interface tracking algorithm has been implemented in a finite element method, some researchers have placed the volume of fluid algorithm on a separate mesh from the finite element mesh (Mashayek and Ashgriz, 1995; Maronnier et al., 1999). This separated approach requires the maintenance of a large and complex set of code paths to tie together the two meshes. Instead our implementation is on the same FEM grid as the rest of the ASPECT computation, thereby leading to much more natural coupling of the two methods.

Level set methods, like VOF methods, and other *interface tracking* methods are designed to model sharp interfaces and therefore, do not allow numerical smearing of the interface. The initial implementation of the level set method for incompressible flow was done in a finite difference method by Sussman et al. (1994). Standard references for the level set method are Sethian (1999) and Osher and Fedkiw (2003). However, since the time when these books were published there has been a lot of new work on level set methods. Most, if not all, early implementations of the level set method were in finite difference or finite volume codes. However, later work included implementations in finite element methods (Tornberg and Engquist, 2000; Nagrath et al., 2005).

One of the primary benefits of the level set method is that, if at each time step the level set function is a signed distance function (i.e, each constant contour is the signed

distance to the interface, which is the zero contour), then the curvature of the interface can easily be found by computing the Laplacian of the level set function. In theory, this enables a straightforward numerical approach to problems involving surface tension and flows by mean curvature. However, there are very few problems in the Earth's mantle that require the curvature or surface tension of and interface. Furthermore, it was discovered early on, in Sussman et al. (1994), that when the distance function is advected in a flow field, it does not remain a distance function. This has lead to a tremendous literature on 'redistancing' algorithms, beginning with Sussman et al. (1994). We note that one successful approach to this problem is the Coupled Level Set Volume of Fluid (CLSVOF) algorithm of Sussman and Puckett (2000a), which couples the strengths of both the level set and volume of fluid algorithms and consequently has been implemented in numerous codes for a wide variety of applications.

For additional background we refer the reader to the books and articles cited above.

1.4 Synopsis of this Research

In this research, we have designed and implemented a second-order accurate VOF interface tracking algorithm in the open source finite element code ASPECT. ASPECT is a parallel, extensible finite element code designed to model thermal convection and other processes in the Earth's mantle in two and three dimensions. It is built on the deal.II Finite Element Library (Arndt et al., 2017a; Bangerth et al., 2007), which includes adaptive mesh refinement (AMR) (Burstedde et al., 2011a) and has been shown to scale to thousands of processors (Gassmoeller, 2016). ASPECT has been extended to model other processes that occur in the mantle, such as modeling grain size evolution in the mantle (Dannberg et al., 2017), melt generation and migration (Dannberg and Heister, 2016), as well as other problems. There is currently a very active community of researchers extending ASPECT to new problem areas and improving existing algorithms. Our VOF algorithm is fully parallelized and is designed to work efficiently with ASPECT's AMR algorithm.

We have validated this new interface tracking algorithm on a variety of simple test problems and benchmarks from the computational mantle convection community; e.g.,

see Puckett et al. (2018); Robey and Puckett (2019) as well as Chapter 4. We have also made an extensive comparison of its efficacy as compared to the three other advection methods in ASPECT (Puckett et al., 2018), and used it for computing a model of LLSVPs with two different initial conditions (Puckett et al., 2018) and Robey and Puckett (2019), respectively. The latter work also contains comparison of the computational results from this model problem with the experiments of Davaille (1999) and Le Bars and Davaille (2004, 2005). This latter work also appears in Section 5.1 and Section 5.1.2 Finally in Section 5.3 we present a model of a subduction zone. This problem involves a more complex material model than the computations mentioned above and also demonstrates how, with the aid of adaptive mesh refinement, one can use our VOF methodology to track more than two materials in a single problem.

1.5 Summary of the Contents of this Dissertation

In Chapter 2 we begin by describing the VOF method in detail as well as the details of our design and implementation of it in ASPECT. Then, in Section 3.1 we describe the equations that we use to model thermochemical convection in the mantle and the additional advection equation that we add in order to model the advection of two distinct materials with our interface tracking algorithm. We also use this additional equation to model the advection of two distinct materials with a Discontinuous Galerkin (DG) method for the purposes of comparison with the VOF results. In Section 3.2 we describe the general numerical methodology of the underlying Finite Element Method (FEM) in ASPECT.

In Chapter 4, we confirm the correct behavior of our new VOF method —in particular that it is second-order accurate —using both interface tracking and mantle convection community benchmarks. In Chapter 5 we then compute two application problems from the field of mantle convection / geophysics. In Chapter 6, we present our conclusions, and in Chapter 7 we discuss possible future continuations of the work.

Table 1.1: A list of symbols used in this dissertation.

Symbol	Quantity	Unit
$\mathbf{u}$	Velocity	$\mathrm{m\,s^{-1}}$
ρ	Density	$\mathrm{kg/m^3}$
p	Dynamic pressure	Pa
$\Delta\rho$	Density difference	$\mathrm{kg/m^3}$
T_0	Temperature at the top	K
D	Compositional diffusivity	$\mathrm{m^2/s}$
T_1	Temperature at the bottom	K
α	Thermal expansion coefficient	$\mathrm{K^{-1}}$
T	Temperature	K
d	Vertical height of fluid layer	m
ΔT	Temperature difference	K
C	Composition variable	-
μ	Viscosity	$\mathrm{Pa\,s}$
κ	Thermal diffusivity	$\mathrm{m^2/s}$
ρ_0	Reference density	$\mathrm{kg/m^3}$
f	Volume Fraction	-
g	Acceleration due to gravity	$\mathrm{m/s^2}$
$\dot{\varepsilon}$	Rate of strain	$\mathrm{s^{-1}}$
Pr	Prandtl number	$\frac{\mu}{\rho\kappa}$
Le	Lewis number	$\frac{\kappa}{D}$
Ra	Rayleigh number	$\frac{\rho_0 \mathbf{g} \alpha \Delta T d^3}{\mu\kappa}$
B	Buoyancy ratio	$\frac{\Delta\rho}{\rho_0 \alpha \Delta T}$

Chapter 2

THE VOLUME-OF-FLUID ALGORITHM

The Volume-of-Fluid (VOF) method is an interface tracking method in which, at each time step, there are two distinct steps. In the first step the interface between two fluids or compositions is explicitly reconstructed with an *interface reconstruction method* in every cell that contains a portion of the interface. For example, in our computations of the model problem described in Section 3.1 below, the compositional variable C will have a value of $C = 1$ in cells completely occupied by the fluid with density $\rho = \rho_0 + \Delta\rho$ and a value of $C = 0$ in cells completely occupied by the fluid with density $\rho = \rho_0$. Thus, cells in which $0 < C < 1$ contain a portion of the interface. Given the explicit (but approximate) reconstructed interface in each cell with $0 < C < 1$ at the current time step one then uses this information to advance the interface in time with an *advection method*. In this sense the VOF method approximates the compositional interface on a subgrid scale.

In Section 4.3 below we examine the degree to which the VOF advection algorithm we use in this work conserves volume and compare the results with a Bound Preserving Discontinuous Galerkin (DGBP) advection method (He et al., 2017), which is also implemented in ASPECT. See Section 3.2.4 below for a brief description of the DGBP method and Section 4.3 for the computational results.

2.1 Overview

In this dissertation we use a two-dimensional VOF algorithm to discretize the conservation equation (2.1), which —when describing the VOF method —we will usually write in the following form,[1]

$$\frac{\partial f}{\partial t} + \nabla \cdot \mathbf{F}(f) = 0\,. \tag{2.1}$$

Here, f is the *volume fraction* of one of the compositional fields, say $C = 1$, the field with density $\rho_0 + \Delta\rho$, which we will refer to as 'Composition 1', or C_1 for short, and

$$\mathbf{F}(f) = (F(f), G(f)) = (u\,f,\, v\,f) = \mathbf{u}\,f\,, \tag{2.2}$$

is the volume fraction flux associated with C_1, where $\mathbf{u} = (u,\, v)$ is the velocity field.[2] Since for incompressible flow, $\nabla \cdot \mathbf{u} = 0$, one can rewrite equation (2.1) as a pure advection equation for f,

$$\frac{\partial f}{\partial t} + u\,\frac{\partial f}{\partial x} + v\,\frac{\partial f}{\partial y} = 0\,. \tag{2.3}$$

From a mathematical point of view the variable $f(x, y, t)$ in the conservation equation (2.1) with fluxes (2.2) may be regarded as the characteristic function (sometimes denoted $\chi(x, y, t)$) associated with the composition C_1. In other words,

$$f(x,y) = \begin{cases} f(x,y) = 1 & \text{if } (x,y) \text{ is occupied by Composition 1}\,, \\ f(x,y) = 0 & \text{if } (x,y) \text{ is not occupied by Composition 1}\,. \end{cases} \tag{2.4}$$

This implies $1 - f(x, y)$ is the characteristic function associated with C_2, the composition with density $\rho = \rho_0$.

Our use of the variable f to represent the quantity that is advected in a VOF method is historical. In particular, in the presentation of our VOF method the variable C and f can be used interchangeably. In practice, we will use the convention that f refers to the volume fraction (a quantity linked to the cell), while C refers to the piecewise continuous

[1] In equation (2.1) and in much of what follows, depending on the context, we will use the symbol f, $f_{i,j}$, or f_e, for the fraction of the fluid or composition denoted by $C = 1$ that occupies the cell $\Omega_{i,j}$ or cell Ω_e.

[2] Throughout this section and beyond we will use the terms "volume" and "volume fraction" of C_1, etc., although it is to be understood that in two dimensions the quantity in question is an area.

function over space which approximates the VOF data for use by the FEM portion of the code. In Section 3.2.4 below we will describe other methods in ASPECT for advecting the quantity C and in Section 4.3 we will use one of these other advection methods to compute a test problem in order to compare the results of an entirely different advection method with the results we obtain with the VOF method presented here. In these sections, as well as in Section 5.1 where we show the results of our computations of the model problem presented in Section 3.1, our use of the variable C is typical of the notation researchers use in the computational geodynamics literature.

```
Initialize mesh
Initialize state
    VOF initialization here
For each timestep
    Do variable update including nonlinear iteration
        VOF update here
    Do mesh refinement update
    Do postprocessing including state output
```

Figure 2.1: General outline of the structure of the Aspect code

A short pseudo-code description of the structure of Aspect and the general structure of the newly implemented VOF advection algorithm is shown in figure 2.1. In particular, we note that the implementation has been designed to be self-contained in a manner externally similar to the standard finite element advection algorithm.

In our VOF implementation in ASPECT we use the 'Efficient Least Squares VOF Interface Reconstruction Algorithm' (ELVIRA), which is described in detail in Pilliod and Puckett (2004) and is based on the ideas in Puckett (1991) and Pilliod (1992). The ELVIRA interface reconstruction algorithm reconstructs lines on a uniform grid with square cells *exactly*; i.e., to machine precision. We explain this in more detail in Section 2.3 and Figure 2.4, and also demonstrate it with a computational example in Section 4.2.1 below. Since the ELVIRA algorithm reconstructs lines in square cells exactly it is natural

to assume that the algorithm is second-order accurate on a uniform grid with identical square cells. This has been proven to be true (Puckett, 2010a, 2014). We use a second-order accurate dimensionally split advection method, which is described in Section 2.2.1 of Puckett et al. (1997) and briefly described in pseudocode in figure 2.6, to update the values of the volume fractions in time. For simplicity of exposition we will assume the finite element grid consists entirely of square cells Ω_e, of side h, indexed by the variable e, and aligned parallel to the x and y axes. This is the case for all of the computations we present in this dissertation.

However, we emphasize that VOF algorithms have been developed and applied to problems on a wide variety of unstructured grids in three dimensions (Korzekwa et al., 1999), including tetrahedral (Williams et al., 1999a), hexahedral (Williams et al., 1999b) and general convex grids (López et al., 2016), as well as having been developed and applied to great many applications on non-rectangular grids in two dimensions. However, the interface reconstruction algorithm and advection algorithm on irregular grids in two and three dimensions must be modified or completely redesigned for such grids. This has been the focus of much research over the past 20 or 30 years. For example, one approach is to minimize the difference, say in the least-squares sense (i.e., the discrete two norm), between the given volume fractions and the volume fractions due to a linear interface in two dimensions or planar interface in three dimensions in a neighborhood of the cell of interest with a minimization algorithm such as Brent's method; e.g., see Puckett (1991); Pilliod and Puckett (1997, 2004); Rider and Kothe (1998) for examples and comments on this approach and Weymouth and Yue (2010) for alternate approaches.

The discretization of equation (2.1) proceeds as follows. Let Ω_e denote an arbitrary cell in the computational domain Ω and let f_e^k denote the discretized volume fraction in Ω_e at time t^k. The variable f_e^k is a scalar that satisfies $0 \leq f_e^k \leq 1$ such that

$$f_e^k \approx \frac{1}{h^2} \int_{\Omega_e} f(x, y, t^k)\, dx\, dy\,. \tag{2.5}$$

Thus, the discretized volume, V_e^k, of C_1 in Ω_e at time t^k is

$$V_e^k = \int_{\Omega_e} f_e^k \, dx \, dy = h^2 \, f_e^k \, . \tag{2.6}$$

Note that for an incompressible velocity field $\mathbf{u} = (u, v)$ we have $\nabla \cdot \mathbf{u} = 0$ and hence, for the true solution the volume of 'parcels' or regions of C_1 are constant as they evolve in time.

For most material models, cells that contain a portion of the interface (i.e., $0 < f_e^k < 1$) will have a density given by

$$\rho = f_e^k \, (\rho_0 + \Delta\rho) + \left(1 - f_e^k\right) \rho_0 \, . \tag{2.7}$$

In other words a weighted average of the two compositions C_1 and C_2 with weights f_e^k and $\left(1 - f_e^k\right)$. In most of our work with the VOF method to track the interface between two compositions we use the weighted average in equation (2.7), with C_1 replacing $(\rho_0 + \Delta\rho)$ and C_2 replacing ρ_0. This is common practice among researchers who use VOF methods; e.g., see equation (2) of Weymouth and Yue (2010).

In this dissertation we restrict ourselves to modeling the interface between two compositions. However, there is currently a great deal of research into modeling two or more interfaces in one cell with a VOF method; e.g., see Jemison et al. (2015) and the references there.

In its simplest form our implementation of the VOF algorithm in ASPECT proceeds as follows. Given the values f_e^k at time t^k and the velocity field at time t^k we do the following to obtain the volume fractions f_e^{k+1} at time t^{k+1}.

For convenience and clarity of exposition, in the remainder of this section we will *usually* use the index notation (i, j), as shown in Figures 2.2–2.4. Thus, we have nine cells with centers $(x_{i'}, y_{j'})$ for $i' = i - 1, \, i, \, i + 1$ and $j' = j - 1, \, j, \, j + 1$ with edges $x = x_{i\pm\frac{1}{2}} = x_i \pm \frac{h}{2}$ and $x = x_{i\pm\frac{3}{2}} = x_i \pm \frac{3h}{2}$ and similarly for y as shown in Figure 2.2. In the ELVIRA interface reconstruction algorithm we use the information in the 3×3 block of cells $\Omega_{i'j'}$ immediately adjacent to the cell $\Omega_e \equiv \Omega_{ij}$ in which we wish to reconstruct the interface. Note that we use both Ω_e and Ω_{ij} to denote the center cell of the 3×3

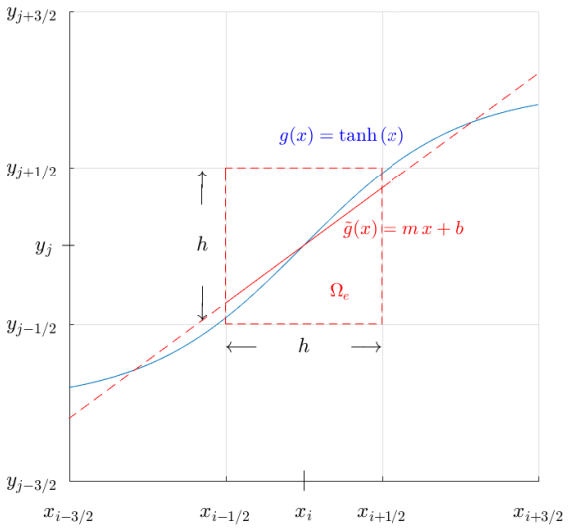


Figure 2.2: In our implementation of the VOF interface reconstruction algorithm the true interface, which in this example is $g(x) = \tanh(x)$, is approximated as a line segment $\tilde{g}_e(x) = m_e\,x + b_e$ in each cell Ω_e that has a volume fraction f_e with $0 < f_e < 1$. The approximate interface in Ω_e is depicted as the solid red line segment in the center cell Ω_e. In this example, as with all VOF methods, the volume $h^2 f_e^{true}$ beneath the true interface in Ω_e is *exactly equal* to the volume $h^2 f_e$ beneath the approximate interface $\tilde{g}$ in Ω_e; i.e., $f_e^{true} = f_e$. Note that, for convenience, we have used the notation (x_i, y_j) to denote the center of the cell Ω_e, $[x_{i-1/2},\, x_{i+1/2}] \times [y_{i-1/2},\, y_{i+1/2}]$ to denote the cell Ω_e, etc.

block of cells. The reason for this is that in peer reviewed and technical (i.e., the manual, etc.) literature that describes ASPECT the notation Ω_e is often used to denote cells, while the notation Ω_{ij} is commonly used in VOF literature, especially when coupled to a finite volume or finite difference method.

1. THE INTERFACE RECONSTRUCTION STEP: Given a cell Ω_{ij} that contains a portion of the interface, so $0 < f_e^k < 1$ where f_e^k is the volume fraction in Ω_e at time t^k, we use the volume fractions $f_{e'}^k$ in the 3×3 block of cells $\Omega_{e'}$ centered on the cell Ω_e to *reconstruct* the interface in Ω_e. The reconstructed interface will be a piecewise linear approximation to the true interface as shown in Figure 2.2 that preserves the given volume $h^2\, f_e^k$ of C_1 in Ω_e. We give a brief description of how we determine the linear approximation $\tilde{g}_e(x) = m_e\,x\, +\, b_e$, to the true interface in cells Ω_e for which $0 < f_e^k < 1$ in Section 2.3 below.

2. COMPUTATION OF THE FLUXES: In the computations presented in this dissertation we use a second-order accurate dimensionally split (also known as "operator split") advection algorithm in order to advance the interface in time. However, for clarity and simplicity of exposition, in this section we will only describe *the simplest possible version* of a dimensionally split advection algorithm for updating the location of the interface. In Section 2.5 below we will note an important modification to the dimensionally split advection algorithm described here. *We use this modified dimensionally split advection algorithm* described in Section 2.5 in all of the computations shown in this dissertation.

   Given the reconstructed interface $\tilde{g}_e = \tilde{g}_{ij}(x)$ in

$$\Omega_e \equiv \Omega_{ij} = [x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{i+1/2}]$$

   as shown in Figure 2.3 and the velocity $u^k_{i\pm1/2,j}$ normal to the right and left edges of Ω_{ij} at time t^k, we wish to determine the volumes $V^k_{i\pm1/2,j}$ of C_1 that cross the right and left edges of Ω_e in the time interval $[t^k, t^{k+1}]$. These volumes are determined *geometrically*. Since we are using a dimensionally split advection method, the total volume of both compositions C_1 and C_2 that crosses each cell edge will be that of a rectangle. This is illustrated in Figure 2.3 for the right edge of the cell Ω_e where we have assumed that $u^k_{i\pm1/2,j} > 0$. The rectangle is shown in green and pink and the volume of the rectangle is $u_{i+i/2,j}\,\Delta t\, h$. We then determine the volume $V^k_{i+1/2,j}$ of C_1 that crosses the right-hand edge of Ω_{ij} in the time interval $[t^k, t^{k+1}]$ is outlined in green on three sides and by a portion of the solid red line on top in Figure 2.3.

3. THE VOLUME FRACTION UPDATE: Now we describe the *unmodified* dimensionally split VOF advection method, which we are referring to as here as the "Volume Fraction Update". One may also think of this as a "Volume Update"; i.e., the update $V^k_{e,1} \to V^{k+1}_{e,1}$ of the volume of C_1 in cell e, which is how we have chosen to present the algorithm here. This is the simplest possible dimensionally split VOF advection method. However, a simple modification, which we will describe in Section 2.5 below,

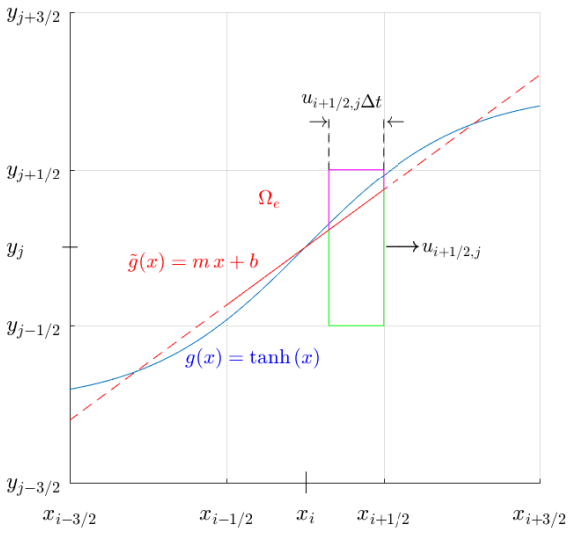


Figure 2.3: The volume $V^k_{i+1/2,j,1} = \delta x\, \tilde{g}(\delta x)$ where $\delta x = u^k_{i+i/2,j}\,\Delta t^k$ of C_1 in the quadrilateral outlined in green on three sides and by a portion of the solid red line on top is the flux of of C_1 that will cross the right-hand edge of Ω_e during the time step from time t^k to t^{k+1}. Here $\Delta t^k = t^{k+1} - t^k$ and we have dropped the superscript k from $u^k_{i+i/2,j}$ and Δt^k in the diagram for clarity. The solid red line in Ω_e is the reconstructed interface $\tilde{g}(x)$ that approximates the true interface $g(x) = \tanh(x)$ in Ω_e at time t^k as shown in Figure 2.2.

greatly improves the volume conservation of the method. The update proceeds in two steps.

Step I Given the volume $V^k_{ij,1} = h^2 f^k_{ij}$ of C_1 in Ω_{ij} at time t^k and the volumes $V^k_{i\pm1/2,j,1}$ of C_1 that cross the left and right-hand edges, respectively, of Ω_{ij} in the time interval $[t^k, t^{k+1}]$ we use the following equation to determine an *intermediate* volume $\tilde{V}^k_{ij,1}$ of C_1 in Ω_{ij} for the first step of the two step dimensionally split algorithm,

$$\tilde{V}^k_{ij,1} \;=\; V^k_{ij,1} \;+\; V^k_{i-1/2,j,1} - V^k_{i+1/2,j,1}\,. \tag{2.8}$$

We do this in every cell Ω_{ij} before advancing to Step II below. In Figure 2.3 we illustrate how we determine the volume $V^k_{i+1/2,j,1}$ that crosses the right edge of Ω_{ij}.

Step II Now, given the nine intermediate volume fractions

$$\tilde{f}^k_{i'j'} \;=\; \frac{1}{h^2}\,\tilde{V}^k_{i'j',1}$$

in Ω_{ij} and the 3×3 block of cells $\Omega_{i'j'}$ surrounding Ω_{ij}, together with all of the intermediate volume fractions in the 3×3 block of cells surrounding each of the cells $\Omega_{i'j'}$, we reconstruct an intermediate interface $\hat{g}_{i'j'}(x)$ in each cell $\Omega_{i'j'}$. We then use this intermediate interface to geometrically determine the volumes $\tilde{V}^k_{i,j\pm1/2,1}$ of C_1 that cross the *top* and *bottom* edges of Ω_{ij} in the time interval $[t^k, t^{k+1}]$ in the same manner as illustrated in Figure 2.3, but this time in the y-direction. Now the volume $V^{k+1}_{ij,1}$ of C_1 in Ω_{ij} at the new time t^{k+1} is,

$$V^{k+1}_{ij,1} \;=\; \tilde{V}^k_{ij,1} \;+\; \tilde{V}^k_{i,j-1/2,1} - \tilde{V}^k_{i,j+1/2,1} \tag{2.9}$$

and the new volume fraction in Ω_{ij} is

$$f^{k+1}_{ij} \;=\; \frac{1}{h^2}\,V^{k+1}_{ij,1}\,. \tag{2.10}$$

This is the simplest of all dimensionally split volume fraction advection algorithms. It can be made to be second-order accurate by alternating the direction of the first volume update at each time step, a procedure that is known a 'Strang splitting" in the numerical methods community (Strang, 1968). There are also *unsplit* VOF advection algorithms; e.g., see Pilliod and Puckett (2004); Puckett et al. (1997); Rider and Kothe (1998).

2.2 Derivation of the Volume-of-Fluid Algorithm using a FEM Approach

Letting cell e be indicated by Ω_e, and letting f be the indicator function for fluid 1, we define the shape function ψ_e, which is constant over cell e and zero elsewhere. The standard advection equation for quantities carried by the fluid, multiplied by the shape function ψ_e is then

$$\frac{\partial f}{\partial t}\psi_e = -\nabla \cdot \mathbf{u}\psi_e\,. \tag{2.11}$$

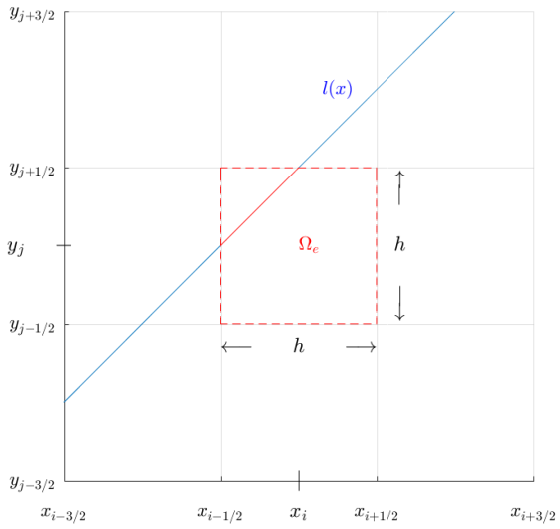


Figure 2.4: In this example the true interface is the line $l(x) = m\,x + b$ Note that that the volumes V_{i-1} and V_i under the line in the first two columns $i-1$ and i are exactly equal to the volumes due to the column sums $\tilde{V}_{i-1} = h^2\,S_{i-1}$ and $\tilde{V}_i = h^2\,S_i$ in the first and second columns of the 3×3 block of cells B_{ij} centered on the center cell $\Omega_e (= \Omega_{ij})$. In this case the slope $\tilde{m} = S_i - S_{i-1}$ is *exactly* equal to the slope m of the interface as shown in (2.23). It is always the case that if the true interface is a line, then one of the four standard rotations of B_{ij} by a multiple of 90 degrees about its center will orient the block so at least one of the divided differences of the column sums in (2.26) or (2.27) is exact and hence, one of the linear approximations to the interface in the center cell Ω_e defined in (2.29) will always equal the interface in that cell, *exactly*, $\tilde{g}_{ij}(x) = m_{ij}\,x + b_{ij} = m\,x + b = l(x)$. In other words, the piecewise linear VOF approximation to $l(x)$ will *always* reconstruct the linear interface exactly.

Integrating this over the spatial dimensions for each ψ_e gives the integral form for the volume of fluid problem as

$$\frac{d}{dt}\int\limits_{\Omega_e} f d\mathbf{x} = -\int\limits_{\Omega_e} \mathbf{u}\cdot\nabla f d\mathbf{x}\,. \tag{2.12}$$

Using the divergence theorem, this becomes

$$\frac{d}{dt}\int\limits_{\Omega_e} f d\mathbf{x} = \int\limits_{\Omega_e} f\nabla\cdot\mathbf{u} d\mathbf{x} - \int\limits_{\partial\Omega_e} f\mathbf{u}\cdot\mathbf{n} d\mathbf{x}\,. \tag{2.13}$$

Integrating over the time dimension and letting $f^n_{h(i)} = \int\limits_{\Omega_e} f d\mathbf{x}$ evaluated at t^n, we have

the equation

$$f_{h(i)}^{n+1} = f_{h(i)}^{n} + \int_{t^n}^{t^{n+1}} \int_{\Omega_e} f \nabla \cdot \mathbf{u} d\mathbf{x} dt - \int_{t^n}^{t^{n+1}} \int_{\partial\Omega_e} f \mathbf{u} \cdot \mathbf{n} d\mathbf{x} dt \,. \tag{2.14}$$

As Ω_e is a quadrilateral, we may indicate the faces of Ω_e as $\partial\Omega_{ej}$, where $j \in \{0, 1, 2, 3\}$ allowing the boundary term to be rewritten as

$$f_{h(i)}^{n+1} = f_{h(i)}^{n} + \int_{t^n}^{t^{n+1}} \int_{\Omega_e} f \nabla \cdot \mathbf{u} d\mathbf{x} dt - \sum_j \int_{t^n}^{t^{n+1}} \int_{\partial\Omega_{ej}} f \mathbf{u} \cdot \mathbf{n} d\mathbf{x} dt \,. \tag{2.15}$$

To farther refine the notation for later use, we let $F\Omega_{ej}^n$ indicate $\mathbf{x} \in \partial\Omega_{ej}, t \in [t^n, t^{n+1}]$ and rewrite the equation as

$$f_{h(i)}^{n+1} = f_{h(i)}^{n} + \int_{t^n}^{t^{n+1}} \int_{\Omega_e} f \nabla \cdot \mathbf{u} d\mathbf{x} dt - \sum_j \int_{F\Omega_{ej}^n} f \mathbf{u} \cdot \mathbf{n} d(\mathbf{x}, t) \,. \tag{2.16}$$

We now prepare to make the approximations. Using $\overline{a}_{\Omega_i}$ to indicate the average of the given quantity a over the region Ω_e, we can write the equation with the error terms as

$$\begin{aligned} f_{h(i)}^{n+1} = f_{h(i)}^{n} &+ \frac{\Delta t}{2} \left(f_{h(i)}^{n+1} \overline{\nabla \cdot \mathbf{u}^{n+1}}_{\Omega_e} + f_{h(i)}^{n} \overline{\nabla \cdot \mathbf{u}^{n}}_{\Omega_e} \right) \\ &+ \int_{t^n}^{t^{n+1}} \int_{\Omega_e} f (\nabla \cdot \mathbf{u} - \frac{1}{2} (\overline{\nabla \cdot \mathbf{u}^{n+1} + \nabla \cdot \mathbf{u}^{n}}_{\Omega_e}) d\mathbf{x} dt \\ &- \sum_j \left(\int_{F\Omega_{ej}} f d(\mathbf{x}, t) \overline{\mathbf{u} \cdot \mathbf{n}}_{F\Omega_{ej}^n} + \int_{F\Omega_{ej}^n} f (\mathbf{u} \cdot \mathbf{n} - \overline{\mathbf{u} \cdot \mathbf{n}}_{F\Omega_{ej}^n}) d(\mathbf{x}, t) \right) . \end{aligned} \tag{2.17}$$

Selecting different means of calculating the divergence term approximation may allow better approximation or overshoot avoidance. Given the current application is expected to be mostly incompressible, this will not be considered in any depth. The two terms with

differences from the average are of order $V(\Omega_e)h\Delta t$, leaving the equation as

$$f_{h(i)}^{n+1} = f_{h(i)}^{n} + \frac{\Delta t}{2}\left(f_{h(i)}^{n+1}\overline{\nabla\cdot\mathbf{u}^{n+1}}_{\Omega_e} + f_{h(i)}^{n}\overline{\nabla\cdot\mathbf{u}^{n}}_{\Omega_e}\right) - \sum_j \left(\int_{F\Omega_{ej}} fd(\mathbf{x},t)\overline{\mathbf{u}\cdot\mathbf{n}}_{F\Omega_{ej}^n}\right) + O(V(\Omega_e)h\Delta t)\,. \tag{2.18}$$

At this point, the discreization is identical to the standard VoF derivation, and also identical to a Discontinuous Galerkin method using zeroth order discretizations (constant over cell) or equivalently a finite volume method depending on how $f_{h(F\Omega_{ej}^n)} = \int_{F\Omega_{ej}} fd(\mathbf{x},t)$ is calculated.

2.2.1 Volume of Fluid Flux Calculation

As noted, the Volume of Fluid method differs from more general methods in the calculation of $f_{h(F\Omega_{ij}^n)} = \int_{F\Omega_{ij}} fd(\mathbf{x},t)$. This must always be done using an approximation, as is obvious from a cursory examination. In the case of a Volume of Fluid method, the required additional accuracy is obtained through use of an interface reconstruction step, which provides the necessary information to obtain a higher order approximation for the face value.

A more detailed discussion of the approach used to approximate $f_{h(F\Omega_{ij}^n}$ is given in section 2.5.

2.3 The ELVIRA Interface Reconstruction Algorithm

Here we describe the ELVIRA interface reconstruction algorithm from Pilliod and Puckett (2004) in more detail. In this example we present the simplest possible case; namely, when the true interface is a line that passes through the center cell of the 3×3 block B_{ij} of cells $\Omega_{i'j'}$ centered on the cell Ω_{ij} as shown in Figure 2.4. The following description is intended to be easy to understand. However, the reader should be aware that there are many VOF interface reconstruction algorithms in both two (Torrey et al., 1985) and three dimensions (Torrey et al., 1987) and on every conceivable grid; e.g., (Korzekwa et al., 1999)

In the ELVIRA algorithm the approximate interface will be a *piecewise linear* approximation $\tilde{g}_{ij}(x) = m_{ij}\,x + b_{ij}$ to the true interface in Ω_{ij} as depicted in Figure 2.2. Furthermore *the approximate interface is subject to the constraint that* the volume fraction in the center cell due to the true interface $g(x)$ and the approximate interface $\tilde{g}_{ij}$ are equal; i.e., $f_{ij}^{true} = f_{ij}$.

Consider the example shown in Figure 2.4. In this example the true interface is a line $l(x) = m\,x + b$. Assume we are given the exact volume fractions $f_{i'j'}$ associated with the line $l(x)$, which is the true interface, in each cell $\Omega_{i'j'}$ of the 3×3 block. Then in this example the first two column sums

$$S_{i-1} \stackrel{\text{def}}{=} \sum_{j'=j-1}^{j+1} f_{i-1,j'} \qquad \text{and} \qquad S_i \stackrel{\text{def}}{=} \sum_{j'=j-1}^{j+1} f_{i,j'} \tag{2.19}$$

are *exact* in the sense that

$$S_i \;=\; \frac{1}{h^2} \int_{x_{i-1/2}}^{x_{i+1/2}} \big(l(x) - y_{j-3/2}\big)\; dx \tag{2.20}$$

and similarly for S_{i-1}, but *not* for S_{i+1}, since the line leaves the 3×3 block B_{ij} of cells $\Omega_{i'j'}$ centered on the cell Ω_{ij} through the top edge, thereby rendering the sum S_{i+1} inexact in the sense that

$$S_{i+1} \;\neq\; \frac{1}{h^2} \int_{x_{i+1/2}}^{x_{i+3/2}} \big(l(x) - y_{j-3/2}\big)\; . \tag{2.21}$$

Thus, using (2.20) we find the difference in the column sums S_i and S_{i-1} is

$$\begin{aligned}
h^2\;(S_i \;-\; S_{i-1}) &= \int_{x_{i-1/2}}^{x_{i+1/2}} (m\,x - b) \;-\; y_{j-3/2}\; dx \;-\; \int_{x_{i-3/2}}^{x_{i-1/2}} (m\,x - b) - \; y_{j-3/2}\; dx \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} m\,x\,dx - \int_{x_{i-3/2}}^{x_{i-1/2}} m\,x\,dx \\
&= m\;\frac{x^2}{2}\;\Big|_{x_{i-1/2}}^{x_{i+1/2}} \;-\; m\;\frac{x^2}{2}\;\Big|_{x_{i-3/2}}^{x_{i-1/2}} \\
&= \frac{m}{2}\left[\big(x_{i+1/2}\big)^2 - \big(x_{i-3/2}\big)^2\right] - \frac{m}{2}\left[\big(x_{i+1/2}\big)^2 - \big(x_{i-1/2}\big)^2\right] \\
&= \frac{m}{2}\,h\;\big(x_{i+1/2} - x_{i-3/2}\big) \\
&= m\;h^2\,.
\end{aligned} \tag{2.22}$$

and hence,

$$m = S_i \; - \; S_{i-1} \, . \tag{2.23}$$

Thus, we have recovered the *exact* slope m of the true interface $l(x)$ in the center cell simply by differencing the correct pair of column sums of volume fractions. Note that this would not have been true if we had used $S_{i+1} - S_i$ instead, since the expression on the RHS of (2.21) is not identically equal to S_{i+1}.

A little thought will show that the constraint

$$f_{ij} = f_{ij}^{true} \tag{2.24}$$

determines b uniquely, thus determining the linear approximation

$$g_{ij}(x) \; = \; m \, x \; + b \tag{2.25}$$

which is exactly equal to the true interface $l(x)$. In actual fact one needs to know whether the region containing the composition C_1 is above, below, or to the left or right of C_2. However, there are a variety of algorithms for doing this; e.g., see Chorin (1985); Pilliod and Puckett (2004); Puckett (2010a,b, 2014). This always works on a uniform grid of square cells with sides of length h.

However, there are a few caveats: There are three ways to difference the column sums,

$$\begin{aligned} m^{x,l} &= (S_i \; - \; S_{i-1}) \\ m^{x,c} &= \frac{(S_{i+1} \; - \; S_{i-1})}{2} \\ m^{x,r} &= (S_{i+1} \; - \; S_i) \end{aligned} \tag{2.26}$$

and three ways to difference the row sums

$$\begin{aligned} m_l^y &= (R_j \; - \; R_{j-1}) \\ m_c^y &= \frac{(R_{j+1} \; - \; R_{j-1})}{2} \\ m_l^y &= (R_{j+1} \; - \; R_j) \end{aligned} \tag{2.27}$$

where the *row sums* are defined by

$$R_{j-1} \equiv \sum_{i'=i-1}^{i+1} f_{i',j-1} \, , \quad R_j \equiv \sum_{i'=i-1}^{i+1} f_{i',j} \quad \text{and} \quad R_{j+1} \equiv \sum_{i'=i-1}^{i+1} f_{i',j+1} \tag{2.28}$$

In order to determine the best linear approximation to the true interface we compare the the volume fractions $f^{x,l}_{i'j'}$, $f^{x,c}_{i'j'}$, $f^{x,r}_{i'j'}$, ... $f^{y,r}_{i'j'}$ due to each of the six lines

$$\begin{aligned} g^x_l &= m^x_l\, x + b^x_l & \qquad g^y_l &= m^y_l\, x + b^y_l \\ g^x_c &= m^x_c\, x + b^x_c & \qquad g^y_c &= m^y_c\, x + b^y_c \\ g^y_r &= m^x_r\, x + b^x_r & \qquad g^y_r &= m^y_r\, x + b^y_r \end{aligned} \tag{2.29}$$

we obtain from each of the six slopes in (2.26) and (2.27) in the 3×3 block B_{ij} centered on the cell of interest Ω_{ij} and use the line that minimizes the difference between the given volume fractions and the volume fractions due to the lines in (2.29). We now explain this procedure in a bit more detail.

2.4 Approximating the Interface from the Volume Fractions

Suppose $g(x)$ is an unknown interface that passes through the center cell Ω_{ij} of a 3×3 block of cells B_{ij} containing nine square cells $\Omega_{i'j'}$, each of side of length h, centered on Ω_{ij}. Furthermore, assume the only information we have are the nine *exact* volume fractions $f_{i'j'}$ in the cells $\Omega_{i'j'}$ due to $g(x)$. For example, in Figure 2.2 the 'unknown' interface is $g(x) = \tanh(x)$, which is the blue curve, and the volume fractions are nonzero only in cells that either contain the curve or are below it. We want to find a line segment $\tilde{g}_{ij}(x) = m_{ij}\, x + b_{ij}$ that is a second-order accurate approximation to $g(x)$, in the following sense,

$$\max |g(x) - \tilde{g}_{ij}(x)| \le \tilde{C}\, h^2 \qquad \text{for all } x \in [x_{i-1/2},\, x_{i+1/2}]\,, \tag{2.30}$$

where $\tilde{C}$ is a constant that is independent of h.

First we define a way to measure the error $E(\tilde{m})$ between the volume fractions $f_{i'j'}$ we are given that are due to the unknown interface and the approximate volume fractions $\tilde{f}_{i'j'}$ due to a line segment $\tilde{g}(x) = \tilde{m}\, x + \tilde{b}$ that passes through the center cell Ω_{ij} and the 3×3 block B_{ij} centered on Ω_{ij},

$$E(\tilde{m}) \;=\; \sum_{i'=i-1}^{i+1} \sum_{j'=j-1}^{j+1} \left(f_{i'j'} - \tilde{f}_{i'j'} \right)^2 . \tag{2.31}$$

Note that this is the square of the *two norm* on vector spaces R^n from linear algebra, where in our case $n = 9$, (Strang, 2016).

```
For each cell
    Skip if volume fraction is 0 or 1
    Gather VOF data for neighboring 3x3 block into working array
    Compute 6 ELVIRA candidate normals, and append computational
        update of previous normal vector as 7th candidate
    For each candidate normal
        Compute correct location of interface to match center
            cell volume fraction
        For each of the cell in the 3x3 block
            Extend the interface from the center cell and
                compute the implied volume fraction
            Accumulate the square of the difference between the
                state volume fraction and the implied volume
                fraction
        If the error for this normal vector is less than the
            minimum so far, Note this as minimizing index
    Set the interface normal to the minimizing value out of the
        candidates, and record the interface to model state
```

Figure 2.5: Pseudocode for 2D ELVIRA Interface reconstruction algorithm

Now take the volume fractions we are given, namely $f_{i'j'}$, and form all six of the slopes in (2.26) and (2.27) and the six candidate lines in (2.29) from these slopes. Remember that the 'y intercept' b for each of the lines in (2.29) is determined by the constraint $f_{ij}^{true} = f_{ij}$. Each of the six lines produces nine volume fractions in the 3×3 block B_{ij}. For example, given the slope $m^{x,c}$ defined in (2.26) we obtain the line $g_c^x = m_c^x\, x + b_c^x$ defined in (2.29), which in turn gives us nine volume fractions $f_{i'j'}^{x,c}$ for $i' = i-1,\ i,\ i+1$ and $j' = j-1,\ j,\ j+1$. Now compute $E(m_c^x)$ and repeat this procedure for each of the

other lines in (2.29) with slopes computed as in (2.26) and (2.27). Finally, take the line from (2.29) that minimizes the error defined in (2.31); i.e., pick the slope from (2.26) and (2.27), call it $\tilde{m}$, that satisfies

$$E(\tilde{m}) \;=\; \min\left\{E(m_l^x),\, E(m_c^x), \ldots,\, E(m_r^y)\right\}\,. \tag{2.32}$$

The line

$$\tilde{g} \;=\; \tilde{m}\, x \,+\, \tilde{b} \tag{2.33}$$

is the linear approximation to the true interface $g(x)$ in Ω_{ij} that we use in the VOF algorithm in this dissertation. In Puckett (2010a) and Puckett (2014) it is proven that this algorithm produces a second-order accurate approximation to the true interface in the sense of (2.30) provided that

$$h \;\leq\; \frac{2}{33\,\sigma_{max}} \tag{2.34}$$

where σ_{max} denotes the maximum curvature of the true interface, h is the grid size of a square grid, and the volume fractions due to the true interface are exact.

We note that this result has only been proven for the *stationary* interface reconstruction problem. In other words, given a smooth (e.g., two times continuously differentiable) interface and the true volume fractions due to this interface on a square grid of side h, then the approximate interface is second-order accurate in the max norm; i.e., the bound in (2.30) holds. This bound is much stronger than a bound in the L^1 or L^2 norms that we use to examine convergence rates in Section 4. It is believed by experts in the field that it should be possible to prove, given the assumptions just stated above on the initial data and assuming the interface stays smooth as it moves in some flow and that (2.34) holds at each time step, then numerical approximation will remain second-order accurate in some norm; e.g., L^1 or L^2. Although this is only a conjecture, we use it as a 'rule of thumb'. It appears to work for the results shown in Section 4.3 below.

A psuedocode implementation of this reconstruction algorithm is given in Figure 2.5.

As mentioned in Section 2.1 above, our current implementation of the VOF advection method in ASPECT is a dimensionally split advection method. However, as we emphasized

in Item 3 of Section 2.1 there is an important modification we make to the algorithm described there, which is necessary in order to more nearly satisfy the conservation of the (total) volumes of C_1 and C_2. We now describe this modification. (A different, yet equivalent description is given in Section 2.2.1 of Puckett et al. (1997), which the interested reader may wish to consult for a description of this algorithm from a slightly different point of view.)

To begin, recall that the advection of the volume fraction function f is governed by equation (2.3) that, in order to improve the ease of exposition, we rewrite here,

$$\frac{\partial f}{\partial t} + \mathbf{u} \cdot \nabla f = 0\,. \tag{2.35}$$

Using the divergence free constraint on the velocity $\mathbf{u} = (u,\, v)$,

$$\nabla \cdot \mathbf{u} \;=\; 0\,, \tag{2.36}$$

we obtain a modified, but equivalent, form of (2.35),

$$\overbrace{\frac{\partial f}{\partial t} + \nabla \cdot (\mathbf{u} f)}^{\text{Conservation}} - \overbrace{f(\nabla \cdot \mathbf{u})}^{\text{Correction}} = 0\,. \tag{2.37}$$

It will be instructive to write this equation in the following form

$$\frac{\partial f}{\partial t} \;+\; \nabla \cdot (\mathbf{u}\, f) \;=\; f\, \frac{\partial u}{\partial x} \;+\; f\, \frac{\partial v}{\partial y}\,. \tag{2.38}$$

Note that in equation (2.37) the first term is the conservation equation (2.1) for f. The key point is that if (2.36) is satisfied *exactly*, then the correction term in (2.37) will be zero and hence, the advection equation in (2.35) for f is equivalent to the conservation equation for f, which here we write in the following form,

$$\frac{\partial f}{\partial t} \;+\; \frac{\partial (u f)}{\partial x} \;+\; \frac{\partial (v f)}{\partial y} = 0\,. \tag{2.39}$$

In the dimensionally split advection algorithm that we described in Step II of the Volume Fraction Update in Section 2.1 above we approximated solutions of following two step dimensionally split advection method,

$$\begin{aligned} \frac{\partial f}{\partial t} + \frac{\partial (u\, f)}{\partial x} \;&=\; 0\,, \\ \frac{\partial f}{\partial t} + \frac{\partial (v\, f)}{\partial y} \;&=\; 0\,. \end{aligned} \tag{2.40}$$

However, unless each component of

$$\nabla \cdot \mathbf{u} \;=\; u_x + v_y \,,$$

equals zero separately (i.e., $u_x = 0$ and $v_y = 0$) numerical solutions of the equations in (2.40) above will fail to be adequate approximate solutions of the advection equation (2.35) for f. The reason for this is that in passing from (2.35) to (2.37) and thence to (2.39) we assumed that the velocity $\mathbf{u}$ was divergence free (2.36).

In short, we need to approximate solutions of the dimensionally split version of the *modified* equation (2.38),

$$\begin{aligned} \frac{\partial f}{\partial t} + \frac{\partial (u\, f)}{\partial x} \;&=\; f\, \frac{\partial u}{\partial x} \,, \\ \frac{\partial f}{\partial t} + \frac{\partial (v\, f)}{\partial y} \;&=\; f\, \frac{\partial v}{\partial y} \,. \end{aligned} \tag{2.41}$$

In the dimensionally split algorithm that we use in this dissertation we approximate solutions of (2.41) as follows,

$$\begin{aligned} \tilde{f}_e \;&=\; f_e^k \;-\; \Delta t\, \frac{\partial (u\, f)}{\partial x} \;+\; \Delta t\, \tilde{f}_e\, \frac{\partial u}{\partial x} \,, \\ f_e^{k+1} \;&=\; \tilde{f}_e \;-\; \Delta t\, \frac{\partial (v\, \tilde{f})}{\partial x} \;+\; \Delta t\, \tilde{f}_e\, \frac{\partial v}{\partial y} \,. \end{aligned} \tag{2.42}$$

Note that we have written these equations in semi-discrete form; i.e., discretized in time but not in space. In (2.42) $\tilde{f}_e$ is the intermediate volume fraction of C_1 in Ω_e and $(v\, \tilde{f})$ is the intermediate volume fraction flux in the y direction obtained from the intermediate interface $\hat{g}$ that has been reconstructed from the volume fractions $\tilde{f}_{e'}$ surrounding Ω_e as explained in Step II of 3 "The Volume Fraction Update"in Section 2.1 above. We have written the equations in (2.42) in this semidiscrete form in order to emphasize that in the first equation $\tilde{f}_e$ is treated *implicitly* while in the second equation $\tilde{f}_e$ is treated explicitly.

Our complete discretization of (2.42) is as follows,

$$\begin{aligned} \tilde{f}_e\, V_e \;&=\; f_e^k\, V_e \;-\; \Delta t\; (f_R\, U_R \;-\; f_L\, U_L) \;+\; \Delta t\; \tilde{f}_e\; (U_R \;-\; U_L) \;, \\ f_e^{k+1}\, V_e \;&=\; \tilde{f}_e\, V_e \;-\; \Delta t\; \left(\tilde{f}_T\, U_T \;-\; \tilde{f}_B\, U_B \right) \;+\; \Delta t\; \tilde{f}_e\; (U_T \;-\; U_B) \;, \end{aligned} \tag{2.43}$$

where e is an index that ranges over all cells Ω_e, V_e is the volume of Ω_e, $r = L, R, B, T$ denotes the left, right, bottom, and top cell edges, respectively, f_r is the volume *fraction* of C_1 that will be fluxed across the rth edge as described in the caption to Figure 2.8, and

$$U_r = \int_{\partial\Omega_{e,r}} \tilde{\mathbf{u}}_r \cdot \mathbf{n}_r \, ds \,, \tag{2.44}$$

where $\tilde{\mathbf{u}}_r$ is a time centered approximation to the velocity $\mathbf{u}$ on the rth edge,

$$\tilde{\mathbf{u}}_r = \frac{\mathbf{u}^{k+1} + \mathbf{u}^k}{2} \,.$$

Again, note that in the dimensionally split algorithm in (2.43) the intermediate value $\tilde{f}_e$ in the first equation is determined via an implicit discretization, while $\tilde{f}_e$ is treated explicitly in the second equation.

There are a number of versions of VOF advection methods in the peer reviewed VOF literature that are similar —or in at least one case —identical to our modified algorithm (2.43). For example, on a grid of square cells of side h our method is identical to the method in Section 2.2.1. of Puckett et al. (1997); i.e., equations (22)–(23). Our algorithm is similar to the the one presented in Section "*4.2 Eulerian Scheme*" in Scardovelli and Zaleski (2003) and attributed to the authors of Rider and Kothe (1998), except in both papers the first step of the algorithm is explicit while the second step is implicit, which is the opposite of our algorithm in (2.43). We also note that —assuming we understand the authors notations correctly —the monodimensional Eulerian-implicit (EI) scheme in equation (18) of Aulisa et al. (2007) is identical to our first (implicit) step in (2.43) and the first (implicit) step in equation (22) of Puckett et al. (1997).

Finally, since we are using Strang splitting, Strang (1968) we evaluate (2.43) once for each spatial dimension in the problem at each time step, alternating the order of the dimensions in the subsequent time step.

2.5 The Implementation of the VOF method in ASPECT

We now describe our implementation in ASPECT of the VOF algorithm described above on square, two-dimensional cells Ω_e in physical space (Ω_e is often referred to as the 'real'

```
If iteration number is even:
    Assemble matrix for X dir
    Solve using Jacobian Preconditioned Conjugate Gradient
    Reconstruct Interface
    Assemble matrix for Y dir
    Solve using Jacobian Preconditioned Conjugate Gradient
    Reconstruct Interface
else:
    Assemble matrix for Y dir
    Solve using Jacobian Preconditioned Conjugate Gradient
    Reconstruct Interface
    Assemble matrix for X dir
    Solve using Jacobian Preconditioned Conjugate Gradient
    Reconstruct Interface
end
Update FEM approx of VOF interface state
```

Figure 2.6: Volume-of-Fluid time step update algorithm

cell). Pseudocode sketches of the structure of the algorithm are given in Figures 2.6 and 2.7. In a VOF method it is natural to use the method of characteristics to calculate the flux of C_1 through each of the cell edges. This is done by tracing backward in time along a linear approximation to each characteristic that crosses the cell edge in the time interval $[t^k, t^{k+1}]$ in order to identify the *total* volume V_F of both compositions C_1 and C_2 that will cross a given edge in the time interval $[t^k, t^{k+1}]$. We then compute that portion of the volume associated with the composition C_1 that is being tracked; i.e., by computing the volume of C_1 in the total volume V_F. This procedure is depicted in Figure 2.3 for a one-dimensional sweep in the x-direction, in which case the linear approximation to the characteristics that cross the right edge of Ω_e are horizontal lines of length $u_{i+i/2,j}\,\Delta t$ that fill out the rectangle (shown in green and pink) of volume $V_F = u_{i+i/2,j}\,\Delta t\,h$. See Colella (1990); LeVeque (1996)

```
For each cell
    Compute cell volume by quadrature
    For each edge in the current split direction
        If this cell should assemble for this edge
            Get neighbor data
            Compute fluid volume fluxed through edge
            Identify which cell is upwind of the current cell
               edge
            Use volume fluxed to compute volume fraction on
               fluxed region for upwind cell
            Update assembly for both volume correction and RHS
```

Figure 2.7: Algorithm used for split VOF advection update

for examples of computing a second-order accurate flux in this manner in a finite volume discretization of (3.26), rather than a VOF discretization of (3.26) or, equivalently, (2.1), as well as higher resolution versions of these algorithms. In our computation of the volume and volume fraction flux we make use of several algorithms that we developed for the interface reconstruction step. We will describe these algorithms is more detail below.

There are a number of approaches one can consider for obtaining the velocities on the rth edge from the approximate FEM solution of the incompressible Stokes equations. Two such approaches are

1. A point sample of the velocity normal to the rth edge $\partial\Omega_{e,r}$ taken at some point on $\partial\Omega_{e,r}$,

2. The velocity integrated along the rth edge $\partial\Omega_{e,r}$ of Ω_e,

$$\int_{\partial\Omega_{e,r}} \mathbf{u}\cdot\mathbf{n}_r \, ds \,, \tag{2.45}$$

   where $\mathbf{n}_r$ denotes the (outward facing) unit normal to $\partial\Omega_{e,r}$ and $k = 1,\ 2,\ 3,$ or 4.

For a finite volume method both are reasonable approximations to the edge velocities.

However, the latter method (2.45) is a closer analogue to the type of procedure one would typically choose for a finite element method.

We now describe our implementation of the computation of the volume flux of C_1 into or out of a square cell Ω_e of side h. (When we employ AMR, h denotes the length of each side of the most finely resolved cells in the FEM grid.) First, note that all of the information that describes the interface; namely, its distance d_e to the center of the cell and the unit normal $\mathbf{n}_e$ to the interface, is stored with respect to the center of the unit cell $\tilde{\Omega}_e$ as depicted in Figure 2.8. In particular, the interface in the unit cell $\tilde{\Omega}_e$ is given by

$$\mathbf{n}_{\tilde{e}} \cdot (\mathbf{x} - \mathbf{x}_{\tilde{e}}^c) = d_{\tilde{e}} \tag{2.46}$$

where $\mathbf{x}_{\tilde{e}}^c$ is the center of $\tilde{\Omega}_e$, $d_{\tilde{e}}$ is the distance of the (mapped) interface from the center $\mathbf{x}_{\tilde{e}}^c$ of $\tilde{\Omega}_e$, and $\mathbf{n}_{\tilde{e}}$ is a unit vector that is perpendicular to the reconstructed linear interface in $\tilde{\Omega}_e$, with the convention that $\mathbf{n}_{\tilde{e}}$ *always points away* from the region containing C_1. The location of the interface is stored by recording $\mathbf{n}_{\tilde{e}}$ and $d_{\tilde{e}}$ for each cell Ω_e that contains a portion of the interface. For the case when the velocity field is perpendicular to a cell edge, say $\partial\Omega_{e,r}$, for some $k = 1, 2, 3, 4$, let $\tilde{\mathbf{n}}_r$ be the outward facing unit normal vector to the rth edge $\partial\tilde{\Omega}_{e,r}$ of the unit cell $\tilde{\Omega}_e$, and, as above, let V_F denote the *total* volume flux that will cross $\partial\Omega_{e,r}$; i.e., the volume flux of C_1 plus the volume flux of C_2.

As shown in Figure 2.8, with only a few computationally inexpensive transformations we can use the same algorithm we used to compute the volume fraction on a cell Ω_e in the reconstruction step to compute the volume flux of C_1 across each of the edges of Ω_e. If we map the rectangular region of size V_F which we expect to pass through the cell edge from $\tilde{\Omega}_e$ to another unit cell $\tilde{\Omega}_I$ and assume the velocity is perpendicular to the rth cell edge $\partial\tilde{\Omega}_{e,r}$ of Ω_e, we find that the line describing the interface within the unit cell $\tilde{\Omega}_I$ is given by

$$\mathbf{n}_I \cdot (\mathbf{x} - \mathbf{x}_I^c) = d_I$$

where $\mathbf{x}_I^c$ is the center of $\tilde{\Omega}_I$ as shown in Figure 2.8. The values of $\mathbf{n}_I$ and d_I in terms of

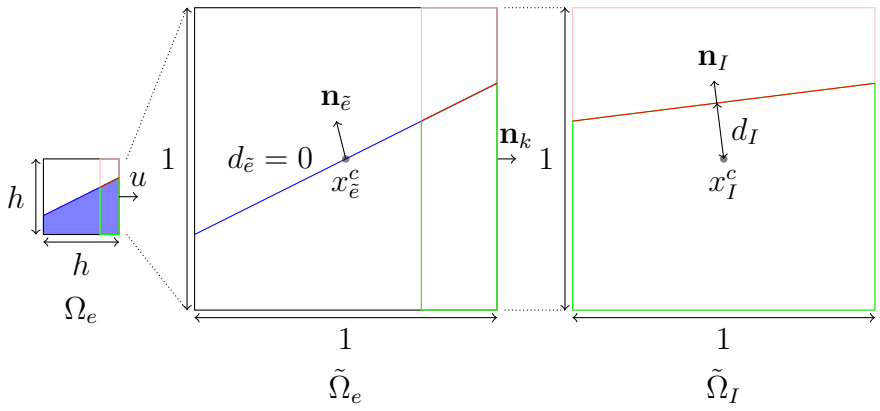


Figure 2.8: A diagram of the mapping of the region (in purple) containing the compositional field C_1 in the *real* cell Ω_e to its associated unit cell $\tilde{\Omega}_e$. In this diagram we have assumed that the velocity field $\mathbf{u} = (u, 0)$ points in the x direction only so that the flux of C_1 across the right edge of $\tilde{\Omega}_e$ is a rectangular region. This allows us to compute the *total* volume V_F of C_1 and C_2 that is fluxed across the right edge of $\tilde{\Omega}_e$; namely, the rectangle on the right edge of $\tilde{\Omega}_e$. We then map this rectangle to another unit cell $\tilde{\Omega}_I$ in order to compute the volume *fraction* f_r of the (mapped) rectangle that contains the composition C_1. Since in this dissertation, linear interfaces map to linear interfaces, we can use the unit normal $\mathbf{n}_I$ and distance d_I to calculate the volume fraction f_r of C_1 in this rectangle. Note that, since this diagram has been chosen to correspond exactly to the one in Figure 2.3 in which the interface and reconstructed interface both pass through the cell center, we have $d_{\tilde{e}} = 0$. However, in general, $d_{\tilde{e}} \neq 0$.

$\mathbf{n}_e$, $\mathbf{n}_k$, and d_e are given by

$$\mathbf{n}_I = \mathbf{n}_{\tilde{e}} + (\frac{V_F}{V_e} - \mathbf{n}_{\tilde{e}} \cdot \mathbf{n}_k)\, \mathbf{n}_k \,,$$
$$d_I = d_{\tilde{e}} - (\frac{1}{2} + \frac{V_F}{2V_e})(\mathbf{n}_{\tilde{e}} \cdot \mathbf{n}_k) \,.$$

where V_e is the volume of Ω_e (the upwind cell for this edge), and $\mathbf{n}_k$ is the outward pointing unit normal to the cell edge $\partial\tilde{\Omega}_{e,r}$.

Since we are computing on a uniform square grid, we have a constant Jacobian and hence, the volumes on the unit cell and the volumes in physical space are related by a constant multiple. With this constraint, for a given interface, there is a simple formula to calculate the volume of C_1 on the side opposite the unit normal $\mathbf{n}$; e.g., see Scardovelli and Zaleski (2000), which we will cover in Section 2.6. We use (2.48) to compute the volume flux of C_1 across the right edge, which is $f(\mathbf{n}_I, d_I)\, V_F$. We use an analogous procedure to compute the flux of C_1 across the other three edges $\Omega_{e,r}$ of Ω_e.

Now let $V_{F,r}$ and $f_r\, V_{F,r}$ denote the total volume flux and the volume flux of C_1 across the rth edge $\Omega_{e,r}$ of Ω_e, respectively. We can now use these quantities to write our *modified* dimensionally split update $f_e^k \to f_e^{k+1}$ of the volume fraction of C_1 in Ω_e in a slightly different form than we did in equation (2.43),

$$\begin{aligned} \tilde{f}_e\,(V_e - \sum_r V_{F,r}) \;&=\; f_e^k\, V_e \;+\; \sum_r f_r\, V_{F,r} \\ f_e^{k+1}\, V_e \;&=\; \tilde{f}_e\, V_e \;+\; \sum_r f_r\, V_{F,r} \;+\; \tilde{f}_e \sum_r V_{F,r}\,. \end{aligned} \tag{2.47}$$

where r only runs over the cell faces on the unit cell that are perpendicular to the direction of that particular sweep. Furthermore, as mentioned before, the order of the sweep directions are alternated at each time step in order to achieve second-order accuracy. Finally, note that the simplified version of these equations; i.e., without the terms $\tilde{f}_e \sum_r V_{F,r}$ in each equation of (2.47), are essentially identical to the equations (2.8)–(2.10), although with a slightly different notation.

2.6 Volume Fraction Computation

A critical component in both the reconstruction and advection steps of the algorithm is the ability to calculate the volume and thus volume fraction implied by an interface on particular types of regions. In particular, the two categories of regions that need to be handled are the discretization cells, and the regions used to compute the flux volume fractions. Additionally, the reconstruction step requires the ability to compute the correct location for an interface with a given normal vector to match a specified volume fraction.

Under the restricted case of real space regions that can be mapped to a unit cell with a mapping that produces a constant Jacobian, the necessary calculation may be done using a formula derived in Scardovelli and Zaleski (2000), and mentioned earlier in section 2.5.

In our notation this formula is,

$$f(\mathbf{n},d)=\begin{cases}1 & \frac{1}{2}\leq\bar{d}\\ 1-\frac{(\bar{d}-\frac{1}{2})^2}{2m(1-m)} & \frac{1}{2}-m<\bar{d}<\frac{1}{2}\\ \frac{1}{2}+\frac{\bar{d}}{(1-m)} & m-\frac{1}{2}\leq\bar{d}\leq\frac{1}{2}-m\\ \frac{(\bar{d}+\frac{1}{2})^2}{2m(1-m)} & -\frac{1}{2}<\bar{d}<m-\frac{1}{2}\\ 0 & \bar{d}\leq-\frac{1}{2}\end{cases} \tag{2.48}$$

where $m=1-\frac{\|\mathbf{n}\|_\infty}{\|\mathbf{n}\|_1}$ and the components of $\mathbf{n}$ are parallel to sides of the unit cell $\tilde{\Omega}_e$, and $\bar{d}=\frac{d}{\|\mathbf{n}\|_1}$.

While the above equation 2.48 is sufficient for the cases appropriate to the current advection and reconstruction algorithms, the ability to extend the implementation to more complex mesh geometries is a natural additional goal. Due to the volume fraction calculation being sufficiently fundamental, it is a natural starting point. The approach that was finally selected was partially inspired by the xFEM Heaviside function implementation in Ventura (2006).

Given a set of basis functions ψ_i, there will be some function $\tilde{H}_{\mathbf{n},d}(\hat{\mathbf{x}})$ that can be constructed in the span of ψ_i such that

$$\int_{\hat{\Omega}_e} H(d-\mathbf{n}\cdot\hat{\mathbf{x}})\psi_i d\hat{\mathbf{x}}=\int_{\hat{\Omega}_e}\tilde{H}_{\mathbf{n},d}(\hat{\mathbf{x}})\psi_i d\hat{\mathbf{x}} \tag{2.49}$$

If the determinant of the Jacobian for the mapping to model space can be ensured to be within the span of ψ_i, then the use of $\tilde{H}$ in the place of the interface defined indicator function will allow standard integration by quadrature without loss of accuracy. The construction of $\tilde{H}$ can be done using the formulas

$$\tilde{H}_{\mathbf{n},d}(\hat{\mathbf{x}})=C_{\mathbf{n},d,i}\psi_i\,, \tag{2.50}$$

$$C_{\mathbf{n},d,i}=\frac{\int_{\hat{\Omega}_e} H(d-\mathbf{n}\cdot\hat{\mathbf{x}})\psi_i d\hat{\mathbf{x}}}{\int_{\hat{\Omega}_e}\psi_i^2 d\hat{\mathbf{x}}}\,. \tag{2.51}$$

$$\tag{2.52}$$

While the above construction obviously satisfies equation 2.49, it does require the ability to integrate a Heaviside function on the unit cell, which we wished to avoid in the main computational model due to both complexity and calculation intensity. It is therefore necessary to be able to obtain $C_{\mathbf{n},d,i}$ in a manner that allows the integration of the Heaviside function to be done outside of the main computational loop. The most reasonable approach is to make use of an external computer/symbolic algebra system to find the correct calculations for the coefficients $C_{\mathbf{n},d,i}$, and then use the derived formulas in the main computation. This approach does impose practical limits on what spaces ψ_i are reasonable to implement, based on the number of coefficients for which it is reasonable to maintain the formulas.

2.7 The Model Coupling Procedure

```
For each cell
    Obtain the FEM composition field unit cell support points
       from Deal.II
    Use the specified heuristics to compute the value of the
       composition field at each unit cell location
    Set the composition field element to the corresponding value
```

Figure 2.9: Algorithm for setting the FEM composition approximation to the true interface data

Having now described our implementation the VOF method in ASPECT, it is necessary to establish how the computed fluid interface is presented as a so-called *compositional* field C in ASPECT, which will be used by the Finite Element methodology in cases where the tracked fluid is an *active* part of the problem, such as the density, or the viscosity, or both. For example, see Puckett et al. (2018) where the density is tracked in ASPECT by placing its values on *active* tracer particles. The psuedocode structure of the algorithm used is given in Figure 2.9.

In order to be compatible with the existing infrastructure for advecting compositional fields in ASPECT, it proved to be most efficient to present the results of the VOF

method as a traditional continuous or discontinuous Galerkin FEM field to the rest of the software. Therefore, the location of the tracked composition is presented as a finite element approximation to the characteristic function implied by the reconstructed interface (i.e., $\chi(d_e - \mathbf{n}_e \cdot (\tilde{\mathbf{x}} - \tilde{\mathbf{x}}_c))$), as one of the "compositional fields" in ASPECT.

In order to avoid additional complexity due to interfering with the field values on neighboring cells, we require that the finite-element field used for the approximation use a discontinuous finite-element discretization; e.g., discontinuous P_q (polynomial elements of maximum total degree q) or discontinuous Q_q (polynomial elements with each individual variable having maximum exponent q) elements, which are typically denoted as P_{-q} and Q_{-q} elements. In this paper we explicitly consider cases suitable for use with discontinuous P_0 and discontinuous Q_1 elements, and make use of the latter approach. For a number of reasons, often relating to the physical interpretation of the quantity C, it is also desirable to ensure that the generated approximation will always be bounded; i.e., $0 \leq C \leq 1$. Among the more obvious reasons for this requirement are physical constraints, such as the density must satisfy $\rho > 0$.

A basic implementation can be done by directly copying the volume fraction data to a discontinuous P_0 element (i.e. the value of the discretized variable is constant on each cell). This is equivalent to a minimum L^2 error approximation when using the discontinuous P_0 element to approximate the indicator field implied by the reconstructed interface.

However, attempting to obtain an ideal (minimum L2 error) approximation using a higher order element such as DG Q_1 or DG P_1 is more difficult, especially when we wish to respect the bounds on the compositional fields $0 \leq C \leq 1$, since the result of a minimum L_2 error approximation for such an element is almost certain to violate the $0 \leq C \leq 1$ bounds in all non-trivial cases. Also, a basic minimum L_2 error approximation for the indicator function would require significant additional computational expense and code complexity. Thus, any approximation using a non-constant element would best be done using a heuristic approach.

In our implementation, in order to generate a DG Q_1 element approximation to the C field that is implied by the reconstructed interface, we apply the following constraints.

1. The gradient of the element is in the same direction as the normal of the interface.

2. The gradient is as large as possible while maintaining $0 \leq C \leq 1$ everywhere.

3. In order to conserve the mass in this step, the volume fraction implied by the DG Q_1 element approximation to the C field must match the volume fraction f_e in the VOF approximation to the C field; i.e.,

$$\int_{\Omega_e} C(\mathbf{x})\, d\mathbf{x} \;=\; f_e\, V_e$$

where V_e is the volume of Ω_e.

On a square mesh, for a cell with the reconstructed interface

$$\mathbf{n}_{\tilde{e}} \cdot (\mathbf{x} \;-\; \mathbf{x}^c_{\tilde{e}}) \;=\; d_{\tilde{e}} \tag{2.53}$$

the above constraints result in the approximation on the unit cell being

$$C(\mathbf{x}) \;=\; f_{\tilde{e}} \;-\; 1 - |\, 2\, f_{\tilde{e}} \;-\; 0.5| \;\; \frac{\mathbf{n}_{\tilde{e}}}{\|\mathbf{n}_{\tilde{e}}\|_1} \cdot (\mathbf{x} \;-\; \mathbf{x}^c_{\tilde{e}}) \tag{2.54}$$

Since we use a DG Q_1 element the above equation produces a bilinear approximation to the VOF method's reconstructed indicator function, with little additional computational cost as compared to using a P_0 approximation.

2.8 Coupling with the AMR Algorithm

The deal.II library (Arndt et al., 2017b) upon which ASPECT is built manages the AMR algorithm through the p4est library (Burstedde et al., 2011b). Deal.II, and hence, ASPECT provides a mechanism for setting the refinement criteria; both when to refine a cell and when to coarsen a cell. Since reconstructing and advecting the interface across different levels of refinement both increases algorithm complexity and decreases the accuracy with which the interface is resolved, in this work we ensure that the interface is always on the finest level of refinement. This approach requires that the cells that contain the interface, including the case where the interface is on a cell boundary, and any cell that shares a vertex with any of those cells must also be at the finest level of refinement.

The criteria for refining a cell that we have adopted is a two step algorithm that requires one pass over the entire mesh and one pass over a subset of the entire mesh. In the first step we check every cell in the entire mesh making a list of all cells that contain a part of the interface. More specifically, we regard all cells Ω_e that satisfy $\epsilon_{vof} < f_e < 1 - \epsilon_{vof}$, where ϵ_{vof} is a small parameter, to contain a portion of the interface. In addition, all cells Ω_e that have a neighboring cell Ω'_e that shares a face with Ω_e and differ in volume fraction sufficiently (e.g., $|f_e - f'_e| > \epsilon_{vof}$) are also added to this list. In the computational results shown in Chapters 4 and 5 we use the value $\epsilon_{vof} = 10^{-6}$. We note here that if working with a distributed mesh, the cell must also be added to the list on the neighboring mesh regions where it appears as a ghost cell for the next step to behave consistently with local meshes. In the second pass over a subset of the entire grid we make a list of all cells that share a vertex with any cell already in the list of cells that contain a portion of the interface and also flag each of these cells for refinement. These flags are then passed to deal.II and thus on to p4est (Burstedde et al., 2011b), which handles the details of the refinement of these cells and the coarsening of those cells that no longer need to be at the finest level of refinement.

Recall that the CFL (Courant-Friedrichs-Lewy) condition is a constraint on the time step Δt that is typically given by

$$\Delta t \leq \sigma \, \frac{h_e}{\|\mathbf{u}\|_\infty} \qquad \text{for all cells } e \text{ in the computation}\,. \tag{2.55}$$

Here the dimensionless constant $\sigma \leq 1$ is the CFL number, h_e is the characteristic size of cell e, and $\|\mathbf{u}\|_\infty$ is the maximum value of the velocity over the entire domain Ω. (See Courant et al. (1967); John (1978); Lax (1967) for more details and, in particular, for an explanation as to why $\sigma \leq 1$ for explicit numerical approximations to the solution of advection equations.) The constraint in (2.55) is required to hold at each time step t^k, which implies that in most computations Δt must be recomputed at each time step, since the magnitude of the velocity $\mathbf{u}$ may have changed from time t^k to time t^{k+1}. In addition, when AMR is a part of the computation, the minimum or maximum value of h_e may also have changed during the time step.

In ASPECT the constraint in (2.55) is modified by dividing the right-hand side by the

order p of the polynomial basis functions that are used to discretize the velocity field

$$\Delta t \leq \sigma \, \frac{h_e}{p \, \|\mathbf{u}\|_\infty} \qquad \text{for all cells } e \text{ in the computation}\,. \tag{2.56}$$

The rational behind this formula is that the number of nodes in a given cell at which a value of the velocity is specified depends on the degree of the polynomial. In other words, in a finite element method the nodes are locations at which the unknowns, such as velocity, are specified just as the corners or centers of a cell in a finite difference or a finite volume method are locations at which the unknowns are specified. Thus, the distance between the nodes $\frac{h_e}{p}$ is analogous to the distance, say Δx, between grid points in a finite difference or finite volume method, for which one would use the formula in (2.55). (See the section entitled "Numerical experiments to determine optimal parameters" of the reference documentation for deal.II (Kronbichler and Bangerth, 2011) for further information.)

Given that the time step Δt is constrained by (2.56), the interface can move at most $\sigma \leq 1$ cell widths in one time step $t^k \to t^{k+1}$. This permits the reduction of the frequency with which we must conduct the remeshing procedure to N time steps where $N < \frac{W-2}{2\sigma}$ and W is the minimum width of the maximally refined band of cells. (See, for example, any of the AMR computations in the second (b) and fourth (d) frames in Figures 5.1–5.2 for explicit examples of W.) For the refinement strategy described above, the safest assumption is that $W = 4$. This takes into consideration the case where the interface is at the cell boundary. A band of larger width $W > 4$ would both require a more complex algorithm to find the necessary cells to flag and would increase the number of refined cells. Thus, there is a balance between cost associated with the frequency of running the algorithm to flag cells for refinement and cost of having a larger value of W. This balance is problem dependent.

Chapter 3

THE GOVERNING EQUATIONS OF CONVECTION AND THEIR DISCRETIZATION

3.1 Thermochemical Convection in the Mantle with Two Materials

In this section we present the equations associated with the model problem concerning LLSVPs (Large Low Shear Velocity Provinces), which we briefly described above. The reason for organizing this dissertation in this order is that when one uses a VOF algorithm it is necessary to supplement the dependent variables with a passively advected 'compositional' variable, which will be used to indicate the location of the material interface between two distinct materials at each time step. We therefore find it convenient to introduce the equations as formulated for a problem with two distinct materials, such as the one we have posed for modeling LLSVPs.

In the VOF literature the compositional variable is typically denoted f for volume *fraction*. However, in the mantle convection literature (and sometimes in the VOF literature) it is denoted C for *composition*. No matter what the notation we always have $0 \leq C,\ f\ \leq 1$ and C (or f) is passively advected with the flow; e.g.,

$$\frac{\partial C}{\partial t} + u\frac{\partial C}{\partial x} + v\frac{\partial C}{\partial y} = 0\,, \tag{3.1}$$

and similarly for f. Thus we augment the standard incompressible equations for mantle convection (e.g., (Schubert et al., 2001)) with (3.1) where it is understood that C may be replaced by f as appropriate.

After giving an overview of the numerical methodology we use to approximate solutions of these equations in Section 3.2, with an in-depth description of how we implement the VOF algorithm in ASPECT having been covered in Chapter 2 we present a sequence of computations of the model problem using the VOF methodology in 5.1.

3.1.1 The Dimensional Form of the Equations

In order to study the efficacy of our implementation of a VOF algorithm in ASPECT to model processes that occur in the Earth's mantle, we compute problems that emphasize the effect of having two materials or compositions. Here we present a set of equations to model thermochemical convection in the mantle with two materials that have a density difference. In Section 5.1 we will use these equations as they are presented here to study the effect that a density difference between a stratified layer has on thermal convection.

We consider a two-dimensional flow in a horizontal fluid layer with a thickness or height d. Our problem domain Ω has width $3\,d$ and height d. At a given reference temperature T_0 the region $d/2 < y \leq d$ has a compositional density of ρ_0 and the region $0 \leq y < d/2$ has a compositional density of $\rho_0 + \Delta\rho$ where $\Delta\rho \ll \rho_0$.

We also introduce a composition variable $C(x, y, t)$ defined by

$$C = \frac{\rho - \rho_0}{\Delta\rho}\,. \tag{3.2}$$

The composition C is the concentration of the dense fluid as a function of space and time. The initial condition for C is

$$C(x, y, t = 0) \;=\; \begin{cases} 1 \quad \text{for} \quad 0 \leq y \leq d/2\,, \\ 0 \quad \text{for} \quad d/2 < y \leq d\,. \end{cases} \tag{3.3}$$

The upper boundary, at $y = d$, has temperature T_0 and the lower boundary at $y = 0$ has temperature T_1. The fluid is assumed to have a constant viscosity μ, which is large. The Prandtl number is assumed to be very large,

$$\mathrm{Pr} \;=\; \frac{\mu}{\rho_0 \kappa} \;\gg\; 1\,, \tag{3.4}$$

where κ is the thermal diffusivity, so that inertial effects can be neglected. The fluids in the high density and low density layers are immiscible; i.e., they cannot mix by diffusion. Similarly, the Lewis number is also assumed to be large,

$$\mathrm{Le} \;=\; \frac{\kappa}{D} \;\gg\; 1\,, \tag{3.5}$$

where D is the diffusion coefficient for the compositional variable C. Thus, the discontinuous boundary between the high density and low density fluids is preserved indefinitely.

The problem we have posed requires the solution of the standard equations for thermal convection with the addition of an equation for the compositional field C that tracks the density field. The governing equations are described in detail in Schubert et al. (2001); Turcotte and Schubert (2014).

We make the assumption that the Boussinesq approximation

$$\rho(x,y,t) \;=\; \rho_0\,(1-\alpha\;(T-T_0)) \;+\; \Delta\rho\,C\,, \tag{3.6}$$

holds; namely, that density differences associated with convection $\rho_0\;\alpha\;(T_1-T_0)$ and $\Delta\rho$ are small compared with the reference density ρ_0.

Conservation of mass requires

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \tag{3.7}$$

where x and y denote the horizontal and vertical spacial coordinates, oriented as shown in Figure 3.1, and u and v denote the horizontal and vertical velocity components, respectively. We use the Stokes equations

$$0=\frac{-\partial P}{\partial x}+\mu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)\,, \tag{3.8}$$

$$0=\frac{-\partial P}{\partial y}+\mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)+\rho_0\alpha(T-T_0)g-\Delta\rho\;C\;g\,, \tag{3.9}$$

where α is the coefficient of thermal expansion, g is the gravitational acceleration in the negative (downward) y-direction as shown in Figure 3.1, and

$$P \;=\; p+\rho_0\,g\,y$$

where p is the dynamic pressure and $\rho_0\, g\, y$ is the isostatic pressure. Conservation of energy requires

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} = \kappa\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right), \tag{3.10}$$

where κ is the thermal diffusivity.

When there is no compositional diffusion, i.e., $D = 0$, the composition variable C satisfies the advection equation

$$\frac{\partial C}{\partial t} + u\frac{\partial C}{\partial x} + v\frac{\partial C}{\partial y} = 0\,. \tag{3.11}$$

3.1.2 The Nondimensional Form of the Equations

We introduce the nondimensional variables

$$\begin{aligned} x' &= \frac{x}{d}, & y' &= \frac{y}{d}, & t' &= \frac{\kappa}{d^2}\,t, \\ u' &= \frac{d}{\kappa}\,u, & v' &= \frac{d}{\kappa}\,v, & \rho' &= \frac{\rho}{\rho_0}, \\ T' &= \frac{T - T_0}{T_1 - T_0}, & P' &= \frac{d^2 P}{\mu\kappa}\,, \end{aligned} \tag{3.12}$$

and the two nondimensional parameters, the Rayleigh number Ra and the buoyancy ratio B

$$\mathrm{Ra} = \frac{\rho_0\, g\, \alpha\, (T_1 - T_0)\, d^3}{\mu\, \kappa}\,, \tag{3.13}$$

$$\mathrm{B} = \frac{\Delta\rho}{\rho_0\, \alpha\, (T_1 - T_0)}\,. \tag{3.14}$$

where $g = 9.80665\, m/s^2$ is the acceleration due to gravity.

Substitution of equations (3.12)–(3.14) into equations (3.7)–(3.11) gives

$$\frac{\partial u'}{\partial x'} + \frac{\partial v'}{\partial y'} = 0\,, \tag{3.15}$$

$$0 = \frac{-\partial P'}{\partial x'} + \frac{\partial^2 u'}{\partial x'^2} + \frac{\partial^2 u'}{\partial y'^2}\,, \tag{3.16}$$

$$0 = \frac{-\partial P'}{\partial y'} + \frac{\partial^2 v'}{\partial x'^2} + \frac{\partial^2 v'}{\partial y'^2} + \mathrm{Ra}\, T' - \mathrm{Ra}\,\mathrm{B}\, C\,, \tag{3.17}$$

$$\frac{\partial T'}{\partial t'} + u'\frac{\partial T'}{\partial x'} + v'\frac{\partial T'}{\partial y'} = \frac{\partial^2 T'}{\partial x'^2} + \frac{\partial^2 T'}{\partial y'^2}\,, \tag{3.18}$$

$$\frac{\partial C}{\partial t'} + u'\frac{\partial C}{\partial x'} + v'\frac{\partial C}{\partial y'} = 0\,. \tag{3.19}$$

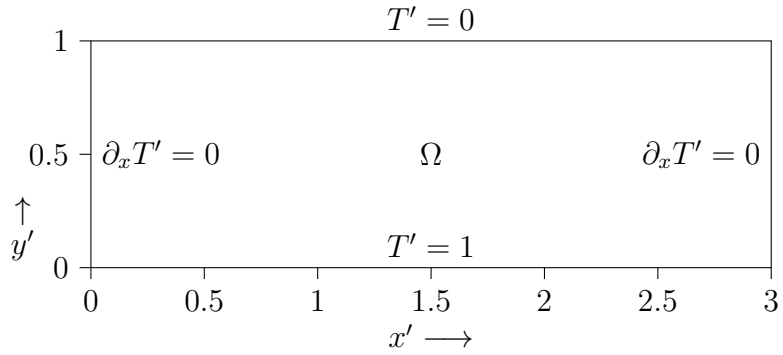


Figure 3.1: The geometry of the (nondimensional) computational domain Ω shown with the temperature boundary conditions on the four side walls. The velocity boundary conditions on the side walls are $\mathbf{u} \cdot \mathbf{n} = 0$ (no flow) and $\partial \mathbf{u}/\partial \boldsymbol{\tau} = \mathbf{0}$ (free slip) where $\mathbf{n}$ and $\boldsymbol{\tau}$ are the unit normal and tangential vectors to the boundary respectively.

This is the superposition of a Rayleigh-Taylor problem and a Rayleigh-Bénard problem (Chandrasekhar, 1961; Turcotte and Schubert, 2014). In the isothermal limit, $T_0 = T_1$, it is the classic Rayleigh-Taylor problem. If C is positive, a light fluid is above the heavy fluid and in a downward gravity field the fluid layer is stable. If $\Delta\rho$ is negative, a heavy fluid lies over a light fluid and the layer is unstable. Flows will transfer the heavy fluid to the lower half and the light fluid to the upper half and the density layer will overturn. If $\Delta\rho = 0$ and hence, B = 0, this is the classic Rayleigh-Bénard problem for thermal convection. The governing parameter is the Rayleigh number Ra. If $0 < \mathrm{Ra} < \mathrm{Ra}_c$, the critical Rayleigh number, no flow will occur; e.g., see Turcotte and Schubert (2014). If $\mathrm{Ra}_c < \mathrm{Ra} < \mathrm{Ra}_t$, where Ra_t is the Rayleigh number beyond which *thermal* turbulence develops, steady cellular flow will occur (Turcotte and Schubert, 2014). If $\mathrm{Ra} > \mathrm{Ra}_t$, the flow becomes unsteady and *thermally* turbulent (Turcotte and Schubert, 2014).

If $\mathrm{Ra} > \mathrm{Ra}_c$ and B is small, the boundary between the density differences will not block the flow driven by thermal convection. Kinematic mixing will occur and the composition will homogenize so that the density is constant. Whole layer convection will occur. If B is large, the density difference boundary will block the flow driven by thermal convection. The compositional boundary will be displaced vertically but will remain intact. Layered convection will occur with the compositional boundary as the boundary between the convecting layers. In this work the Rayleigh number Ra defined in equation (3.13) is

based on the domain thickness d and this is the case for which we will show numerical computations.

3.2 Discretizations

In the following discussion of the numerical methodology, we will only consider the dimensionless equations (3.15)-(3.19) and drop the primes associated with the dimensionless variables. The vector form of the dimensionless equations on the two dimensional rectangular domain $\Omega = [0,3] \times [0,1]$ shown in Figure 3.1 are given by

$$-\nabla^2 \mathbf{u} + \nabla P = (-\mathrm{Ra}\ T + \mathrm{Ra}\ \mathrm{B}\ C)\,\mathbf{g}\ , \tag{3.20}$$

$$\nabla \cdot \mathbf{u} = 0\ , \tag{3.21}$$

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \nabla^2 T \tag{3.22}$$

$$\frac{\partial C}{\partial t} + \mathbf{u} \cdot \nabla C = 0, \tag{3.23}$$

where $\mathbf{u} = (u, v)$ is the velocity and $\mathbf{g} = (0, -1)$ is the unit vector pointing downward.

Note that the composition equation (3.23) is equivalent to

$$\frac{DC}{Dt} = \frac{\partial C}{\partial t} + u\,\frac{\partial C}{\partial x} + v\,\frac{\partial C}{\partial y} = 0\,, \tag{3.24}$$

where

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + u\,\frac{\partial}{\partial x} + v\,\frac{\partial}{\partial y} \tag{3.25}$$

is the *material derivative*. Equation (3.24) implies that the composition C is constant on particle paths in the flow (Chorin and Marsden, 1993). Furthermore, since by (3.21) the velocity $\mathbf{u}$ is divergence free, the composition equation (3.23) can be written in conservation form

$$\frac{\partial C}{\partial t} + \nabla \cdot (\mathbf{u}\,C) = 0\,, \tag{3.26}$$

implying that the composition C is a conserved quantity —it is neither created nor destroyed as it is advected in the flow field.

We assume no-flow and free-slip velocity boundary conditions on all boundaries,

$$\mathbf{u} \cdot \mathbf{n} = 0 \qquad (\text{no-flow})\,, \tag{3.27}$$

$$\frac{\partial \mathbf{u}}{\partial \boldsymbol{\tau}} = 0 \qquad (\text{free slip})\,, \tag{3.28}$$

where $\mathbf{n}$ and $\boldsymbol{\tau}$ are the unit normal and tangential vectors to the boundary respectively. We impose Dirichlet boundary conditions for the temperature on the top and bottom of the computational domain and Neumann boundary conditions (no heat flux) on the sides of the computational domain,

$$T(x, 0, t) = 1\,, \tag{3.29}$$

$$T(x, 1, t) = 0\,, \tag{3.30}$$

$$\partial_x T(0, y, t) = 0\,, \tag{3.31}$$

$$\partial_x T(0, y, t) = 0\,. \tag{3.32}$$

The geometry of the computational domain together with the boundary conditions on the temperature are shown in Figure 3.1. In this work we only consider no-flow boundary conditions (3.27). Therefore, we do not need to specify boundary conditions on the compositional field C, since there can be no flow of the composition through the boundaries.

3.2.1 Decoupling of the Nonlinear System

The incompressible Stokes equations can be considered as a constraint on the temperature and composition at any given time leading to a nonlinear system of equations. To solve this nonlinear system, we apply the Implicit Pressure Explicit Saturation (IMPES) approach, originally developed for computing solutions of equations for modeling problems in porous media flow (Huber and Helmig, 1999; Sheldon et al., 1959), to decouple the incompressible Stokes equations (3.15)–(3.17) from the temperature and compositional equations (3.18)–(3.19). This leads to three discrete systems of linear equations, the Stokes equations, the temperature equation, and the composition equation, thereby allowing each equation to be solved easily and efficiently.

3.2.2 Discretization of the Stokes Equations

Let t^k denote the discretized time at the kth time step with a time step size of $\Delta t^k = t^k - t^{k-1}$, $k = 0, 1, \ldots$ Given the temperature T^k and composition C^k at time $t = t^k$, we first solve

the Stokes equations (3.15)–(3.17) to obtain the velocity $\mathbf{u}^k = (u^k, v^k)$ and pressure P^k

$$-\nabla^2 \mathbf{u}^k + \nabla P^k = \left(-\mathrm{Ra}\, T^k + \mathrm{Ra}\, \mathrm{B}\, C^k\right)\ \mathbf{g}\ , \tag{3.33}$$

$$\nabla \cdot \mathbf{u}^k = 0\ . \tag{3.34}$$

For the incompressible Stokes equations (3.33)–(3.34), we use the standard mixed FEM method with a Taylor-Hood element (Donea and Huerta, 2005) for the spatial approximation. We refer the interested reader to Kronbichler et al. (2012) for a more detailed discussion of the spatial discretization and the choice of Stokes preconditioners and solvers.

3.2.3 The Discretization of the Temperature Equation

In mantle convection the thermal diffusivity κ is very small compared to the magnitude of the velocity. Thus, if we let $\|\mathbf{u}\|_{\infty,e}$ denote the maximum magnitude of the velocity on cell e and h_e denote the characteristic size of this cell, in some computations, even for very fine meshes (i.e., small h_e), the local Péclet number on cell e,

$$\mathrm{Pe}_e \ \stackrel{\mathrm{def}}{=}\ \frac{h_e\, \|\mathbf{u}\|_{\infty,e}}{\kappa}\ , \tag{3.35}$$

is usually in the range 10^2 to 10^4. For such high local Péclet number problems, standard finite element discretizations introduce spurious oscillations in the vicinity of steep gradients of advected quantities, even in the presence of some (relatively small) diffusion (Donea and Huerta, 2005). Therefore, some form of stabilization must be added to the discrete formulation of the advection-diffusion equation for the temperature.

In all of the computations presented here we use the algorithm currently implemented in ASPECT to approximate the spatial and temporal terms *in the temperature equation* (3.22) *only*. This algorithm is based on the so-called 'entropy viscosity' method, which is described in detail in Guermond et al. (2011); Kronbichler et al. (2012). The entropy-viscosity stabilization method adds additional (i.e., artificial) 'viscosity' where the local Péclet number is large and the solution is not smooth. In other words, we approximate solutions of the modified temperature equation

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T\ =\ \nabla \cdot (\kappa\ +\ \nu_h(T))\, \nabla T\ , \tag{3.36}$$

with an artificial diffusion term $\nu_h(T)$ added to the equation. Here the entropy viscosity function $\nu_h(T)$ is a non-negative constant within each cell, which can vary from cell to cell.

Note that we have written equation (3.36) in the dimensional form (3.10) in order to make our discussion of the entropy-viscosity stabilization technique consistent with the discussion in Kronbichler et al. (2012).[1] Also note that in all of the computational results shown in this dissertation the thermal diffusivity κ is constant, but that in general this need not be the case. Hence, we have written the advection diffusion equation for the temperature in (3.36) in a more general form.

Conceptually, in regions where the temperature field T is smooth ν_h should be small, and in regions with significant variability ν_h should be of a size that is roughly the same as the diffusive flux in a first-order upwind method. This nonlinear definition of the artificial viscosity ensures that the dissipation is as small as possible, while still large enough to prevent oscillations in the temperature field. In particular, the global approximation property of the method will not be affected, as would be the case with the addition of a simple linear artificial diffusion with a constant value ν_h.

The details concerning how ν_h is determined on cell e, which is denoted $\nu_h|_e$, are as follows. As in Guermond et al. (2011) and Kronbichler et al. (2012) we let

$$\nu_h|_e \;=\; \min\left(\nu_h^{\max}|_e,\; \nu_h^{E}|_e\right) \tag{3.37}$$

In equation (3.37) the maximum viscosity $\nu_h^{\max}|_e$ is defined by

$$\nu_h^{\max}|_e \;=\; \beta\; h_e\, \|\mathbf{u}\|_{\infty,e}\,, \tag{3.38}$$

where the parameter $\beta = 0.078$ is the (current) default value in ASPECT. This parameter controls the maximum dissipation of the entropy viscosity, which is the part that only scales with the cell diameter h_e and the maximum velocity $\|\mathbf{u}\|_{\infty,e}$ in cell e, but does not depend on the solution field itself or its residual.

[1] There are only three differences in our notation from that in Kronbichler et al. (2012). First, in Kronbichler et al. (2012) the authors use the letter 'K' to denote a specific cell rather than the letter 'e' as we do here. In addition, we use c_R and β instead of α_E and $\alpha_{\max}$, respectively, which are used in Kronbichler et al. (2012) but are now also denoted c_R and β, in the ASPECT manual (Bangerth et al., 2019), which is the definitive source for information concerning these parameters.

The entropy viscosity $\nu_h^E|_e$ in equation (3.37) is defined by

$$\nu_h^E|_e \;=\; c_R \,\frac{h_e^2 \,\|r_E(T)\|_{\infty,e}}{\|E(T) - E_{\text{avg}}\|_{\infty,\Omega}}\,, \tag{3.39}$$

where $c_R = 0.33$ is the (current) default value in ASPECT. This parameter controls the part of the entropy viscosity that depends on the solution field itself and its residual in addition to the cell diameter and the maximum velocity in the cell. See the ASPECT manual (Bangerth et al., 2019) for additional information.

Now, if we let

$$T_m \;=\; \frac{1}{2}\,(T_{\min} + T_{\max})\;,$$

then the function $E(T)$ in (3.39) is defined by

$$E(T) \;=\; \frac{1}{2}\,(T - T_m)^2\,.$$

The entropy viscosity in (3.39) is scaled globally by the term

$$\|E(T) - E_{\text{avg}}\|_{\infty,\Omega}\;,$$

which is the maximum deviation of $E(T)$ from its spacial average,

$$E_{\text{avg}} \;=\; \frac{1}{|\Omega|}\int_\Omega E(T)\,.$$

Also, the residual $r_E(T)$ in (3.39) is defined by

$$r_E(T) \;=\; \frac{\partial\, E(T)}{\partial t} + (T - T_m)\,\left(\mathbf{u}\cdot\nabla T \;-\; \kappa\,\nabla^2 T\right)\;.$$

This residual is zero if applied to the true solution T of the temperature equation (3.10), leading to no artificial diffusion. However, it is non-zero when applied to the numerical approximation of the true solution T and will be large in areas where the numerical approximation is poor, such as close to strong gradients.

There is a detailed explanation of how the default values of the parameters β and c_R were chosen in the section entitled "Numerical experiments to determine optimal parameters" of the reference documentation for deal.II (Kronbichler and Bangerth, 2011). The exact value of these parameters may have been been modified since this documentation

was written. However, the manner in which they were chosen is similar to the procedure discussed in the above reference. As of this writing the values $\beta = 0.078$ and $c_R = 0.33$ are the default values in ASPECT. More importantly, these are the values we used in all of the computations of the model problem defined in Section 3.1 that are shown in Section 5.1. These are the only computations in this dissertation that involve approximating solutions of the temperature equation (3.36).

In addition, we note a study on the effect the entropy-viscosity algorithm has on the computed solution as a function of the local Péclet number in a problem that involves a rising square, which is about one-fifth the size of a larger two dimensional square in which the smaller square is hotter than the surrounding fluid done by He et al. (2016). In this work it is demonstrated that on a 100×100 grid the local Péclet number had to be $\mathrm{Pe}_e > 10^2$ in order for the approximate solution of the temperature equation *without* entropy viscosity to cause oscillations in the flow field, while the approximate solution of the temperature equation with entropy viscosity did not allow these oscillations. Furthermore, for computations with $\mathrm{Pe}_e \leq 10^2$ the approximate solution of the temperature equation with and without entropy viscosity were visually identical, suggesting that the entropy viscosity was zero or near zero in most if not all of the cells. In summary, we confirmed that the additional diffusion added by the entropy-viscosity algorithm for approximating solutions of the temperature equation is sufficiently small that it does not adversely affect our computed solutions until the local Péclet number was $\mathrm{Pe}_e > 10^2$.

Now let

$$(\psi, \phi)_\Omega = \int_\Omega \phi(x, y)\, \psi(x, y)\, dx\, dy \tag{3.40}$$

be the inner product of two scalar functions ϕ and ψ on the domain Ω and let $\Gamma_D = \{(x, y) : y = 0\}$ denote the bottom boundary of Ω. Multiplying equation (3.36) by the test function $\psi(x, y)$ and integrating over Ω we obtain the weak form of the spatial discretization

of (3.36)

$$\left(\frac{\partial T}{\partial t},\psi\right)_{\Omega}+(\mathbf{u}\cdot\nabla T,\,\psi)_{\Omega}=-\,(\nabla T,\nabla\psi)_{\Omega} \\ -\,((\kappa\,+\,\nu_h(T))\,\nabla T,\nabla\psi)_{\Omega}+(\frac{\partial T}{\partial\mathbf{n}},\psi)_{\Gamma_D} \tag{3.41}$$

where $\nu_h(T)$ is the entropy viscosity function defined above.

We use the fully implicit adaptive Backward Differentiation Formula of order 2 (BDF2) (Heister et al., 2017; Wanner and Hairer, 1991) to discretize the weak form of the temperature equation with entropy-viscosity in time. Thus, the full discretization of the temperature equation is

$$\frac{1}{\Delta t^{k+1}}\left(\frac{2\Delta t^{k+1}+\Delta t^{k}}{\Delta t^{k+1}+\Delta t^{k}}T^{k+1}-\frac{\Delta t^{k+1}+\Delta t^{k}}{\Delta t^{k}}T^{k} \right. \\ \left. +\frac{(\Delta t^{k+1})^2}{\Delta t^{k}(\Delta t^{k+1}+\Delta t^{k})}T^{k-1},\psi\right)_{\Omega} \\ =-\Big(\mathbf{u}^{k}\cdot\nabla T^{k+1},\psi\Big)_{\Omega}-\Big(\nabla T^{k+1},\nabla\psi\Big)_{\Omega} \\ -\Big(\big(\kappa\,+\,\nu_h^{k}(T)\big)\,\nabla T^{k+1},\,\nabla\psi\Big)_{\Omega}+\left(\frac{\partial T^{k+1}}{\partial\mathbf{n}},\psi\right)_{\Gamma_D}. \tag{3.42}$$

3.2.4 Alternate Discretizations of the Composition Equation in ASPECT

In all of the work described in this dissertation we use the Volume-of-Fluid interface tracking algorithm described in Chapter 2 above to approximate solutions of the composition equation (3.11)[2]. However, there are three other algorithms implemented in ASPECT that one can use to approximate solutions of the advection equation (3.11) for the compositional variable C. In practice, this variable can be density, viscosity, or any other quantity that is passively transported with the flow. Users may have multiple distinct compositions that are each passively advected with the flow. For completeness we briefly describe them here.

[2]This is true albeit with one caveat; namely, in the benchmark problem in Section 4.3 and our computations of processes in the Earths mantle in Chapter 5 we compare the computational results obtained with the VOF method to results obtained with the Bound Preserving Discontinuous Galerkin (DGBP) advection method described in this section.

Two of these three alternate advection methods are based on a spatial discretization of the weak form of the advection equation (3.11) for the composition; namely,

$$\left(\frac{\partial C}{\partial t},\, \psi\right)_{\Omega} \,+\, (\mathbf{u}\cdot\nabla C,\, \psi)_{\Omega} \,=\, 0\,. \tag{3.43}$$

1. The first method that was implemented in ASPECT for approximating solutions of the advection equation (3.11) for the quantity is a *continuous* Galerkin finite element method. Since equation (3.43) is the weak form of the advection equation (3.11) and our numerical approximation to solutions of (3.43) are based on a continuous Galerkin finite element formulation, this advection method also includes an entropy-viscosity stabilization term $\nu_h(C)$ *for the compositional field* on the right-hand side of (3.43),

$$\left(\frac{\partial C}{\partial t}, \psi\right)_{\Omega} \,+\, (\mathbf{u}\cdot\nabla C,\, \psi)_{\Omega} \,=\, -\,(\nu_h(C)\,\nabla C,\, \nabla\psi)_{\Omega}\,. \tag{3.44}$$

   We emphasize that the entropy-viscosity stabilization term $\nu_h(C)$ in equation (3.44) *does not* have the same value in each cell as the entropy-viscosity function $\nu_h(T)$ for the temperature that appears in equation (3.41); they are computed separately and are unlikely to have the same value on any given cell Ω_e. The time discretization of the composition equation in this advection method is also the adaptive BDF2 algorithm. This leads to the following FEM Entropy Viscosity (FEM-EV) discretization of equation (3.11),

$$\begin{aligned}\frac{1}{\Delta t^{k+1}}\Bigg(&\frac{2\,\Delta t^{k+1}+\Delta t^{k}}{\Delta t^{k+1}+\Delta t^{k}}\,C^{k+1} - \frac{\Delta t^{k+1}+\Delta t^{k}}{\Delta t^{k}}\,C^{k} \\ &\qquad + \frac{(\Delta t^{k+1})^2}{\Delta t^{k}(\Delta t^{k+1}+\Delta t^{k})}C^{k-1},\, \psi\Bigg)_{\Omega} \\ &= -(\mathbf{u}^{k}\cdot\nabla C^{k+1}, \psi)_{\Omega} - (\nu_h^{k}(C)\,\nabla C^{k+1},\, \nabla\psi)_{\Omega}\,.\end{aligned} \tag{3.45}$$

   We often refer to this advection method as the FEM-EV advection method, where 'EV' is an abbreviation for entropy-viscosity. Also, we emphasize that this is the only advection method in ASPECT that has any form of artificial viscosity or entropy-viscosity.

2. Another algorithm for modeling solutions of equation (3.11) that we have implemented in ASPECT is a Discontinuous Galerkin (DG) method with a Bound Preserving limiter (DGBP). See He et al. (2017) for a detailed description of this method and a comparison with the continuous Galerkin FEM-EV method described in item 1.

3. We have also implemented a particle-in-cell method in ASPECT (Gassmöller et al., 2018; Puckett et al., 2018), which one can use to approximate the solution of the composition equation (3.23).

See Puckett et al. (2018) for a detailed comparison of these three advection methods with the VOF method described here.

In closing this section we wish to emphasize that in all of the previous and current work designing VOF methods reviewed in this paper no type of artificial viscosity, including entropy viscosity, has been used to stabilize the method. In fact, we are not aware of any version of a VOF algorithm in which some form of stabilization other than the application of an appropriate CFL constraint was required, regardless of whether the VOF method was coupled to a finite element method or to a finite difference method.

Chapter 4

COMPUTATIONAL TESTS AND GEODYNAMIC BENCHMARKS

In this chapter we present our numerical results. First, in Section 4.1 we describe how we measure the error between the approximate and true interface. In Section 4.2 we compute two test problems with prescribed velocity fields to verify the accuracy of our implementation of the VOF algorithm. Next, in Section 4.3 we present a sequence of computations of a time-dependent problem; namely, a falling circular region of greater density than the surrounding fluid and measure the convergence rate of the VOF method in this time-dependent flow field with an interface across which there is a jump in density. Then, in Section 4.4, we compute two well-known benchmarks from the computational mantle convection community to verify that our VOF method has been correctly implemented in the underlying mantle convection code ASPECT. In the next chapter we present the results of two computations that are relevant to problems in the Earth's mantle that are of current interest to the computational Mantle convection community.

4.1 Definition of the Error Measurement

We begin by defining the norm in which we will measure the error between our computed and true solutions (Section 4.2) or estimate the error between computed solutions on successive grids in order to obtain an estimate of the convergence rate of our interface tracking method using a technique that is based on Richardson extrapolation (Section 4.3).

Since each volume fraction f_e is constant on its grid cell Ω_e, we use P_0 elements to store the value of the volume fraction f_e on each Ω_e. Given a *fixed* grid with cells Ω_e indexed by e we define the error between the exact f_e^{exact} and computed f_e^{comp} volume fractions by

$$\text{Error}\left(f^{exact} - f^{comp}\right) = \sum_e \left| f_e^{exact} - f_e^{comp} \right| V(\Omega_e) \tag{4.1}$$

where $V(\Omega_e)$ denotes the volume of the cell Ω_e. Note that (4.1) is the discrete L^1 norm of the difference between f_e^{exact} and f_e^{comp} with weight $V(\Omega_e)$.

4.2 Interface Tracking Benchmark Problems in Stationary Flows

In this section, we compute two test problems in stationary velocity fields with known exact solutions to ensure that our the implementation of the VOF algorithm is exact to machine precision, ϵ_{mach}, when we use it to advect a line in a constant velocity field of the form $\mathbf{u}_{\text{const}} = (u_{\text{const}}, v_{\text{const}})$ and that it converges at its second-order accurate design rate when the flow field is solid body rotation and the interface is a smooth closed curve that does not intersect itself. These very simple problems are what some researchers refer to as 'sanity checks'. In other words, if we do not obtain the expected error / convergence rate, then we know something is wrong with our implementation of the VOF algorithm in the FEM code.

4.2.1 Advection of a Linear Interface in a Constant Velocity Field Benchmark

Our first benchmark is the advection of a linear interface in a constant velocity field $\mathbf{u}_{const} = (\frac{20}{100}, \frac{25}{100})$ as shown in Figures 4.1 and 4.2. In these figures, the computational domain is $[0,1] \times [0,1]$ square covered with a grid of square cells of side $h = 2^{-4}$, and the initial interface is one of the two main diagonals of the domain, namely, $y = 1 - x$. At each time step $t^k \to t^{k+1}$ the interface is advanced with the velocity field $\mathbf{u}_{\text{const}}$, and at $t = 1.0$ the resulting interface is then compared with the exact solution at time $t = 1.0$, $y = \frac{145}{100} - x$. Note that the velocity field *is not* perpendicular to the interface and that neither the interface nor is the flow aligned with the grid.

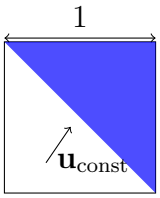


Figure 4.1: A diagram of the initial condition for the "Advection of a Linear Interface in a Constant Velocity Field" benchmark.

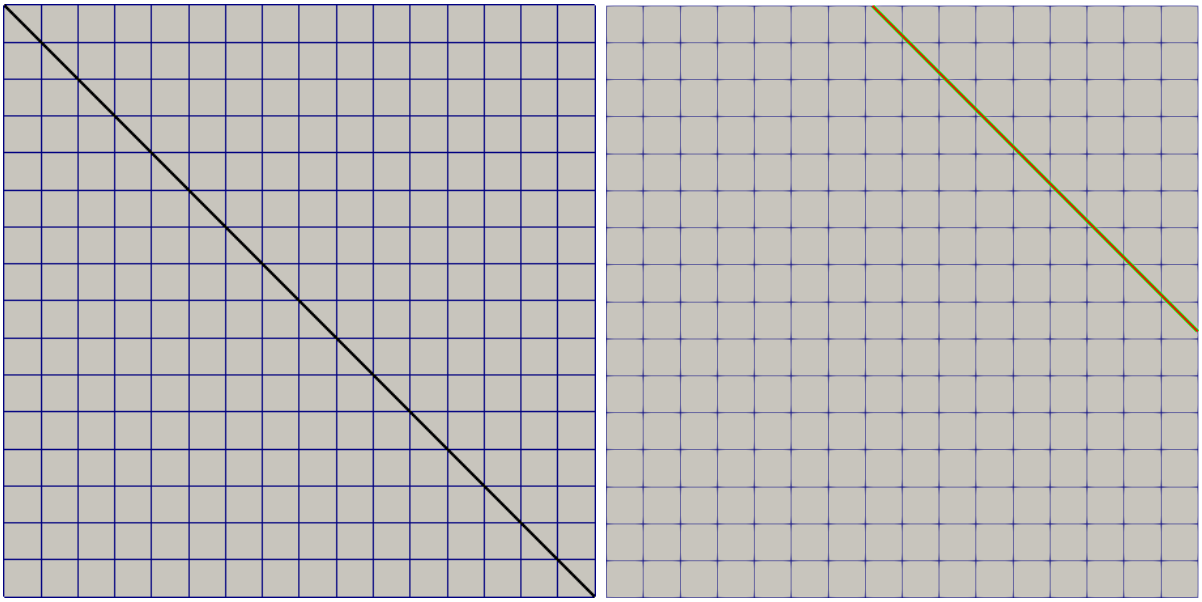


Figure 4.2: On the left is the initial condition as generated by the initialization procedure; namely, a diagonal line reconstructed by the ELVIRA interface reconstruction algorithm. On the right is a comparison between the exact and computed interface at time $t = 1.0$, with the exact interface in red and the computed interface in green. The contour for the (green) computed interface is wider so that the two may be compared visually. It is apparent that the two interfaces are visually indistinguishable. It is also evident from Table 4.1 that the error between the approximate and true interfaces is $O(\epsilon_{\text{mach}})$ where $\epsilon_{\text{mach}} \approx 10^{-16}$ denotes machine precision. This is because the ELVIRA interface reconstruction algorithm will *always* reconstruct a linear interface *exactly* on a grid of equally sized square cells, i.e., up to to machine precision ϵ_{mach} and hence, in a constant velocity field of the form $\mathbf{u}_{\text{const}} = (u_{\text{const}}, v_{\text{const}})$ the approximate interface will remain a line for all time.

In this computation we used a CFL number of $\sigma = \frac{1}{2}$, which resulted in, for example, a total of 23 time steps on the least refined grid of $h = 2^{-4}$. (See equation (2.55) above and the accompanying text for the definition of the CFL number σ and, in particular, its modification for advection problems in ASPECT.) Since the ELVIRA interface reconstruction algorithm reconstructs lines exactly (i.e., to $\epsilon_{\text{mach}} \approx 10^{-16}$), we expect the error

h	Error
2^{-4}	$1.23382 \cdot 10^{-16}$
2^{-5}	$1.21675 \cdot 10^{-16}$
2^{-6}	$2.96083 \cdot 10^{-16}$
2^{-7}	$5.92738 \cdot 10^{-16}$

Table 4.1: The error in advecting a linear interface in a constant velocity field $\mathbf{u}_{\text{const}} = (\frac{20}{100}, \frac{25}{100})$ that is not aligned to the mesh nor perpendicular to the interface. Note that the error is on the order of machine precision $\epsilon_{\text{mach}} \approx 10^{-16}$ and the number of cells that the interface passes through is approximately $\frac{L}{h}$ where L is the distance traveled by the interface from time $t = 0.0$ to time $t = 1.$.

in computations of a linear interface in a constant velocity field to be exact to machine precision $\epsilon_{\text{mach}} \approx 10^{-16}$. The errors from computations with $h = 2^{-4}$, 2^{-5}, 2^{-6}, and 2^{-7} shown in Table 4.1 confirm that this is true for our implementation of our VOF method in ASPECT; namely, in all cases the error is $O(\epsilon_{\text{mach}})$.

4.2.2 The Circular Interface Rotation Benchmark

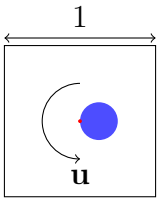


Figure 4.3: Diagram of the Circular Interface Rotation Benchmark problem. Note that the red dot is the center of rotation and the circle is offset from the center of rotation by exactly one radius so that the edge of the circle just touches the center of rotation.

The second benchmark problem is the advection of a circular disk containing composition 1 in a rotating velocity field as shown in Figure 4.3. In this problem the angular velocity is π radians per unit time with an end time of $t = 2.0$. Note that the center of rotation is *not* at the center of the circle, but rather it lies on the boundary of the circle and is marked with a red dot. In each of these computations we used a CFL number of $\sigma = \frac{1}{2}$. The initial and final states for a computation on a grid with $h = 2^{-6}$ are shown in Figure 4.4.

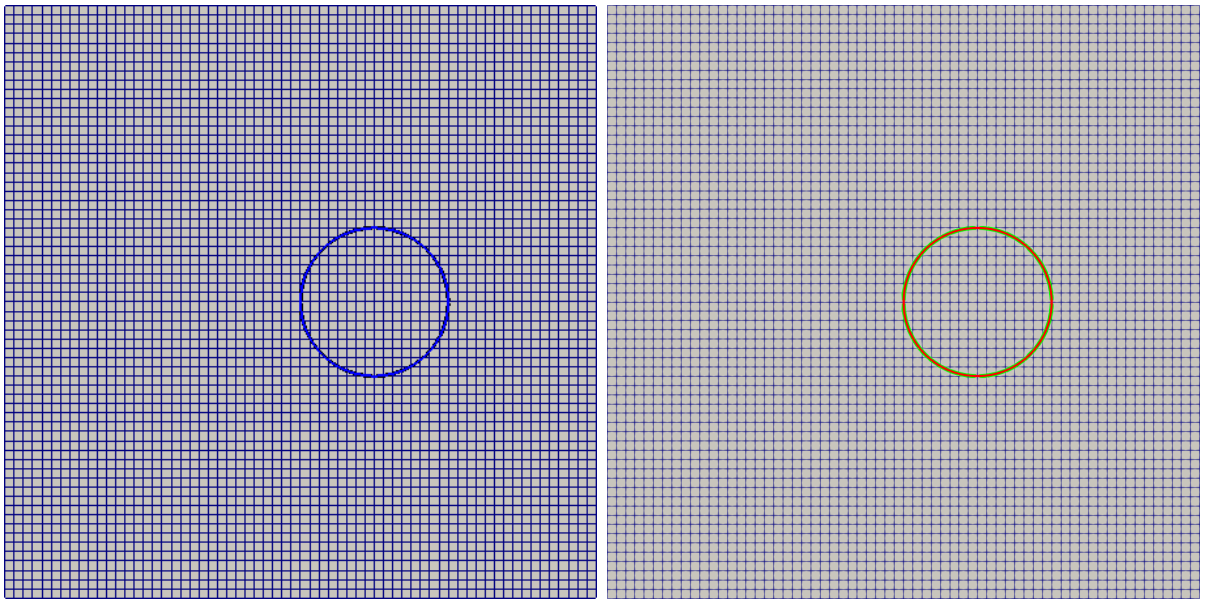


Figure 4.4: The initial and final states for the Circular Interface Rotation Benchmark on a uniform grid with $h = 2^{-6}$. On the left is the initial condition as reconstructed by the ELVIRA interface reconstruction algorithm. On the right is a comparison between the true and computed interface after one full rotation ($t = 2.0$), with the true interface in red and the computed interface in green. The green contour for the computed interface is drawn wider so that the two may be compared visually. It is apparent that the two interfaces are *visually* indistinguishable. It is also apparent from Table 4.2 that the error between the true and computed interfaces is $O\left(h^2\right)$ as $h \to 0$ where h is the length of the side of the square cell shown in the table heading.

Since our interface reconstruction and advection algorithms are designed to be second-order accurate for smooth interfaces in smooth flows, in this problem we expect the approximate interface to be a second-order accurate approximation to the true interface . The (discrete) L^1 error in the volume fractions f_e and the corresponding convergence rates for six computations with increasing grid resolutions of $h = 2^{-4}, 2^{-5}, \ldots, 2^{-9}$ are shown in Table 4.2. It is apparent that the convergence rate asymptotes to 2.00, confirming that the VOF method produces a second-order accurate approximation to the true interface.

4.3 The Sinking Ball Benchmark

We now present a nondimensional variation of the Gerya and Yuen (2003) 'sinking box' problem in order to perform a convergence study on a non-trivial problem. The problem is constructed in non-dimensional form, and therefore none of the presented quantities will

h	Error	Rate
2^{-4}	$6.03897 \cdot 10^{-3}$	
2^{-5}	$1.74516 \cdot 10^{-3}$	1.79
2^{-6}	$3.92745 \cdot 10^{-4}$	2.15
2^{-7}	$1.05605 \cdot 10^{-4}$	1.89
2^{-8}	$2.63464 \cdot 10^{-5}$	2.00
2^{-9}	$6.48952 \cdot 10^{-6}$	2.02

Table 4.2: The error and convergence rate after the true and approximate circular interfaces have rotated $2\,\pi$ radians. It is apparent that the convergence rate tends to 2.00 as $h \to 0$.

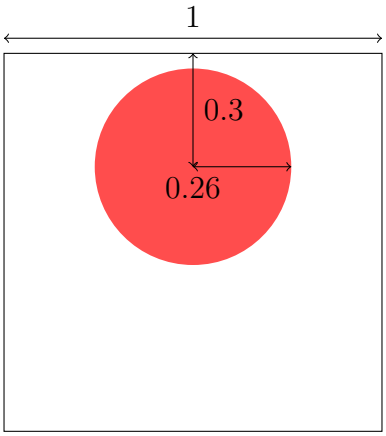
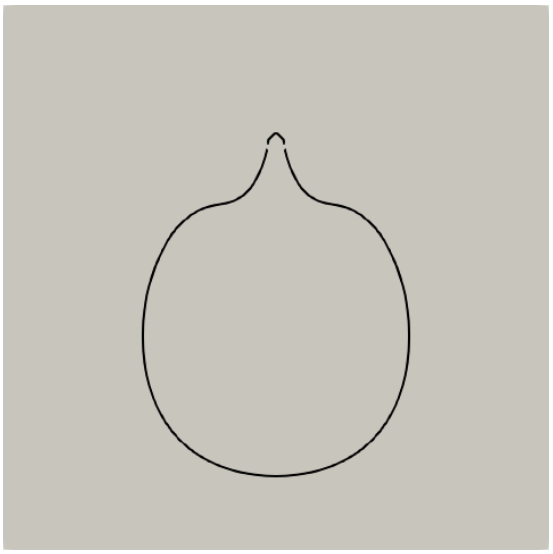
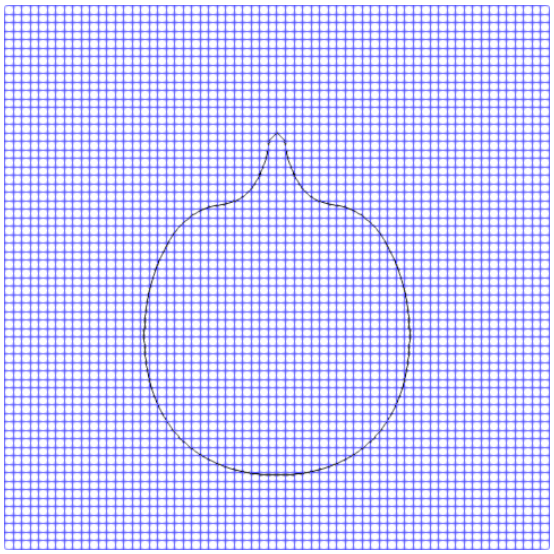


Figure 4.5: Diagram of the initial condition for the Sinking Ball Benchmark problem

include units. Our version of the problem is defined on a 1×1 square domain in which a ball (disk) of heavier fluid of radius 0.26 is horizontally centered 0.3 units below the top edge of the domain as shown in Figure 4.5. The ball's density is $\rho_1 = 110$, while the background density is $\rho_0 = 100$. The viscosity of both the ball and the background fluid is $\mu_0 = \mu_1 = 10^7$. We approximate the solution (of the nondimensional version) of the incompressible Stokes equations in (3.7)–(3.9), but with the term $\rho_0\,\alpha\,(T - T_0)\,g - \Delta\rho\,C\,g$ replaced by $\rho\,g$ in equation (3.9), together with the above initial conditions. Thus, we



(a) The interface plotted against a tan background.



(b) The interface plotted on top of the uniform grid.

Figure 4.6: The interface at time $t = 5 \cdot 10^6$ for the sinking ball test problem on a uniform grid of square cells with sides of length $h = 2^{-6}$.

hold the following parameters fixed:

$$\begin{array}{ll} \mathbf{g} = (0, 9.8) & \text{acceleration due to gravity} \\ L = 1 & \text{domain height and width} \\ \mu_0 = 10^7 & \text{background viscosity} \\ \mu_1 = 10^7 & \text{ball viscosity} \\ \rho_0 = 100 & \text{background density} \\ \rho_1 = 110 & \text{ball density} \end{array} \tag{4.2}$$

For comparison, we also compute the same problem in which we use the 'Bound Preserving Discontinuous Galerkin (DGBP)' method in ASPECT (He et al., 2017), which we described briefly in item (2) of Section 3.2.4, to advect the denser material in the ball. In both cases, the velocity and pressure are discretized by Q_2 and Q_1 elements, respectively. In the DGBP computations, the fluid indicator function $\chi(x, y, t)$ is discretized in space with a discontinuous Q_2 element (often denoted by Q_{-2}) that carries a compositional field

$C \stackrel{\text{def}}{=} C_{DGBP}$. This field is initialized by placing the values of χ at time $t = 0$ given by

$$\chi(x,\, y\,;\, t=0) = \begin{cases} 1 & \text{if} \quad 0 \le (x-0.50)^2 + (y-0.70)^2 \le (0.26)^2\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{4.3}$$

on the support points of the element.

Except for the error in the volume fractions f_e, we estimate the difference between fields w_{2h} and w_h on grids with square cells of side $2h$ and h, respectively, with the following norm,

$$E = \left(\int_{\Omega^h} |w_{2h} - w_h|^p d\,\mathbf{x} \right)^{\frac{1}{p}}, \tag{4.4}$$

where $p = 1$ or $p = 2$ and w_{2h} indicates that w was computed on a grid with square cells of side $2h$, and similarly for w_h. In equation (4.4) w represents quantities such as the pressure and the two velocity components. Since we use a continuous Galerkin finite element method to discretize w in space, the approximation to w is a piecewise continuous function. The integration in (4.4) is performed by quadrature, using points and weights generated by a standard Gauss-Legendre quadrature rule on the more refined cells; i.e., those with side h. Asymptotically, the norms of these differences are proportional to the errors on the coarser grids. This allows us to estimate the convergence rate using a formula such as the one in equation (13) of (Aulisa et al., 2007), despite not having the true solution to the problem. In all of the following convergence studies the final states of the computations were compared at $T_{end} = 5 \cdot 10^6$.

In the case of the volume fractions f_e, it is not appropriate to approximate them as piecewise *continuous* functions, since they are constant on each cell and typically have discontinuities at some of the edges of cells that contain a portion of the interface. Therefore we estimate the difference between the volume fractions f_e^{2h} on a grid with square cells of side $2h$ and f_e^h on a grid of with square cells of side h with the following norm

$$\text{Error}\left(f_e^{2h} - f_e^h\right) = \sum_e \left| \tilde{f}_e^h - f_e^h \right| V(\Omega_e^h)\,. \tag{4.5}$$

where $\tilde{f}_e^h$ are volume fractions on the fine grid Ω^h that are obtained from the reconstructed interfaces on the coarse grid Ω^{2h} via a procedure that is described below and in Figure 4.7.

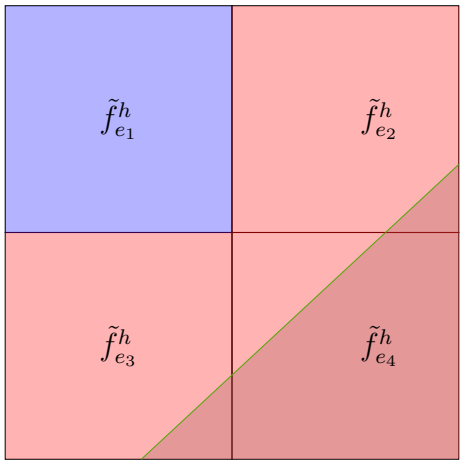


Figure 4.7: An example of how the difference between the volume fractions f_e^{2h} on the coarse cells and the volume fractions f_e^h on the finer cells are computed. The reconstructed interface on the coarse cell, shown in green, produces four volume fractions on each of the four more refined subcells of the coarse cell. These subcells, each with side h, are colored pink and blue in the figure and the volume fractions on the subcells, which are derived from the coarser grid, are denoted $\tilde{f}^h_{e_1}$, $\tilde{f}^h_{e_2}$, $\tilde{f}^h_{e_3}$, $\tilde{f}^h_{e_4}$. We difference these four volume fractions with the four volume fractions $f^h_{e_j}$, $j = 1, 2, 3, 4$ from the finer grid that correspond to the same cells as the subcells of the coarse cell. Note that no linear interface in the coarse cell can pass through all four refined cells. Thus, for example, the compositional field $\tilde{C}^h$ associated with the volume fractions $\tilde{f}^h_1, \ldots, \tilde{f}^h_4$ on the refined cells will have an $O(1)$ jump between the value $0 < C^{2h} < 1$ on the coarse cell and the value $\tilde{C}^h$ on the blue refined cell where, say, in the figure $\tilde{C}^h_{blue} \equiv 0$. The same reasoning applies if $\tilde{C}^h_{blue} \equiv 1$. For this reason the VOF compositional field $C^h \stackrel{\text{def}}{=} C^h_{VOF}$ converges at a first-order rate whereas the volume fractions f^h_e themselves converge at a second-order rate as shown in Table 4.3 and Figure 4.8.

Just as in equation (4.4), the integration in (4.5) is performed on the finer grid Ω^h. Some researchers estimate the accuracy and convergence rate of a VOF interface tracking algorithm by using the difference in the values of the volume fractions on successive grids Ω^{2h} and Ω^h as an estimate of the error in the volume fractions on the coarser grid as we have done here; e.g., Aulisa et al. (2007); Scardovelli and Zaleski (2003). Others have chosen to estimate the error by integrating the difference between the characteristic functions χ^{2h} and χ^h associated with one of the fluids on two successive grids in order to estimate the error in the characteristic functions; e.g., Sussman and Puckett (2000b).

Unlike the functions w that are approximated with a continuous Galerkin element it is necessary to explicitly transfer the volume fraction data f_e^{2h} on the coarse grid to

h	C_{DGBP} L^1 Error	Rate	C_{VOF} L^1 Error	Rate	VOF f_e L^1 Error	Rate
2^{-4}	$5.03 \cdot 10^{-2}$		$3.24 \cdot 10^{-2}$		$1.65 \cdot 10^{-2}$	
2^{-5}	$2.26 \cdot 10^{-2}$	1.16	$1.96 \cdot 10^{-2}$	0.73	$6.93 \cdot 10^{-3}$	1.25
2^{-6}	$1.23 \cdot 10^{-2}$	0.88	$1.03 \cdot 10^{-2}$	0.93	$2.34 \cdot 10^{-3}$	1.57
2^{-7}	$6.22 \cdot 10^{-3}$	0.98	$4.87 \cdot 10^{-3}$	1.08	$5.59 \cdot 10^{-4}$	2.07
2^{-8}	$3.29 \cdot 10^{-3}$	0.92	$2.28 \cdot 10^{-3}$	1.09	$1.25 \cdot 10^{-4}$	2.16
2^{-9}	$1.79 \cdot 10^{-3}$	0.88	$1.14 \cdot 10^{-3}$	1.01	$3.10 \cdot 10^{-5}$	2.02
2^{-10}	$9.88 \cdot 10^{-4}$	0.86	$5.82 \cdot 10^{-4}$	0.96	$9.02 \cdot 10^{-6}$	1.78

Table 4.3: The columns labeled C_{DGBP} and C_{VOF} contain the errors computed for the DGBP and VOF compositional fields, respectively. The column labeled VOF f_e contains the errors in the volume fraction data f_e that we use to reconstruct the fluid interface.

h	DGBP Velocity L^2 Error	Rate	VOF Velocity L^2 error	Rate
2^{-4}	$4.38 \cdot 10^{-9}$		$1.74 \cdot 10^{-9}$	
2^{-5}	$1.23 \cdot 10^{-9}$	1.83	$4.76 \cdot 10^{-10}$	1.87
2^{-6}	$3.64 \cdot 10^{-10}$	1.76	$1.41 \cdot 10^{-10}$	1.76
2^{-7}	$1.09 \cdot 10^{-10}$	1.74	$3.47 \cdot 10^{-11}$	2.02
2^{-8}	$3.78 \cdot 10^{-11}$	1.53	$8.63 \cdot 10^{-12}$	2.01
2^{-9}	$2.44 \cdot 10^{-11}$	0.64	$2.16 \cdot 10^{-12}$	2.00
2^{-10}	$4.69 \cdot 10^{-12}$	2.38	$5.49 \cdot 10^{-13}$	1.98

Table 4.4: Errors and convergence rates in the discrete L^2 norm for the velocity. We note that for this problem, the RMS velocity is typically on the scale of 4×10^{-8}. Note that the second order convergence rates for the velocity are what one expects for the $Q_2 \times Q_1$ element combination we have used to approximate the solution of the underlying incompressible Stokes flow. Note also that an error of $O\left(10^{-12}\right)$ is roughly the smallest error we expect to be able to compute accurately given the tolerance set for the iterative solver of the Stokes matrix equation, in part due to the magnitude of the pressure.

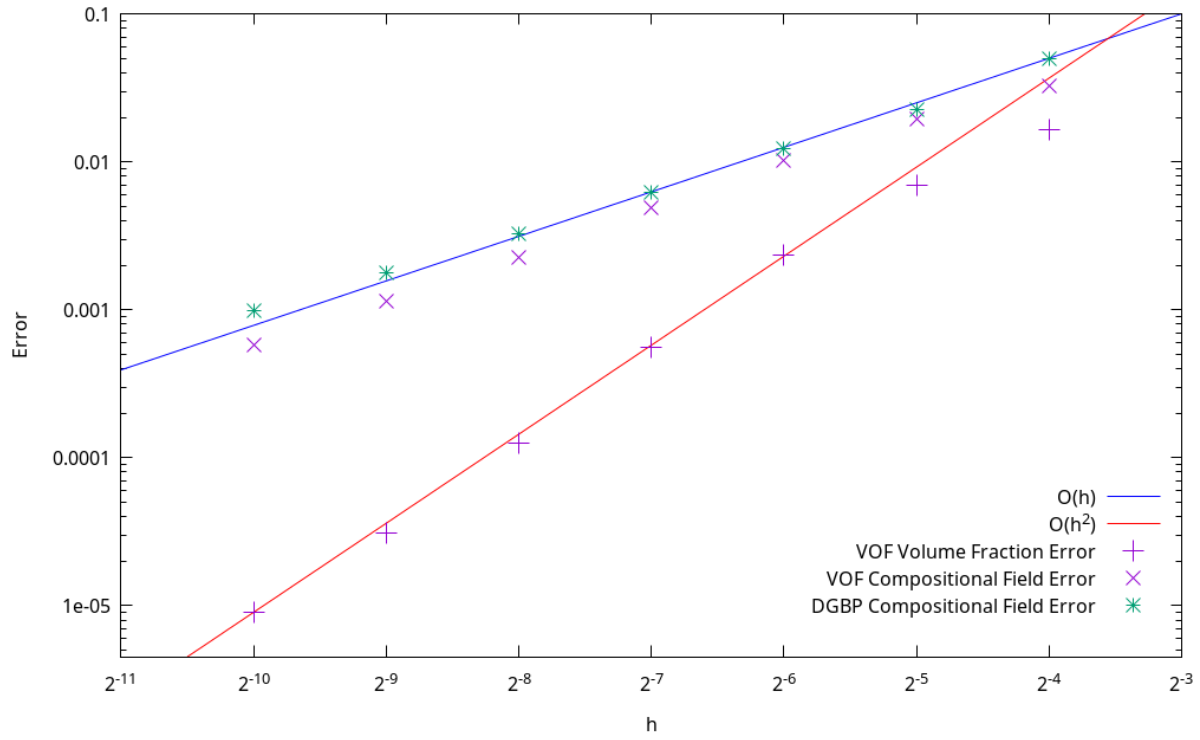


Figure 4.8: Convergence rates in the discrete L^1 norm for the volume fractions f_e and the compositional fields associated with the VOF and DGBP computations of the sinking ball test problem.

corresponding values $\tilde{f}^h_{e_j}$ for $j = 1, 2, 3, 4$ on the finer grid, in order compare them with the given values $f^h_{e_j}$ on the finer grid. We do this by computing the volume fractions $\tilde{f}^h_e$ on the more refined cells with side h by using the interface reconstructed from the volume fractions f^{2h}_e on the coarser grid with side $2h$ as shown in Figure 4.7. We use this reconstructed interface to 'interpolate' (or project) one volume fraction on a coarse cell to four volume fractions $\tilde{f}^h_e$ on the more refined cells that are contained in that coarse cell. We then treat the volume fractions as constant fields on the refined cells and the error is estimated as in equation (4.4).

Due to the nature of the error estimation algorithm, we expect a maximum rate of first-order for the compositional field approximation to the reconstructed interface C derived from the VOF data f_e due to being treated as a standard FEM field during the refinement for comparison. This can be seen as follows. First, for a given level of refinement,

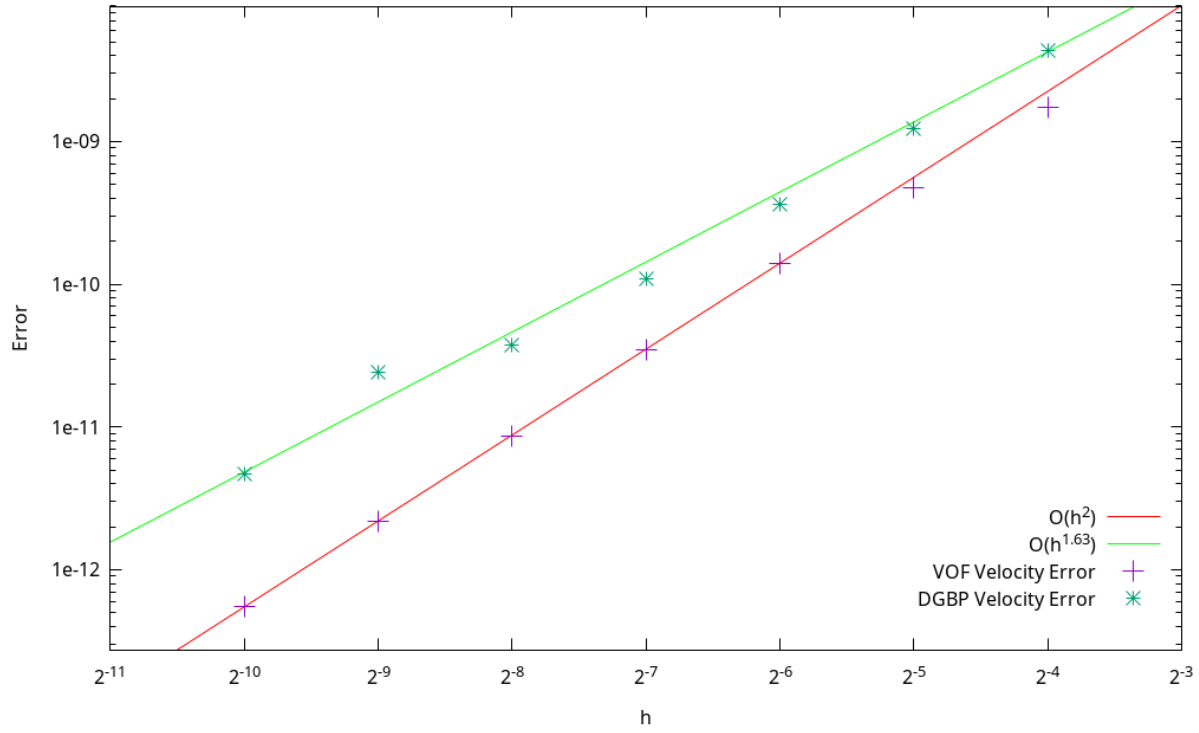


Figure 4.9: Errors and convergence rates in the discrete L^2 norm for the velocity fields from the DGBP and VOF computations of the sinking ball test problem.

say h, we consider the number of cells the fluid interface passes through. Assuming the length of the interface L_I is approximately constant under refinement, we can expect the number of cells the fluid interface passes through to be proportional to $\frac{L_I}{h}$, so the number of cells that contain a fluid interface is $O(h^{-1})$. Upon refinement, it is apparent from Figure 4.7 that each coarse cell will have at least one refined cell that does not contain a fluid interface, and therefore will have a value of either $C = 0$ or $C = 1$ on that cell. Since f_e can be expected to differ significantly from both 0 and 1 in most coarse cells that contain an interface, this results in an $O(1)$ difference in C between the coarse C^{2h} and the fine $\tilde{C}^h$ values for the compositional field on any coarse cell containing the interface, where $\tilde{C}^h$ are the compositional field values due to the volume fractions $\tilde{f}_e^h$ on the subcells that were obtained from the coarse grid volume fractions f_e^{2h} by the procedure described in Figure 4.7. Since the volume of a single refined cell is $O(h^2)$, the maximum convergence rate of the C_{VOF} field should therefore be $O(h^{-1}) \times O(h^2) \times O(1) = O(h)$. This is not

expected to hold for the volume fraction due to the subcell data obtained by the interface reconstruction which is used in the refinement for comparison as described in the above paragraph. This analysis agrees with the computational results shown in Figure 4.8 and Table 4.3.

h	DGBP Pressure L^2 Error	Rate	VOF Pressure L^2 Error	Rate
2^{-4}	$1.30 \cdot 10^{0}$		$4.67 \cdot 10^{-1}$	
2^{-5}	$3.10 \cdot 10^{-1}$	2.06	$1.52 \cdot 10^{-1}$	1.62
2^{-6}	$1.14 \cdot 10^{-1}$	1.44	$5.22 \cdot 10^{-2}$	1.55
2^{-7}	$2.89 \cdot 10^{-2}$	1.98	$1.42 \cdot 10^{-2}$	1.88
2^{-8}	$1.61 \cdot 10^{-2}$	0.85	$4.54 \cdot 10^{-3}$	1.64
2^{-9}	$8.11 \cdot 10^{-3}$	0.99	$1.40 \cdot 10^{-3}$	1.69
2^{-10}	$2.13 \cdot 10^{-3}$	1.93	$4.88 \cdot 10^{-4}$	1.52

Table 4.5: Errors and convergence rates for the pressure in the discrete L^2 norm. We note that for this problem, the average pressure is typically on the scale of 5×10^2. Note that the convergence rates for the pressure are essentially what one expects for the $Q_2 \times Q_1$ element combination that we have used to approximate the solution of the underlying incompressible Stokes flow based on the FEM literature (Heister et al., 2017).

The convergence rates for the volume (Figure 4.9 and Table 4.4) and pressure (Figure 4.10 and Table 4.5) are consistent with the design convergence rate for the VOF algorithm and the selected FEM discretization.

We note that the curvature of the initial interface is $\frac{50}{13}$ and hence, according to equation (2.34), the VOF computation will be underresolved as long as $h > \frac{13}{825} \approx 0.015625 = 2^{-6}$. Furthermore, one can see in Figure 4.8 and Table 4.3 that the volume fractions f_e begin to converge at the full, second-order design rate once $h \leq 2^{-6}$ is satisfied.

4.3.1 A Study of the Effectiveness of AMR on this Problem

The primary benefit of AMR is reduced computation time, which is due to a reduced problem size while still permitting a finer mesh in areas of interest. The precise trade-off

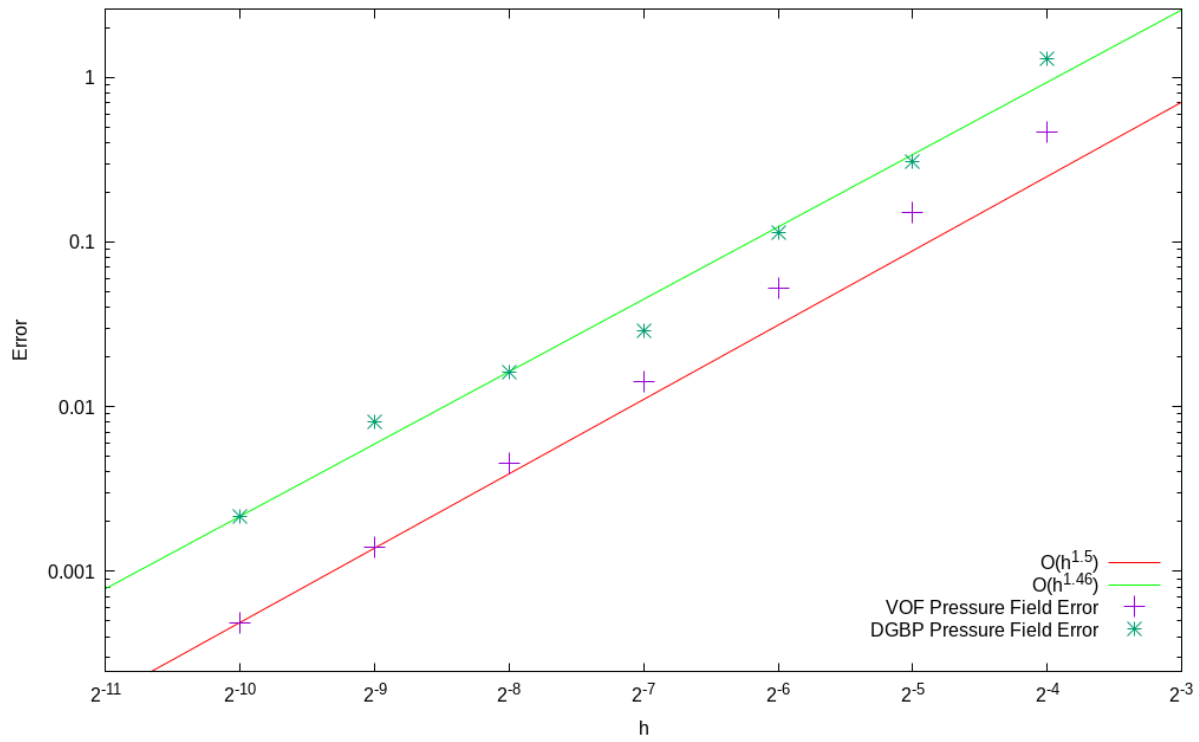


Figure 4.10: Convergence rates in the discrete L^2 norm for the pressure from the DGBP and VOF computations of the sinking ball test problem.

is dependent on the problem under consideration and the strategies used for refining the grid.

This problem, as with many problems involving fluid interfaces, is especially well suited to AMR. The sole feature in the sinking ball problem is an (initially) smooth region with a higher density than the surrounding fluid and the velocity field is largely only affected in a neighborhood of the fluid interface. In this case the AMR strategy we use, which is to only refine the grid to the maximum level of refinement in a neighborhood of the fluid interface, is very well suited to the problem.

In order to examine the performance of the AMR algorithm versus computing on a uniform grid with cell size $h_{min} \times h_{min}$, where h_{min} denotes the size of the most refined cell in the AMR computation, we compare the performance of the two grid strategies when applied them to the sinking ball problem. In all of these computations we use the AMR grid refinement strategy described in Section 2.8. Thus, the fluid interface is the

only feature of importance in this comparison. The AMR algorithm is configured so that the coarsest (square) cell size possible is $h = 2^{-3}$ and we recalculate the mesh at every time step.

In order to examine the efficacy of the AMR computations, the solutions of the uniform and AMR computations are compared using the following value for the error

$$E = \left(\int_{\Omega^h} |w_{uniform} - w_{AMR}|^p \, d\mathbf{x} \right)^{\frac{1}{p}} \tag{4.6}$$

for $p = 1$ or $p = 2$ where, as in equation (4.4), the integration is performed by quadrature, with points and weights generated by a standard Gauss-Legendre quadrature rule on the uniform grid. Since the refinement strategy requires that the fluid interface always be at the maximum level of refinement, it is sufficient to compare the f_e values by treating them as a field that is constant over the corresponding cell.

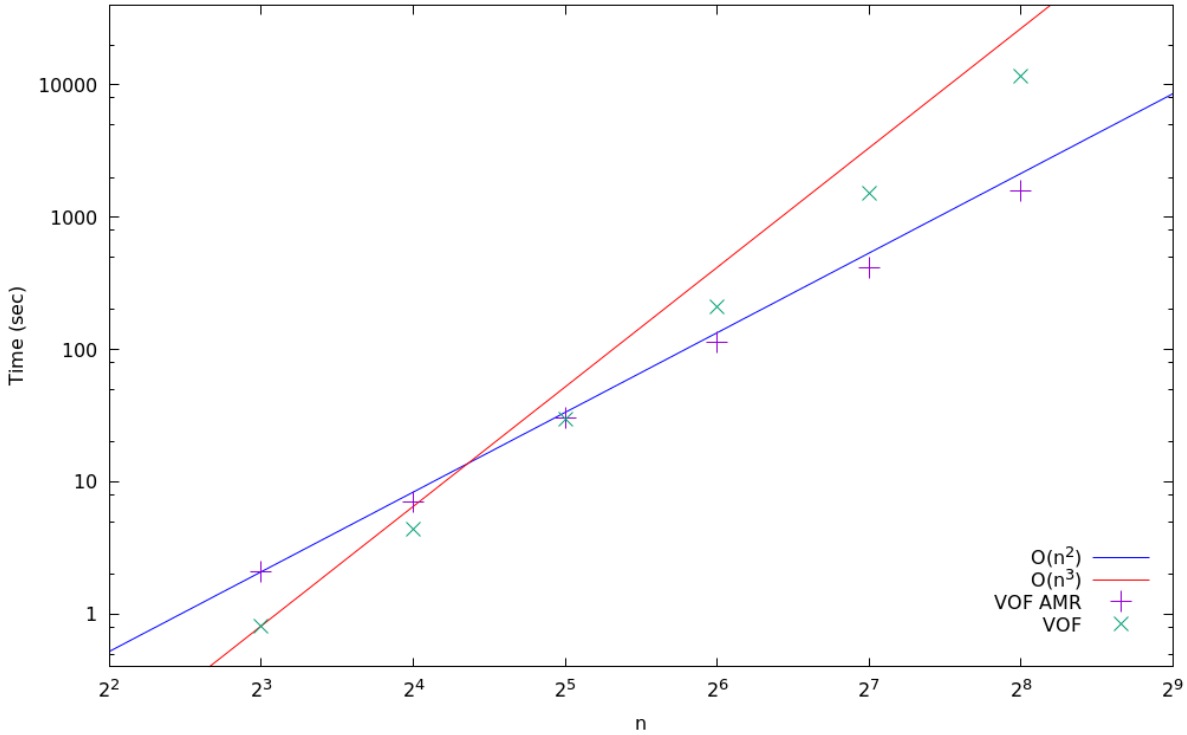


Figure 4.11: Wall clock run times for AMR versus a uniform grid.

In Figure 4.11 and Table 4.6 it is apparent that AMR reduces the computational cost

n	AMR cells	AMR(sec)	Rate	Uniform(sec)	Rate
2^3	$6.40000 \cdot 10^1$	$1.56 \cdot 10^0$		$8.19 \cdot 10^{-1}$	
2^4	$1.84000 \cdot 10^2$	$6.99 \cdot 10^0$	2.16	$4.43 \cdot 10^0$	2.44
2^5	$4.60000 \cdot 10^2$	$3.02 \cdot 10^1$	2.11	$2.96 \cdot 10^1$	2.74
2^6	$1.04800 \cdot 10^3$	$1.14 \cdot 10^2$	1.92	$2.11 \cdot 10^2$	2.83
2^7	$2.26000 \cdot 10^3$	$4.14 \cdot 10^2$	1.86	$1.51 \cdot 10^3$	2.84
2^8	$4.66000 \cdot 10^3$	$1.59 \cdot 10^3$	1.94	$1.16 \cdot 10^4$	2.94

Table 4.6: The number of cells at the final time T_{end} for our AMR computations and the wall clock runtimes with growth rates for both AMR and uniform grids. Here $n = h^{-1}$ is the number of grid cells on a side of the computational domain when the computation is made at the maximum permitted level of refinement.

n	$\|C - C_{AMR}\|_1$	$\|f_e - f_{eAMR}\|_1$	$\|\mathbf{u} - \mathbf{u}_{AMR}\|_2$
2^4	$3.80 \cdot 10^{-6}$	$3.80 \cdot 10^{-6}$	$4.30 \cdot 10^{-11}$
2^5	$1.41 \cdot 10^{-4}$	$1.41 \cdot 10^{-4}$	$4.40 \cdot 10^{-11}$
2^6	$5.41 \cdot 10^{-5}$	$5.26 \cdot 10^{-5}$	$4.10 \cdot 10^{-11}$
2^7	$2.36 \cdot 10^{-5}$	$2.34 \cdot 10^{-5}$	$4.16 \cdot 10^{-11}$
2^8	$1.28 \cdot 10^{-5}$	$1.27 \cdot 10^{-5}$	$4.10 \cdot 10^{-11}$

Table 4.7: The L^1 norm of the difference between the VOF compositional field $C = C_{VOF}$, (i.e., the integral of C_{VOF} integrated over the entire computational domain Ω), the volume fractions f_e, and the L^2 norm of the difference of the velocity field $\mathbf{u}$, each computed on a uniform grid versus an AMR grid at time, T_{end}. As above, we note that the average magnitude of $\mathbf{u}$ for this problem is 4×10^{-8}.

from $O(h^{-3}) = O(n^3)$ to $O(h^{-2}) = O(n^2)$ where $n = h^{-1}$ is the number of grid cells on a side of the computational domain when the computation is made at the maximum permitted level of refinement. This is a significant benefit.

Additionally, in Table 4.7 we show the difference in C_{VOF}, f_e, and the velocity $\mathbf{u}$ for a computation with AMR versus a computation on a uniform grid. We remind the reader that the uniform mesh matches the cell size at the highest level of the AMR computation, and that the chosen refinement strategy focuses only on the compositional interface. It is

apparent from this table that the norm of the differences at the final time for C_{VOF} and f_e are both small and decreasing when $h \leq 2^{-6}$, which, according to (2.34), is the grid size at which the interface is "well-resolved". Note that since $(2^{-8})^2 = 2^{-16} \approx 1.53 \cdot 10^{-5}$, only for the most refined case (i.e., $h = 2^{-8}$) do we have a difference in the volume of the advected fluid that is equivalent in magnitude to the volume of a single cell. For all other refinement levels it is *less than* the volume of a single cell. On the other hand the L^2 norm of the difference in the velocities is $O(10^{-11})$ for all refinement levels, which may be expected due to not being considered as a factor when selecting cells for refinement.

$h = 2^{-k}$	**Uniform Mesh**	**Relative Change**	**AMR**	**Relative Change**
2^{-3}	$-5.81 \cdot 10^{-4}$	$2.73 \cdot 10^{-3}$	$-5.81 \cdot 10^{-4}$	$2.73 \cdot 10^{-3}$
2^{-4}	$-1.59 \cdot 10^{-4}$	$7.48 \cdot 10^{-4}$	$-1.60 \cdot 10^{-4}$	$7.55 \cdot 10^{-4}$
2^{-5}	$-3.94 \cdot 10^{-5}$	$1.85 \cdot 10^{-4}$	$-3.90 \cdot 10^{-5}$	$1.83 \cdot 10^{-4}$
2^{-6}	$-9.77 \cdot 10^{-6}$	$4.60 \cdot 10^{-5}$	$-9.51 \cdot 10^{-6}$	$4.48 \cdot 10^{-5}$
2^{-7}	$-2.45 \cdot 10^{-6}$	$1.15 \cdot 10^{-5}$	$-2.33 \cdot 10^{-6}$	$1.10 \cdot 10^{-5}$
2^{-8}	$-6.11 \cdot 10^{-7}$	$2.88 \cdot 10^{-6}$	$-5.64 \cdot 10^{-7}$	$2.66 \cdot 10^{-6}$
2^{-9}	$-1.53 \cdot 10^{-7}$	$7.20 \cdot 10^{-7}$	$-1.42 \cdot 10^{-7}$	$6.69 \cdot 10^{-7}$
2^{-10}	$-3.80 \cdot 10^{-8}$	$1.79 \cdot 10^{-7}$	$-4.40 \cdot 10^{-8}$	$2.07 \cdot 10^{-7}$

Table 4.8: The total change in the volume of the composition C_{VOF} between the initial and final state when we use the VOF advection method. The correct volume is approximately 0.21238. For the AMR computation h is the maximum level of refinement; i.e., the smallest size a refined cell is allowed to be.

Tables 4.8 and 4.9 demonstrate that when we use AMR instead of a uniform grid it does not have a significant effect on the computed (total) volume of either C_{VOF} (Table 4.8) or C_{DGBP} (Table 4.9). Note that in Table 4.9 the magnitude of the difference between the true value of approximately 0.21238 for the integral of C_{DGBP} over the domain Ω and the computed value is $O(h)$. Also note that in Table 4.9 we have not displayed the results of our computations with $h = 2^{-10}$, which are $O(10^{-10})$, since the difference between the true and computed values of this quantity is $O(10^{-10})$, which is too small to be reliable given that ASPECT only outputs the computed values to an accuracy of $O(10^{-10})$. Finally,

$h = 2^{-k}$	**Uniform Mesh**	**Relative Change**	**AMR**	**Relative Change**
2^{-3}	$-4.21 \cdot 10^{-5}$	$1.98 \cdot 10^{-4}$	$-4.21 \cdot 10^{-5}$	$1.98 \cdot 10^{-4}$
2^{-4}	$-1.64 \cdot 10^{-5}$	$7.70 \cdot 10^{-5}$	$-2.13 \cdot 10^{-5}$	$1.00 \cdot 10^{-4}$
2^{-5}	$-3.18 \cdot 10^{-6}$	$1.50 \cdot 10^{-5}$	$-2.98 \cdot 10^{-6}$	$1.40 \cdot 10^{-5}$
2^{-6}	$-4.23 \cdot 10^{-7}$	$1.99 \cdot 10^{-6}$	$-3.31 \cdot 10^{-7}$	$1.56 \cdot 10^{-6}$
2^{-7}	$-6.10 \cdot 10^{-8}$	$2.87 \cdot 10^{-7}$	$-4.00 \cdot 10^{-8}$	$1.88 \cdot 10^{-7}$
2^{-8}	$-8.00 \cdot 10^{-9}$	$3.77 \cdot 10^{-8}$	$-4.00 \cdot 10^{-9}$	$1.88 \cdot 10^{-8}$
2^{-9}	$-2.00 \cdot 10^{-9}$	$9.42 \cdot 10^{-9}$	$-1.00 \cdot 10^{-9}$	$4.71 \cdot 10^{-9}$

Table 4.9: The total change in the volume between the initial and final state of the composition C_{DGBP} when we used the DGBP advection method. The correct volume is approximately 0.21238. For the AMR computation h is the maximum refinement level; i.e., the smallest size a refined cell is allowed to be. Note that for the DGBP advection algorithm, the difference in the correct volume and the computed volume is approximately $O(h)$. The output from ASPECT of the total value of the composition C over the domain Ω is limited to an accuracy of about $O(10^{-10})$ and hence, for $h = 2^{-10}$ we could not accurately subtract the computed values of these quantities from the true value. Thus, we have omitted the last row of this table since the difference between the computed and true volumes were $O(10^{-10})$.

as we mentioned in Chapter 2 there are operator split VOF advection algorithms that conserve the volume of the fluid to machine precision, which would negate the errors shown in Table 4.8; e.g., see Aulisa et al. (2007); Scardovelli and Zaleski (2003); Weymouth and Yue (2010).

4.4 Mantle Convection Benchmark Problems

In this section we compute two 'benchmark' problems that are well-known and frequently used in the computational mantle convection community to demonstrate that our VOF interface tracking algorithm can reproduce previously published computational results of the same problem. In our view the first problem, commonly known as the "van Keken problem" or the "van Keken isoviscous Rayleigh-Taylor problem" is not a reasonable 'benchmark', since the problem is mathematically ill-posed. In other words, it is unstable (Chandrasekhar, 1961) and perturbations due to different numerical methods can yield vastly differing

results. In fact, in Puckett et al. (2018) we demonstrated that it suffices to change only the algorithm with which the composition variable C is advected in order to obtain clearly different results at the same output time. For example, see Figure 11 of Puckett et al. (2018) or compare Figures 5(c)–(d) of Samuel and Evonuk (2010) to our results here or in Puckett et al. (2018) or to the results in van Keken et al. (1997).

4.4.1 The van Keken Isoviscous Rayleigh-Taylor Problem

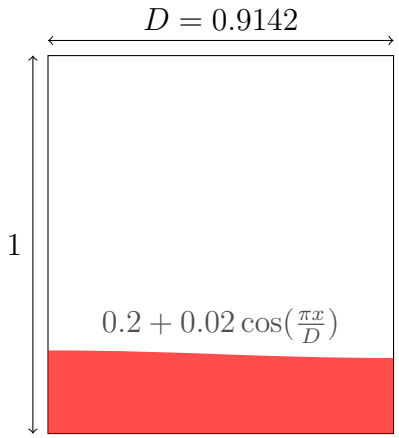


Figure 4.12: A diagram of the initial condition for the "van Keken" test problem (van Keken et al., 1997). Note that although the interface appears to be a straight line it actually has a very small (0.02) amplitude perturbation that is barely visible on this scale. This perturbation produces a Rayleigh-Taylor instability, the results of which can be seen in Figure 4.13. In the formula for the perturbation D = 0.9142 is the width of the domain.

In this section, we present our computation of the van Keken isoviscous Rayleigh-Taylor problem (van Keken et al., 1997). In spite of the fact that the problem is unstable and hence ill-posed, it has become a standard "benchmark" in the computational geodynamics community. In this problem a less dense (buoyant) fluid lies beneath a denser fluid, with a perturbed interface between the two layers. The problem is computed in a $[D,\, 1]$ computational domain where $D = 0.9142$ is the width of the domain. The initial discontinuity between the two compositional / density layers is given by

$$C(x, y, t = 0) = \begin{cases} 0, & \text{if} \quad 0 \leq y < 0.2 + 0.02 \cos(\pi\, x\, /\, D)\ , \\ 1, & \text{otherwise}\,. \end{cases} \tag{4.7}$$

This initial condition has a (discontinuous) interface along the curve

$$y = 0.2 + 0.02 \cos\left(\frac{\pi x}{D}\right). \tag{4.8}$$

The general structure of the solution shown in Figure 4.13 matches the expected form, in particular that observed with other advection algorithms in ASPECT as documented in the manual (Bangerth et al., 2019).

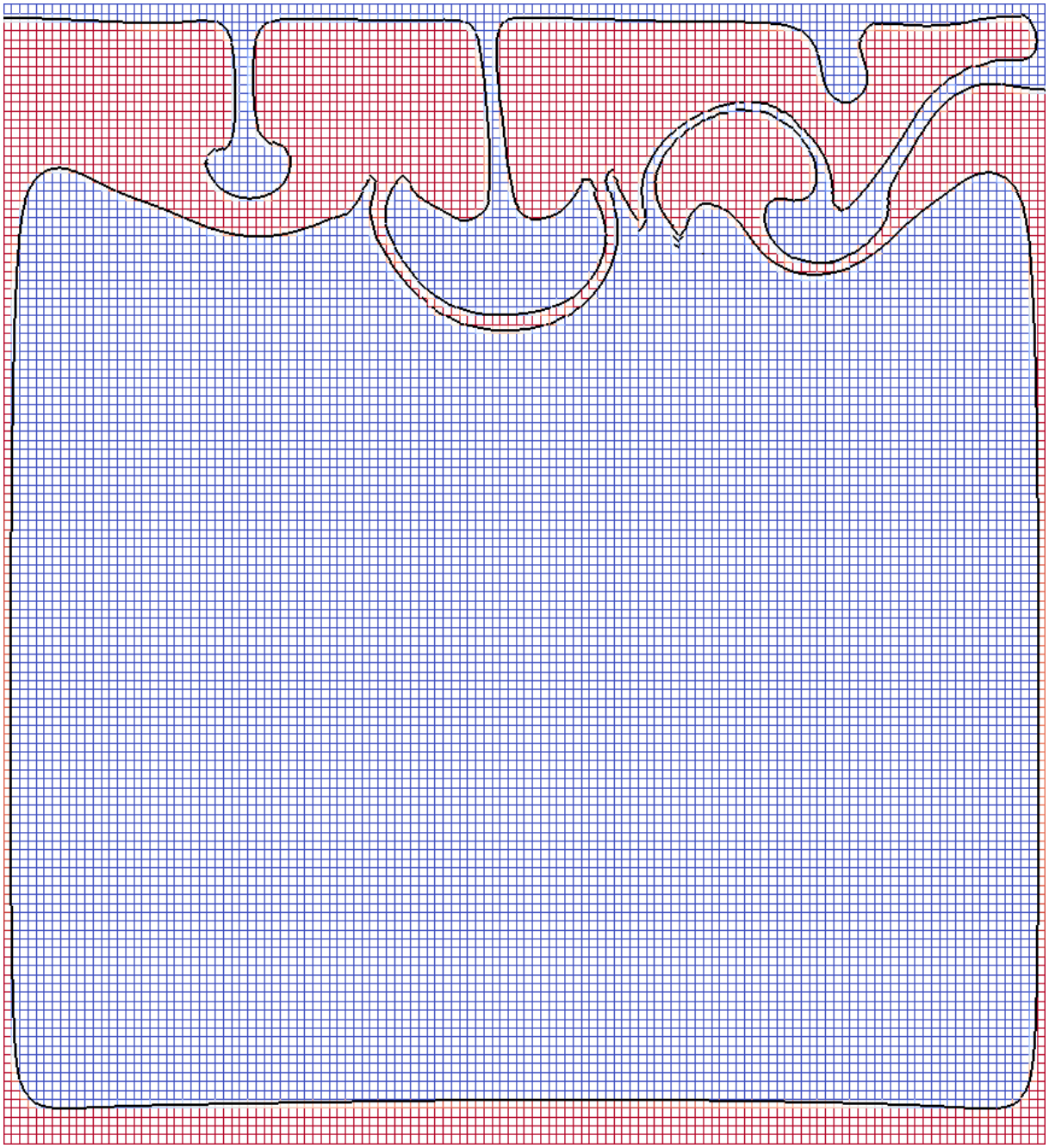


Figure 4.13: Computed solution of the van Keken isoviscous Rayleigh-Taylor problem at time $t = 2000$ on a uniform grid of 128×128 cells. Compare with the computational results in Kronbichler et al. (2012), Puckett et al. (2018), Samuel and Evonuk (2010), and van Keken et al. (1997).

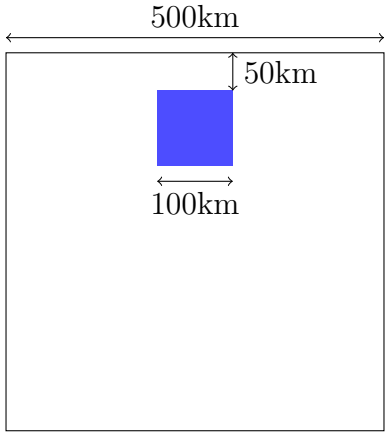


Figure 4.14: Initial conditions for the Gerya-Yuen "Sinking-Box" benchmark.

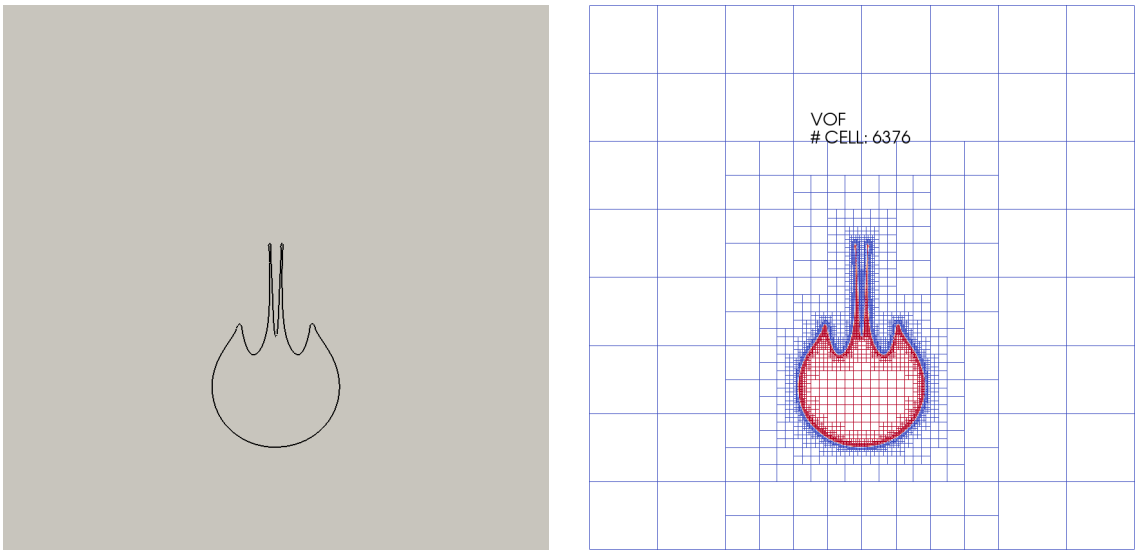
4.4.2 The Gerya-Yuen Sinking Box Benchmark

Following the original authors, we pose the Gerya-Yuen 'sinking box' problem (Gerya and Yuen, 2003) in dimensional form. The problem is defined on a 500 km × 500 km two-dimensional Cartesian computational domain. A small horizontally centered 100 km × 100 km square is placed with its top edge 50 km below the top of the domain so that the the initial location and dimension of the box is defined by the composition field $C(\mathbf{x}, t)$ as follows:

$$C(\mathbf{x}, 0) = \begin{cases} 1, & \text{if } (x, y) \in [200 \text{ km}, 300 \text{ km}] \times [350 \text{ km}, 450 \text{ km}], \\ 0, & \text{otherwise}. \end{cases} \tag{4.9}$$

The block's density is $\rho_1 = 3300$ kg/m^3, while the background density is $\rho_0 = 3200$ kg/m^3. We approximate the solution of the incompressible Stokes equations (i.e.,equations (3.7)–(3.9) without the term $\rho_0 \alpha (T - T_0) g$ in equation (3.9)) with these initial conditions and holding the following parameters fixed:

$$\begin{aligned} &\mathbf{g} = (0, 9.8) \text{ m/s}^2, && \text{acceleration due to gravity} \\ &L = 500 \text{ km} && \text{domain height and width} \\ &\mu = 10^{21} \text{ Pa} \cdot \text{s} && \text{viscosity} \\ &\rho_0 = 3200 \text{ kg/m}^3, && \text{background density} \\ &\rho_1 = 3300 \text{ kg/m}^3, && \text{small box density} \end{aligned} \tag{4.10}$$



(a) The interface against a tan background.

(b) The interface together with the underlying AMR grid.

Figure 4.15: (a) The fluid interface computed by the VOF algorithm for the Gerya-Yuen (Gerya and Yuen, 2003) 'sinking box' benchmark at time $t = 9.81$ myr is a *piecewise linear approximation* to the true interface and (b) the underlying adaptive (AMR) grid on which the interface was computed.

A diagram of the initial conditions is shown in Figure 4.14. The general structure of the final configuration shown in Figure 4.15 matches that documented in the originating paper (Gerya and Yuen, 2003), which suggests that the volume of fluid algorithm is correctly linked to the rest of the ASPECT code.

4.5 Summary of Results

In Section 4.2 we demonstrated that our implementation of the VOF method in ASPECT is exact to machine precision, ϵ_{mach}, when we use it to advect a line in a constant velocity field of the form $\mathbf{u}_{\text{const}} = (u_{\text{const}}, v_{\text{const}})$ and that it converges at its second-order accurate design rate when the flow field is solid body rotation and the interface is a smooth closed curve that does not intersect itself.

Then in Section 4.3 in order to examine the convergence rate of our VOF methodology on a more difficult time-dependent problem we introduced a problem in which a circular region with a higher density than the surrounding fluid falls. We also use this problem

to carefully assess the efficacy of computing with the interface refined using AMR versus computing the same problem on a uniform grid with the (uniform) cell size being the same as the smallest cell size we allowed in the AMR computations. The results of our tests confirm that the algorithm converges at the full design rate. We also confirmed that the AMR strategy yields a significant increase in computational efficiency while remaining close to the uniform mesh result.

In Section 4.4 we demonstrated that the method reproduces (visually) two benchmarks from the computational mantle convection literature.

Chapter 5

COMPUTATIONS OF PROBLEMS IN THE EARTHS MANTLE

In this chapter we present our numerical results when we apply the VOF methodology to two problems of current interest in the field of geodynamics and computational mantle convection. In section 5.1 we present computational results of a model problem first proposed in Puckett et al. (2018), which is designed to provide insight into how material known to originate in structures on the core-mantle boundary known as Large Low Shear wave Velocity Provinces (LLSVPs) reach the Earth's surface via thermal plumes forming hotspot volcanism. In section 5.3, we present computations of a problem intended to test model configurations for computational studies of the behavior of subducting slabs.

5.1 Thermochemical Convection in a Density Stratified Fluid

The work in this section is examined in greater detail in Robey and Puckett (2019).

Recent studies utilizing seismic imaging have revealed large regions with anomalous seismic properties in the lower mantle. In particular, there are two dome-like regions beneath Africa and the Pacific Ocean with low shear-wave velocities that extend some 1000 km above the core-mantle boundary and have horizontal dimensions of several thousand kilometers (Cottaar and Romanowicz, 2012; French and Romanowicz, 2015). Most interpretations propose that the heterogeneities are compositional in nature, differing

from the surrounding mantle, an interpretation that would be consistent with chemical geodynamic models (Kellogg et al., 1999). Based on geological and geochemical studies it has been argued that LLSVPs have persisted for billions of years (Burke et al., 2008).

The model of LLSVPs that we compute here consists of two horizontal layers, equal in height, in a rectangle, with a density difference of $\Delta\rho = \rho - \rho_0 \geq 0$, where ρ_0 is the density of the upper layer. The initial condition for the temperature is a perturbation from the well-known static temperature field (Turcotte and Schubert, 2014) that connects the temperature boundary condition T_0 at the top of the rectangle and to the temperature boundary condition T_1 at the bottom of the rectangle. We study a range of density differences $\Delta\rho$, which we characterize by the non-dimensional buoyancy number B, which is the ratio of $\Delta\rho$ to $\rho_0\,\alpha\,\Delta T$, where $\Delta T = T_1 - T_0$, and α is the volumetric coefficient of thermal expansion. The temperature perturbation initially drives the convection and, depending on the value of B, determines the dynamics and structure of the resulting flow field.

The equations that govern the dynamics of this model problem are those that we presented in Section 3.1. In these computations the Rayleigh number is fixed at $\mathrm{Ra} = 10^5$ and we vary only the buoyancy ratio as follows: $\mathrm{B} = 0.0,\ 0.1,\ 0.2,\ \ldots,\ 1.0$ and $\mathrm{B} = 2.0$. The domain for all of the computational results shown below is a two-dimensional rectangular region that we denote by $\Omega = [0,3] \times [0,1]$ as shown in Figure 3.1.

The initial conditions for the temperature T are,

$$T(\mathbf{x},0) = \begin{cases} (1-5\,y) \;+\; A\,\sin(10\,\pi\,y)\,(1-\cos(\frac{2}{3}\,k\,\pi\,x)) & \text{if} \quad 0 \leq y \leq \frac{1}{10}\,, \\ (5-5\,y) \;+\; A\,\sin(10\,\pi\,y)\,(1-\cos(\frac{2}{3}\,k\,\pi\,x \;+\; \pi)) & \text{if} \quad \frac{9}{10} \leq y \leq 1\,, \\ 0.5 & \text{otherwise}\,, \end{cases} \tag{5.1}$$

where the period of the perturbation $k = 1.5$ and the amplitude of the perturbation $A = 0.05$. Note that $A = 0.05$ ensures that $0 \leq T(x,y;0) \leq 1$ throughout the entire computational domain. The initial conditions for the composition are,

$$C(x,y;t=0) = \begin{cases} 1 \quad \text{if} \quad 0 \leq y < \frac{1}{2}\,, \\ 0 \quad \text{if} \quad \frac{1}{2} \leq y \leq 1 \end{cases} \tag{5.2}$$

and the boundary conditions for the velocity and temperature are as specified in (3.27)–(3.32).[1]

All of the results shown below were computed twice: once on a fixed, uniform grid with 192×64 square cells each with side $h = 64^{-1}$, and then on the same underlying grid but with the addition of two levels of an adaptively refined mesh in a neighborhood of the interface. Each level of refinement increases the grid resolution by a factor of two; i.e., $h \to \frac{h}{4}$ when we use two levels of adaptive mesh refinement.

We compared these computational results with computational results we obtained with the Bound Preserving Discontinuous Galerkin (DG) advection method (He et al., 2017), which we described briefly in Section 3.2.4 above and compared with the VOF method on the sinking ball test problem in Section 4.3.

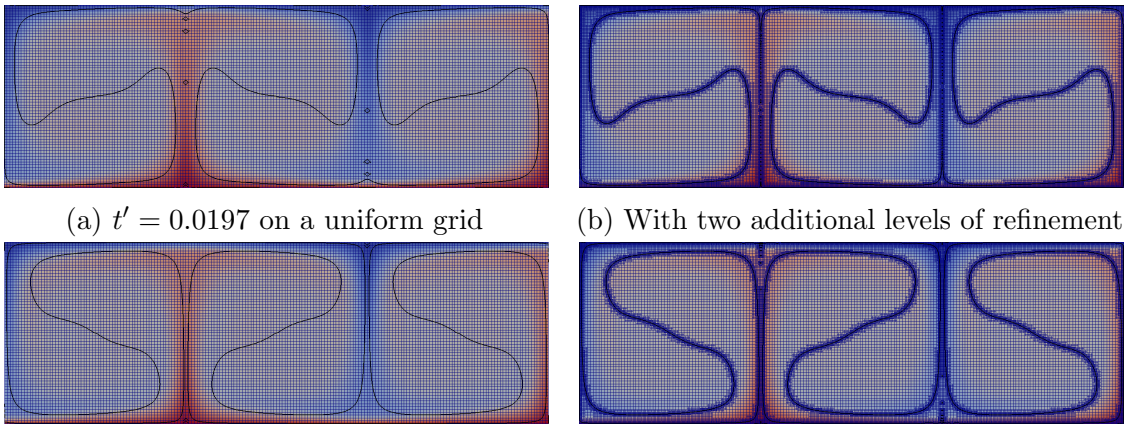
In all of our computations of the model problem defined in Section 3.1 above, the local Péclet number is $\mathrm{Pe}_e < 10$ for the entire time of the computation. Furthermore, in computations of this same model problem, but with different initial conditions as described in the footnote after equation (5.2), with all four of the advection methods that are currently implemented in ASPECT, including this VOF method, at the end time the temperature fields are *visually* indistinguishable for the VOF, DG, and particle advection methods, but not for the advection method with 'Entropy Viscosity' stabilization, which was the first advection method implemented in ASPECT. In this previous work the computational results for the compositional field C were also nearly visually identical, modulo small numerical artifacts associated with the DG and particle methods, for all values of B that did not yield unstable results. See Section 3.2.4 above for a brief description of these three other advection methods in ASPECT.

On the other hand, we have conducted several extensive studies (He et al., 2016; Puckett et al., 2018), which have lead us to conclude that using the algorithm described in

[1] We note that this model problem was also studied in Puckett et al. (2018), partly to compare the quality and efficacy of four different advection methods, including the VOF method presented here, but also to study the computational results of this model problem. However, although the initial conditions for the temperature with $\mathrm{B} = 0.0, 0.1, 0.2$ were the same as those here, for $\mathrm{B} > 0.2$ in the previous paper we eliminated the '$+\pi$' term inside the cosine in the upper boundary layer (i.e., the middle line in equation (5.1)) in order to study the difference in the dynamics of the flow when we used what we refer to as the 'in-phase' temperature initial conditions.

Section 3.2.3 with the entropy-viscosity stabilization technique for approximating solutions of the advection-diffusion equation for the temperature (3.22) does not change these computational results. Nevertheless, we emphasize that is apparent from the studies cited above that the 'pure' advection method with entropy-viscosity stabilization is far too diffusive when used solely as an advection method.

5.1.1 Discussion of the Computational Results



(a) $t' = 0.0197$ on a uniform grid

(b) With two additional levels of refinement

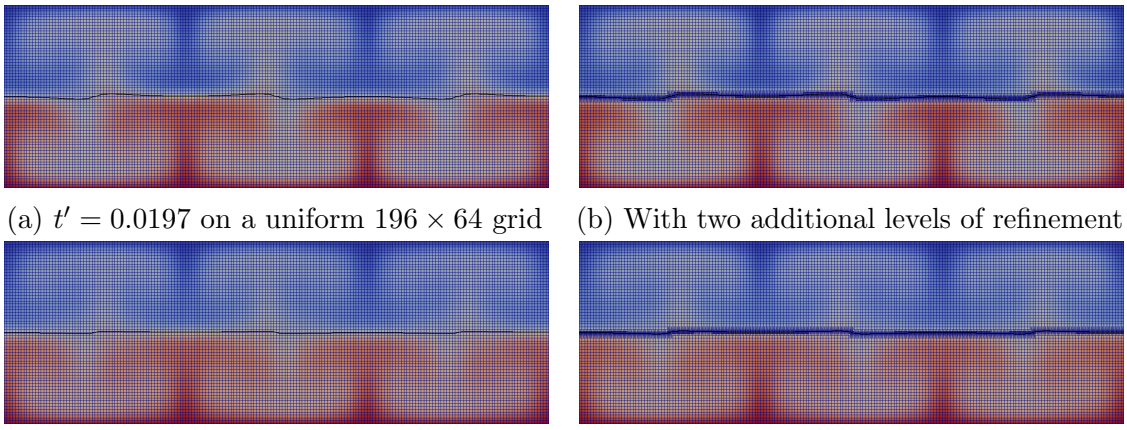
(c) $t' = 0.0236$ on a uniform grid

(d) With two additional levels of refinement

Figure 5.1: Computations with $\mathrm{B} = 0.0$ and $\mathrm{Ra} = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).

Examining the results in Section 5.1 of our computations of thermochemical convection in a density stratified fluid for values of the nondimensional buoyancy parameter $\mathrm{B} = 0.0, 0.1, 0.2, \ldots, 1.0$ and $\mathrm{B} = 2.0$ at Rayleigh number $\mathrm{Ra} = 10^5$, we note a fundamental change in the dynamics and structure of the flow field as B increases from $\mathrm{B} = 0.0$ to $\mathrm{B} = 2.0$. Since we are working with a convection problem, the structure of the velocity field may be deduced by the structure of the temperature field when the flow is steady, or nearly so. First, considering only the extreme values $\mathrm{B} = 0.0$ and $\mathrm{B} = 2.0$, we observe the following difference in the qualitative behavior of the interface.

For $\mathrm{B} = 0.0$ (Figure 5.1), which is the classic Rayleigh-Bénard problem in which there is no difference in the densities of the two fluids (i.e., $\Delta\rho = 0$), the height of the convection



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

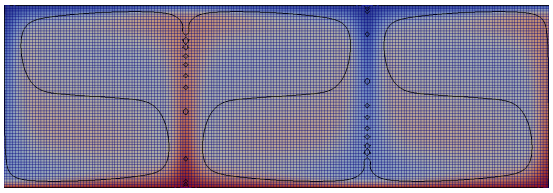
(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

Figure 5.2: Computations with $\mathrm{B} = 2.0$ and $\mathrm{Ra} = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).

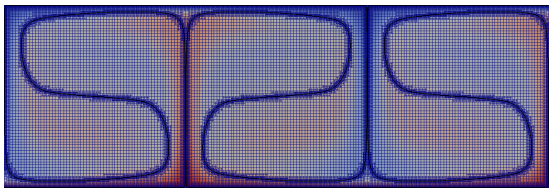
cells is equal to the height of the domain Ω and we observe the steady cellular convection structure with three 1×1 counter rotating cells as predicted by the analysis in Section 6.21 of Turcotte and Schubert (2014). In particular, we note that the convection flow advects the dense fluid all the way to the upper domain boundary, such that the depth of the convection cells include the full depth of the model domain. In Le Bars and Davaille (2005) which we will later used for qualitative comparison, this general flow structure is referred to as "Whole Layer" convection. That the flow in our computational results in Figure 5.1 is nearly steady, (i.e., independent of time) is apparent by noting that the temperature fields (the only factor that should affect the velocity field in this case) at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$ are nearly identical (to visual examination).

On the other hand, for $\mathrm{B} = 2.0$ the magnitude of $\Delta\rho$ prevents the denser fluid from reaching the top of the domain and producing convection cells, on the scale of the height of the domain. Rather, the structure of the flow shown in Figure 5.2 consists of six (roughly) square counter rotating $\frac{1}{2} \times \frac{1}{2}$ cells below $y = 0.5$ and a similar structure above $y = 0.5$. Thus, for $\mathrm{B} = 2.0$ we observe a permanently stratified convection structure, where the fluid boundary forms a barrier for the convection cells. In Le Bars and Davaille (2005) which we will later used for qualitative comparison, this general flow structure is referred

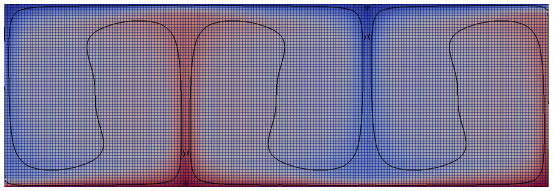
to as "Stratified" convection. Furthermore, from $B = 0.7$ in Figure 5.9 and, *perhaps*, from $B = 0.4$ in Figure 5.6 or $B = 0.5$ in Figure 5.7, on; i.e., as $B \to 2.0$ from below with $B > B_c$ where $0.3 < B_c \leq 0.7$, it appears that *at the times shown* the flow is tending *continuously* toward the stratified convection pattern shown in Figure 5.2.



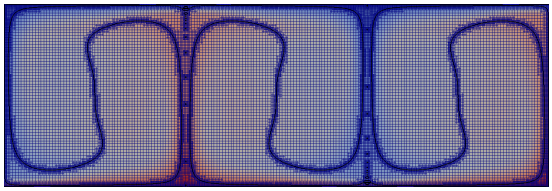
(a) $t' = 0.0197$ on a uniform 196×64 grid



(b) With two additional levels of refinement

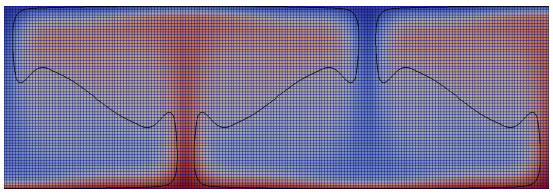


(c) $t' = 0.0236$ on a uniform 196×64 grid

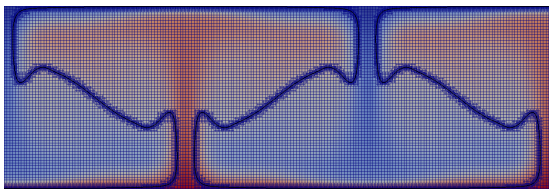


(d) With two additional levels of refinement

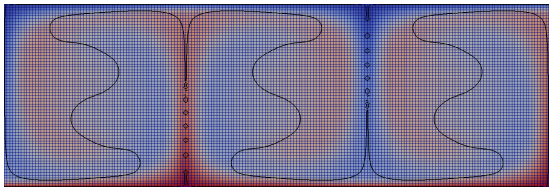
Figure 5.3: Computations with $B = 0.1$ and $Ra = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



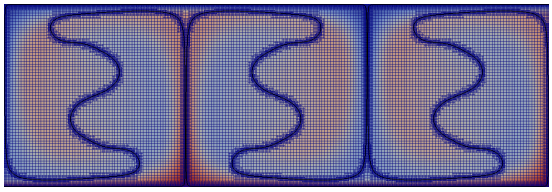
(a) $t' = 0.0197$ on a uniform 196×64 grid



(b) With two additional levels of refinement

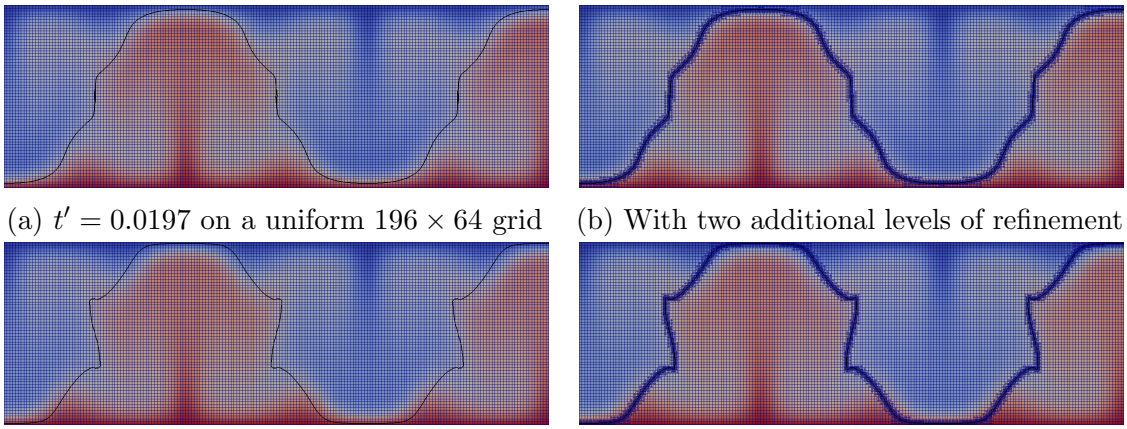


(c) $t' = 0.0236$ on a uniform 196×64 grid



(d) With two additional levels of refinement

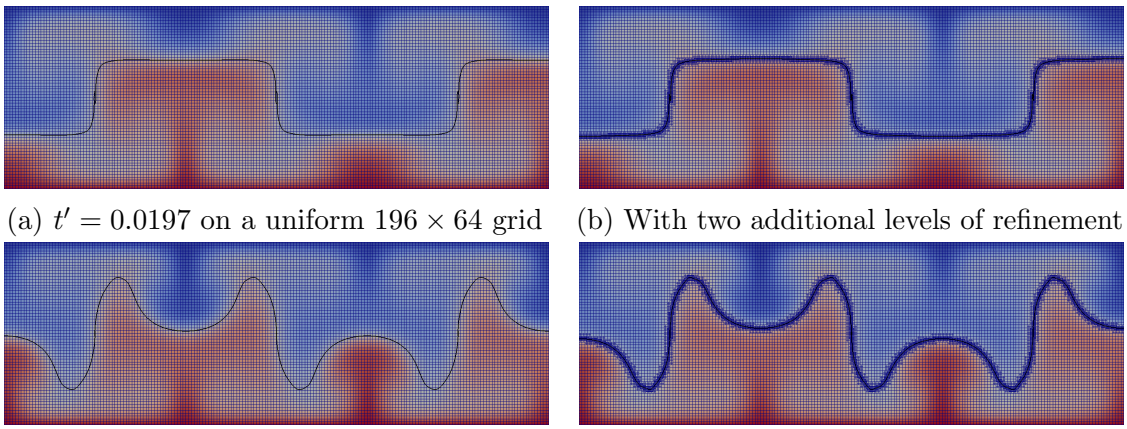
Figure 5.4: Computations with $B = 0.2$ and $Ra = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

Figure 5.5: Computations with $\mathrm{B} = 0.3$ and $\mathrm{Ra} = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

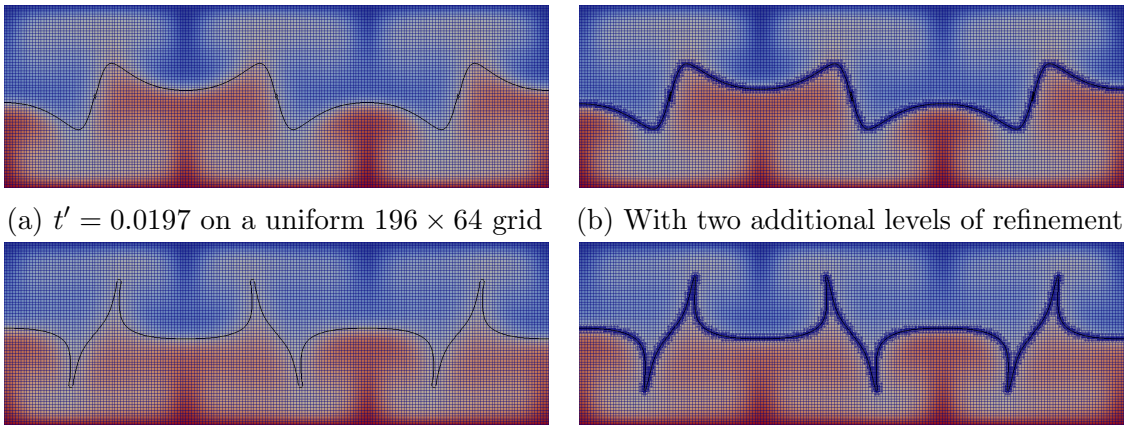
(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

Figure 5.6: Computations with $\mathrm{B} = 0.4$ and $\mathrm{Ra} = 10^5$, (a) and (c), on a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$, and $t' = 2.36 \cdot 10^{-2}$ respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).

Due to the variations in the velocity solution, the thermal Péclet numbers are dependent on the regime in which the solution falls. Because the VOF scheme is inherently non-diffusive, the compositional Péclet numbers are always $\mathrm{Pe} = +\infty$. For the whole layer convection regime the thermal Péclet number is $\mathrm{Pe} = 3.602 \cdot 10^2$, while in the stratified regime it is $\mathrm{Pe} = 1.004 \cdot 10^2$. In both cases, the Péclet number is sufficiently small to justify the use of the entropy viscosity stabilized continuous Galerkin advection-diffusion method for computing the advection and diffusion of the temperature T.

It is possible to obtain additional insight into the structure and dynamics of the flow for various values of B from the results shown in Figures 5.1–5.12. As B increases from 0.0 to 0.1, 0.2, and 0.3 in Figures 5.1, 5.3, 5.4, and 5.5 we observe that the time required for the initial formation of the convection cell increases, until for $\mathrm{B} = 0.3$ the denser material has just reached the top of the domain at $t' = 2.36 \cdot 10^{-2}$ (Figures 5.5a and 5.5b), whereas for smaller values of B the dense fluid has reached the top of the domain and continued to be advected by a full depth convection cell by $t' = 2.36 \cdot 10^{-2}$. For $\mathrm{B} = 0.4$ we can see from Figures 5.6c and 5.6d, that the dense fluid does not reach the uppper boundary and appears to have collapsed backwards by $t' = 2.36 \cdot 10^{-2}$ suggesting that there may be a transition between the qualitative dynamics of the flow at some critical bouyancy number B_c in the range $0.3 \leq \mathrm{B} \leq 0.4$. In Le Bars and Davaille (2005) the authors find $\mathrm{B}_c = 0.302$ when the viscosity ratio is $\gamma = 6.7$.

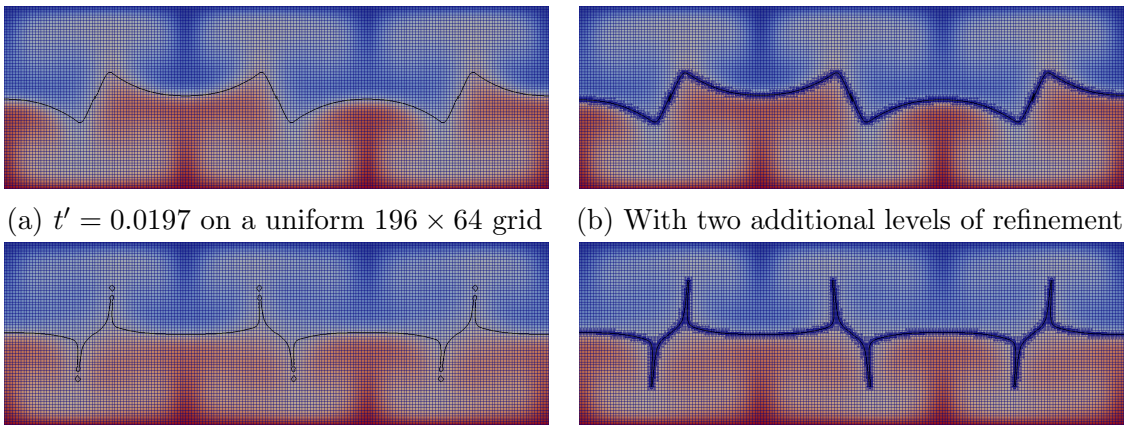
For $0.5 \leq \mathrm{B} \leq 1.0$ in Figures 5.7–5.12 the general interface structures are similar, although with smaller volumes for the "pinched" regions that are produced during the transition to the steady "Stratified" flow. As shown in Figure 5.2, for $\mathrm{B} = 2.0$, the stratification is sufficiently strong that the pinched structures do not form, although a standing wave does form as a slight perturbation from the initial location of the interface at $y = \frac{1}{2}$ with boundaries at $x \approxeq 0.5, 1.5, 2.5$.



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

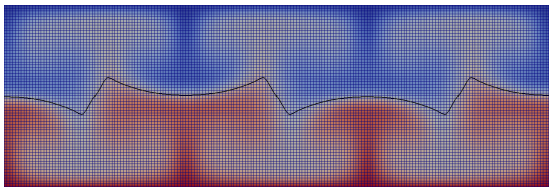
Figure 5.7: Computations with $\mathrm{B} = 0.5$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and (b) and (d) with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



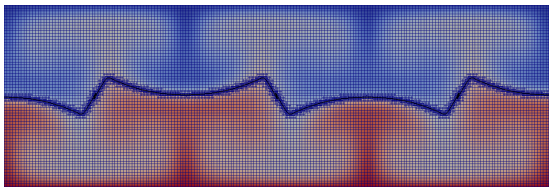
(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

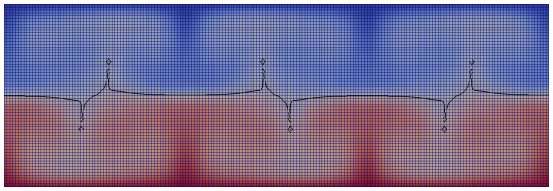
Figure 5.8: Computations with $\mathrm{B} = 0.6$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



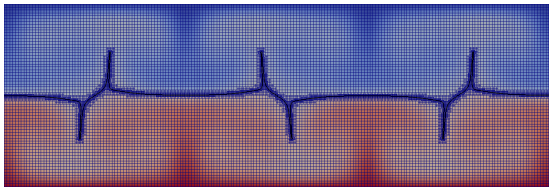
(a) $t' = 0.0197$ on a uniform 196×64 grid



(b) With two additional levels of refinement

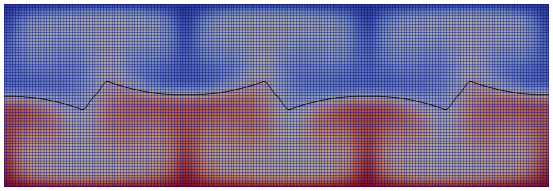


(c) $t' = 0.0236$ on a uniform 196×64 grid

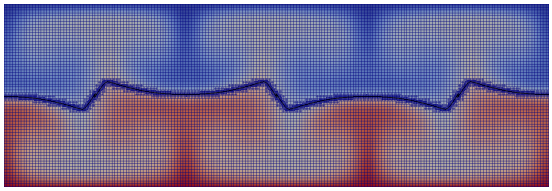


(d) With two additional levels of refinement

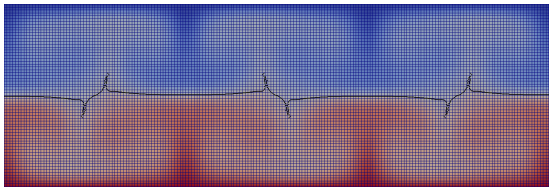
Figure 5.9: Computations with $\mathrm{B} = 0.7$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



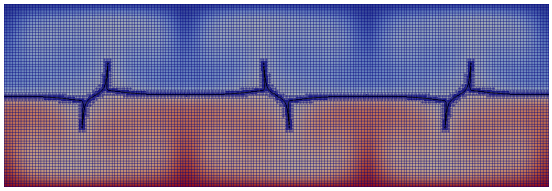
(a) $t' = 0.0197$ on a uniform 196×64 grid



(b) With two additional levels of refinement

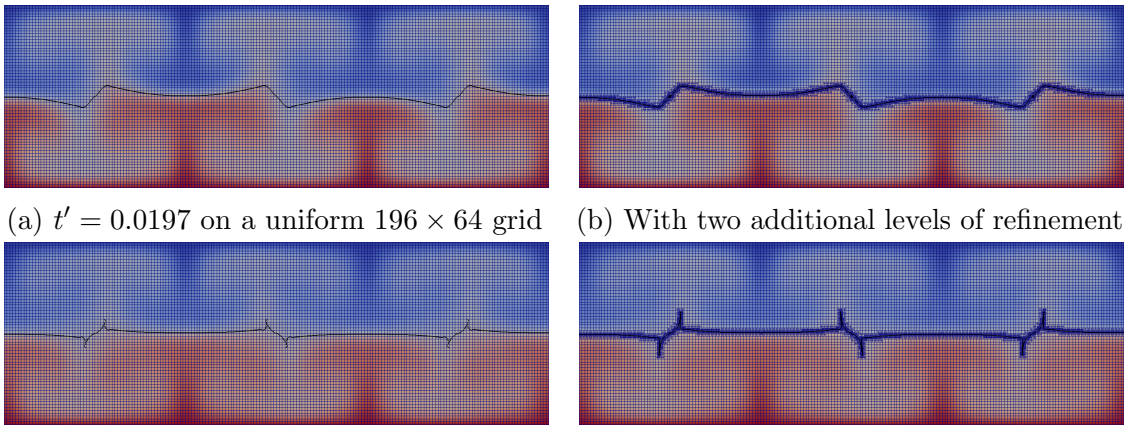


(c) $t' = 0.0236$ on a uniform 196×64 grid



(d) With two additional levels of refinement

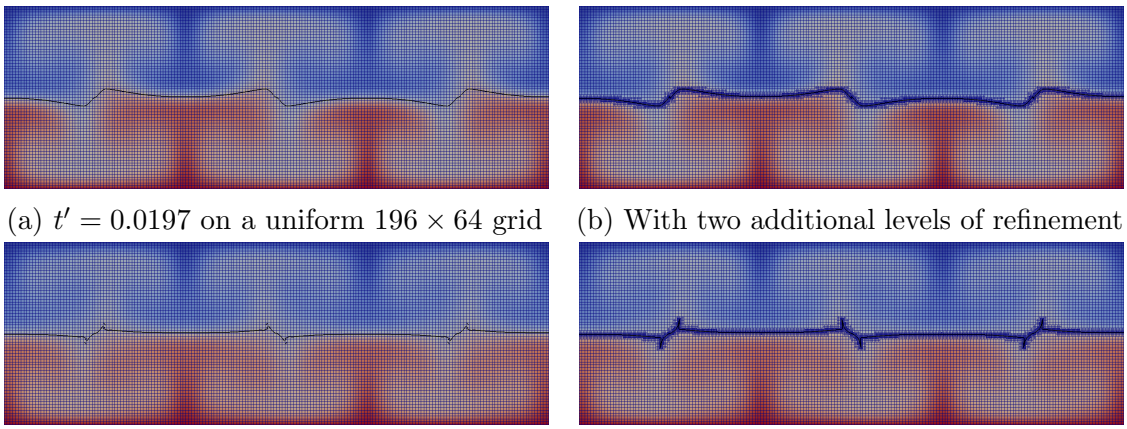
Figure 5.10: Computations with $\mathrm{B} = 0.8$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red). The disordered contours in the uniform computation are an artifact indicative of insufficient resolution.



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

Figure 5.11: Computations with $\mathrm{B} = 0.9$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).



(a) $t' = 0.0197$ on a uniform 196×64 grid (b) With two additional levels of refinement

(c) $t' = 0.0236$ on a uniform 196×64 grid (d) With two additional levels of refinement

Figure 5.12: Computations with $\mathrm{B} = 1.0$ and $\mathrm{Ra} = 10^5$ on, (a) and (c), a uniform grid of 196×64 square cells at $t' = 1.97 \cdot 10^{-2}$ and $t' = 2.36 \cdot 10^{-2}$, respectively, and, (b) and (d), with two additional levels of refinement *only on the interface*. The background color is the temperature, which varies from $T = 0.0$ (dark blue) to $T = 1.0$ (dark red).

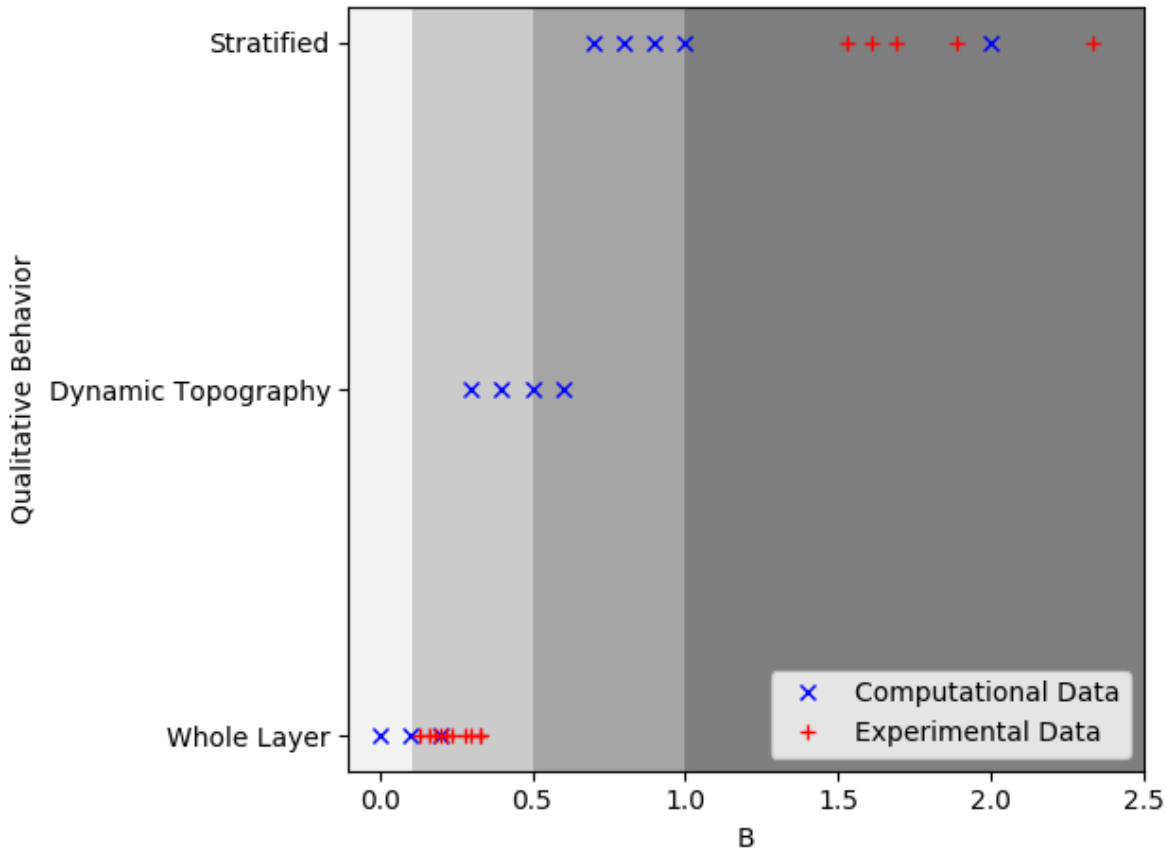


Figure 5.13: A *qualitative* comparison of the computations presented in this paper to the experimental results of Davaille (1999) and Le Bars and Davaille (2004, 2005). The grayscale parameter regions correspond to boundaries of the qualitative regions shown in in Figure 3 of Davaille (1999) and Figure 2 of Le Bars and Davaille (2005) for $a = 0.5$. The experimental data is from Table 3 of Davaille (1999) and Table 3 of Le Bars and Davaille (2004), selecting the cases with $a = 0.5$ as is the case for all of the computations in this dissertation. The terms "Stratified", "Dynamic Topography", and "Whole Layer" used to describe the qualitative state of the flow are the same as those used by the authors of Le Bars and Davaille (2004, 2005) and Davaille (1999).

5.1.2 A Qualitative Comparison to the Experiments Davaille and Le Bars

In this section we briefly make some additional *qualitative* comparisons of our computational results to the experimental results of Davaille Davaille (1999) and Le Bars & Davaille Le Bars and Davaille (2004, 2005).

The features at either end of the interval $\mathrm{B} = [0.0\,, 2.0]$ are consistent with the

results — obtained from experiments — described in Le Bars and Davaille (2005) and shown in Figure 1 of that paper that describe the "Whole Layer" (convection cells are the full depth) and "Stratified" (convection cells are contained within the distinct fluids) qualitative regimes. We therefore find that our computational results correctly correspond *qualitatively* to what the authors of Le Bars and Davaille (2005) observe in their experiments when the nondimensional parameters a, the relative depth of the dense fluid, and γ, the ratio of the fluid viscosities, are held fixed at $a = 0.5$ and $\gamma = 1.0$.

Before making further comparisons, it is first necessary to make several caveats. First, as we mentioned above, in the experiments the authors varied two additional nondimensional parameters; namely, (1) the ratio a of the height of the lower layer to the height of the entire domain and (2) the ratio γ of the viscosity of the fluid that initially occupies the lower layer to the viscosity that initially occupies the upper layer. In our computations, shown in Section 5.1, we kept these parameters fixed at $a = 0.5$ and $\gamma = 1.0$. Second, in the experiments the two fluids are *miscible*, whereas in our computations the two fluids are *immiscible* (i.e. the two types of fluid are always distinct with a well defined boundary between the two). In both cases there is no surface tension at the boundary between the two fluids.

The general transition between one type of structure and another (e.g., "Whole Layer" convection to "Stratified Convection") is similar to that found in the experiments shown in Le Bars and Davaille (2005), although the precise location of the transition may differ. A rough comparison is show in Figure 5.13. The different grayscale backgrounds in Figure 5.13 correspond to the grayscale regions in Figure 3 of Davaille (1999) and Figure 2 of Le Bars and Davaille (2005) for $a = 0.5$. In the results presented here we do not continue the computation for sufficiently long times to confirm that in the $0.3 \leq \mathrm{B} \leq 0.5$ regime the flow oscillates before beginning an overturn. However, the observed behavior does produce structures that match those described in Le Bars and Davaille (2005) for the length of time for which we do have computational results. This difference may be in part due to the fact that in Le Bars and Davaille (2005) the two fluids also vary in viscosity ratio γ, and Rayleigh number Ra.

5.2 Numerical Artifacts That Occur When the Interface Is Underresolved

Since the VOF method maintains a sharp interface between the two compositional fields, it is able to capture features that are approximately on the order of the grid scale h. However, in cases where the structures formed by the interface become sufficiently small, for example, a thin column of fluid of width $2\,h$, the interface reconstruction algorithm might produce numerical artifacts that are "characteristic" of the combination of the particular reconstruction algorithm and advection algorithm one chooses to use in the VOF method.[2] Here we briefly examine of the nature of one particular numerical artifact that appears frequently in Section 5.1.

The most common numerical artifact in the computational results shown in Section 5.1 is the tendency for the reconstructed interface to form 'droplets' that are diamond shaped and generally occupy a square of four cells. For example, droplets such as these appear in Figure 5.3a. In the computations shown in Section 5.1 these droplets typically resolve into a thin vertical column of fluid of approximately $2h - 4h$ in width with a length that is nearly the entire height of the computational domain. For example Figures 5.3a and Figure 5.3b, in which the more refined computation in Figure 5.3b appears to be sufficiently well-resolved to draw the conclusion that a thin column of fluid is forming in the locations where in Figure 5.3a there are only a few droplets and no real indication of what the flow "should" look like. Or the droplets may resolve into a thin finger that is shorter than the height of the computational domain such as in Figures 5.8c 5.8d, 5.9c, and 5.9d.

We note that if a feature of the interface is underresolved, it can help the user determine if additional refinement is required. In some instances, perhaps after making a second, more refined computation, it will be clear that additional refinement is necessary, sometimes even more refined than the second computation was. For example see Figures 5.3c and 5.3d,

[2]It is important to recognize that this is not a failing of the VOF method in general or of the specific interface reconstruction and advection algorithms we have chosen for our work here, since whenever a computation is underresolved, *all* numerical methods will exhibit some sort of numerical artifact or artifacts that are "characteristic" of that particular method.

neither of which appear sufficiently well resolved to accept the computation in Figure 5.3d as well resolved enough to determine the true nature of the flow. On the other hand, there are instances when the numerical artifact is sufficiently small so as not to affect the dynamics of the interface that are of interest and additional resolution might not be required. For example, depending on the user and the underlying scientific application, this might be the case for Figures 5.10c, 5.10d, 5.11c, and 5.11d, even though under magnification the fingers in the refined computations shown in Figures 5.10d and 5.11d do not yet appear fully resolved. In other words, depending on the application, these computations may or may not be well resolved enough for the user to arrive at conclusions *appropriate for their application* concerning the flow at this point in time.

In conclusion, we emphasize that the required degree of resolution for a given computation will depend on the purpose of the computation and the user's need for fine detail as opposed to general qualitative information concerning the flow.

5.3 A Computational Model of a Subducting Slab

One of the important problems in computational geoscience is that of modeling plate tectonics. A significant repeating structural element in this problem is a subduction zone, where one tectonic plate is forced under the other and descends into the mantle (Turcotte and Schubert, 2014). The plate that has been forced into the mantle is referred to as a

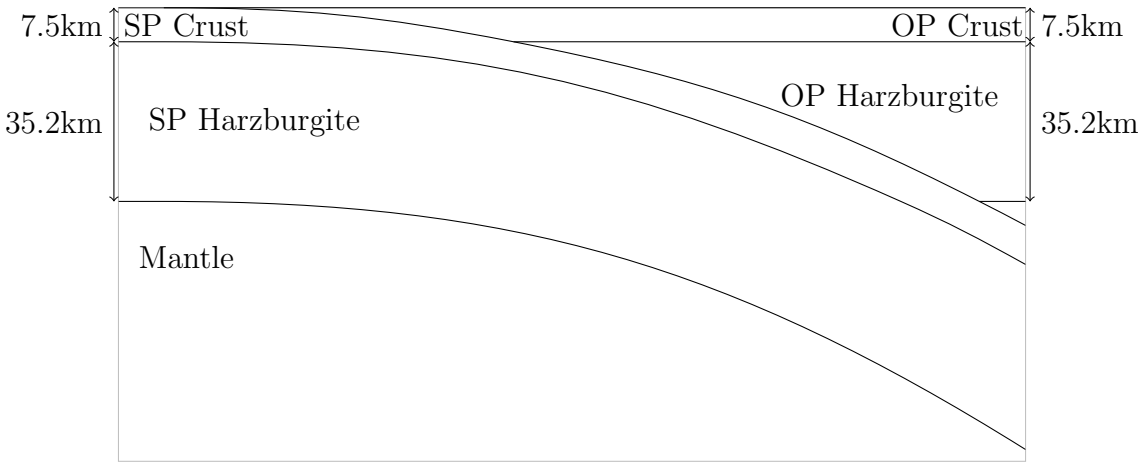


Figure 5.14: Sketch of initial subduction zone configuration

subducting slab. The manner in which the slab sinks or is *subducted* into the mantle has been observed to vary and computational models suggest that this variation depends on a number of factors. Since the behavior of the subducting slab is believed to be a significant driving factor in the motion of tectonic plates, a better understanding of the contributing elements is highly desirable.

Problems involving plate tectonics and subducting slabs are arguably among the most important open problems in geodynamics (Arredondo and Billen, 2016; Billen and Arredondo, 2018; Agrusta et al., 2017). Consequently, accurate numerical computations of these problems are extremely important. The computation we present here is a (somewhat) simplified model of a subducting slab, which was chosen to allow tuning of the various computational parameters in order to compute problems containing a additional features and physics. Some important factors one must consider include the choice of algorithm, the refinement strategy, the size of the mesh necessary to avoid edge effects, the necessary resolution to capture the phenomena of interest, and where efficiency may be gained by adjusting the minimum refinement level.

The initial conditions (i.e., in the ASPECT parameter file) for our computations were provided by Professor Magali Billen of the Earth and Planetary Sciences Department at UCD. These initial conditions correspond to the basic reference model in Billen and Arredondo (2018). This problem includes distinct materials for the crust and harzburgite[3] of the overriding plate (OP) and subducting plate (SP), respectively, sinking or subducting into the Earth's mantle. The mantle lies between Earth's dense, super-heated core and its thin outer layer, the crust. The mantle is about 2,900 kilometers (1,802 miles) thick, and makes up 84% of Earth's total volume (National Geographic Society, 2019). In this particular case, we only model the subducting crust as having a distinct composition from the subducting harzburgite and hence their behavior will differ from one another. However, our view (and hope) is that the ability to model the interface between other distinct regions of this problem will be of significant interest to geoscientists.

[3]Harzburgite, an ultramafic, igneous rock, is a variety of peridotite consisting mostly of the two minerals, olivine and low-calcium pyroxene; it is named for occurrences in the Harz Mountains of Germany. It commonly contains a few percent chromium-rich spinel as an accessory mineral.

Variable	Quantity	Mantle	SP Crust	SP Harz.	OP Crust	OP Harz.
A	Prefactor	8.571×10^{-16}	5.000×10^{-20}	8.571×10^{-16}	8.571×10^{-16}	8.571×10^{-16}
n	Stress Exponent	1.	1.	1.	1.	1.
m	Grain Size Exponent	3.	0.	3.	3.	3.
E	Activation Energy [$\mathrm{J\,mol^{-1}}$]	335.0×10^3	0.0	335.0×10^3	335.0×10^3	335.0×10^3
V	Activation Volume [$\mathrm{m^3/mol}$]	4.0×10^{-6}	0.0	4.0×10^{-6}	4.0×10^{-6}	4.0×10^{-6}
ϕ	Angle of internal friction	25.0°	25.00°	25.00°	25.00°	25.00°
C	Cohesion [Pa]	10.0×10^6	10.0×10^6	10.0×10^6	10.0×10^6	10.0×10^6
ρ	Density [$\mathrm{kg/m^3}$]	3300	3000	3235	3000	3235
C_p	Heat capacity [$\mathrm{J\,kg^{-1}\,K^{-1}}$]	1250	1250	1250	1250	1250
κ	Thermal diffusivity [$\mathrm{m^2/s}$]	1.0×10^{-6}	1.0×10^{-6}	1.0×10^{-6}	1.0×10^{-6}	1.0×10^{-6}
α	Thermal expansivity [$\mathrm{K^{-1}}$]	3.1×10^{-5}	3.1×10^{-5}	3.1×10^{-5}	3.1×10^{-5}	3.1×10^{-5}

Table 5.1: Values of the constants for each material in the subducting slab computation. Note that OP refers the the overriding plate, and SP to the subducting plate.

As with the previous mantle convection model problem in Section 5.1, we use the Boussinesq approximation to model the effect of the temperature on the material density, and compressibility in the momentum equation. However, for the viscosity parameter in this problem we use the viscoplastic material model that is implemented in ASPECT, which is significantly more complex than the material models we used in the other problems presented in this dissertation. As noted in Section 2.7, the ability to use the same material models as with the other advection algorithms in ASPECT, without incurring additional development cost and the risk of introducing model errors by duplicating existing implementations is one of the primary reasons for the choosing this approach. Our use of such a more complex material model than in our previous computations is therefore a useful demonstration of the benefits of this approach.

A description of the precise material model implementation is given in detail in the ASPECT manual (Bangerth et al., 2019) Here we will include a brief overview of the relevant aspects of this material model to this particular application. The material model we use in our computations contains both a physical viscosity model, which is based on diffusion creep and a plasticity yield criterion to limit the viscosity. The diffusion creep viscosity equation is

$$\mu = \frac{1}{2} A^{-\frac{1}{n}} d^{\frac{m}{n}} \dot{\varepsilon}_{ii}^{\frac{1-n}{n}} \exp\left(\frac{E + pV}{nRT}\right) \tag{5.3}$$

and the Drucker-Prager yield criterion (for two-dimensions) is

$$\sigma_y = C\,\cos(\phi) + p\,\sin(\phi)\,. \tag{5.4}$$

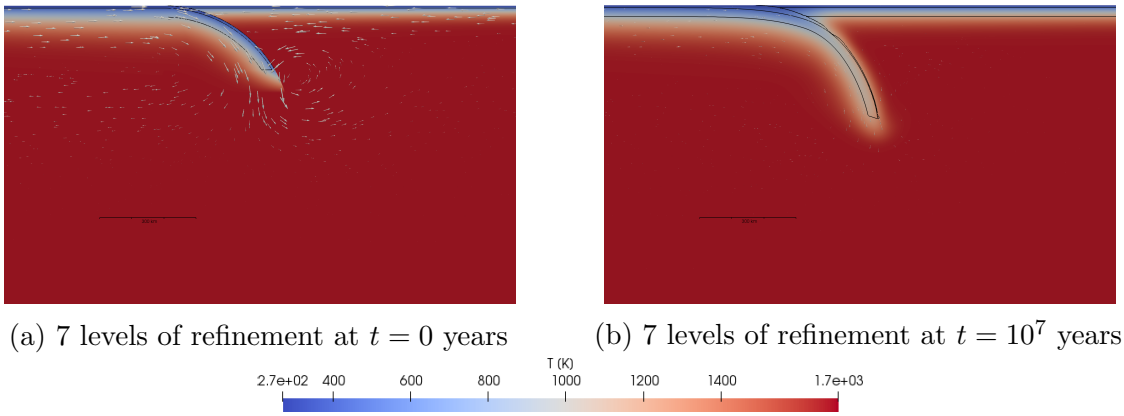
Here, if the stress is greater than the yield stress (ie $\sigma \geq \sigma_y$) then, the viscosity is set to $\mu = \frac{\sigma_y}{2\varepsilon}$. For the sake of ensuring stability of the model, the viscosity is then constrained to satisfy $1 \times 10^{19} \leq \mu \leq 1 \times 10^{24}$.

In the above equations $d = 8.2 \times 10^{-3}$ m is the grain size, $\dot{\varepsilon}_{ii}$ is the square root of the scond invariant of the deviatoric strain rate tensor, p is pressure, R is the gas constant, and T is temperature. The relevant material constants and the values for each of the distinct materials are defined in Table 5.1.

We computed this problem on a 5.780×10^6 m by 2.89×10^6 m Cartesian mesh, with AMR allowing cells sizes to range between approximately 1.80×10^5 m and 1.41×10^3 m with 7 levels of adaptive mesh refinement beyond the initial four levels of refinement; i.e., the initial grid is of square cells of side approximately 1.80×10^5 m. (Note that ASPECT will refine the initial grid according to the grid refinement parameters *prior* to taking the first time step.) The initial state of the distinct compositional elements is a layer cake at the upper mesh boundary with a very localized variation as shown in Figure 5.14. The variation shown in this figure consists of the subducting plate being extended into a 4×10^5 m radius quarter circle arranged such that the variation is nearly centered in the mesh, with the overriding plate boundary beginning as shown in the figure. For the overriding and subducting plate the crust depth is 7.5×10^3 m and the harzburgite depth is 3.52×10^4 m, respectively.

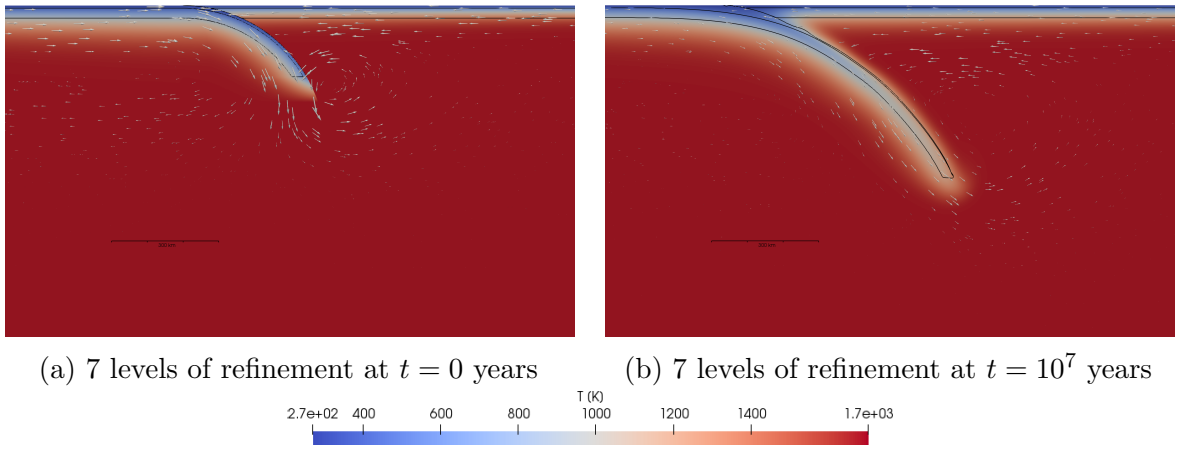
5.3.1 Discussion of the Subducting Slab Computations

Figures 5.15 and 5.16 show the initial state of the subducting slab at $t = 0$ years and at the later time of $t = 10^7$ years computed with the VOF and DG algorithms, respectively. The noticeable contrast in the observed solutions, in particular the length of the subducted region of the slab at $t = 1 \times 10^7$ yr, is caused by a noticeable difference in the velocity profiles. Below we will make a more detailed examination of the differences between the solutions produced by the two advection algorithms. Here and in all of the following



(a) 7 levels of refinement at $t = 0$ years (b) 7 levels of refinement at $t = 10^7$ years

Figure 5.15: The state of the subducting slab, which we computed with the VOF interface tracking method, at the initial time of $t = 0$ years and at the later time of $t = 1 \times 10^7$ years. The colorscale is the temperature field with the material interfaces superimposed in black. All of the material interfaces we computed with the VOF method shown here and below are the actual interfaces as reconstructed by the VOF method as described in Section 2.3 above. Note that the slight rounding of sharp corners, such as the tip of the crustal slab, is due to the VOF interface tracking algorithm and, in general, tends to also occur with other interface tracking algorithms. The arrow glyphs indicate the current velocity field.



(a) 7 levels of refinement at $t = 0$ years (b) 7 levels of refinement at $t = 10^7$ years

Figure 5.16: The state of the subducting slab, which we computed with the DG advection method, at the initial time of $t = 0$ years and at the later time of $t = 1 \times 10^7$ years. Unless noted otherwise all of the material interfaces we display from our computations with the DG method shown here and below are contours of the composition C at $C = \frac{1}{2}$. The colorscale is temperature. The arrow glyphs indicate the current velocity field.

figures the times quoted are in years. All mention of levels of refinement are with respect to the initial 4 levels of refinement. Thus, 7 levels of refinement indicate a grid with side $h = \mathrm{L}/2^{11}$ where $\mathrm{L} = 5.780 \times 10^6$ m denotes the length of the computational domain.

Note that this very finest level of refinement (i.e., 7 levels) is *only* on the subducting slab boundaries.

Our primary goal in using our new VOF interface tracking methodology to compute this model problem is to demonstrate the suitability of the VOF method for modeling of this type of mantle convection / geodynamics problem. The problem as defined here includes some simplifications that are likely to make any conclusions one might draw from the computational results inapplicable to the physical system upon which it is based. However, it is sufficiently representative of more realistic versions of the problem to demonstrate the usefulness of our approach to problems of this type. We note that this problem is expected to have qualitatively distinct modes of behavior that are based on an uncertain set of parameters, which is one of the primary goals of numerical studies of it (Billen and Arredondo, 2018; Arredondo and Billen, 2016; Agrusta et al., 2017).

In particular, since here we have computed the same problem with the VOF interface tracking methodology and the Discontinuous Galerkin method, which is not an interface tracking method[4], the inherent differences in the computational results produced by the two methods leads to differences in the qualitative behavior of the slabs in the long time limit. Thus, the comparison of computations made with an interface tracking method and an interface capturing method or, with different advection methods, might indicate the effect of one or more parameters of interest on the computation or perhaps even the degree to which some parameter influences the model's relevance to the real world problem.

Another important point we wish to make here concerns the possibility that a particular computation may be under-resolved. For example, examination of the plot in figure 5.17 suggests that the initial condition is not resolved on the coarser mesh at a refinement level of 6. The initial behavior of the more refined DG computation, at a refinement level of 7, suggests that the VOF method is better able to produce the correct behavior at this resolution. In other words, note that in this figure that initially — for approximately one to two million years — the RMS velocity as computed with the DG method at a refinement

[4]Many computational scientists, including the author, refer to advection methods such as the DG method, which do not explicitly produce a sharp interface, as "interface capturing" methods rather than interface tracking methods.

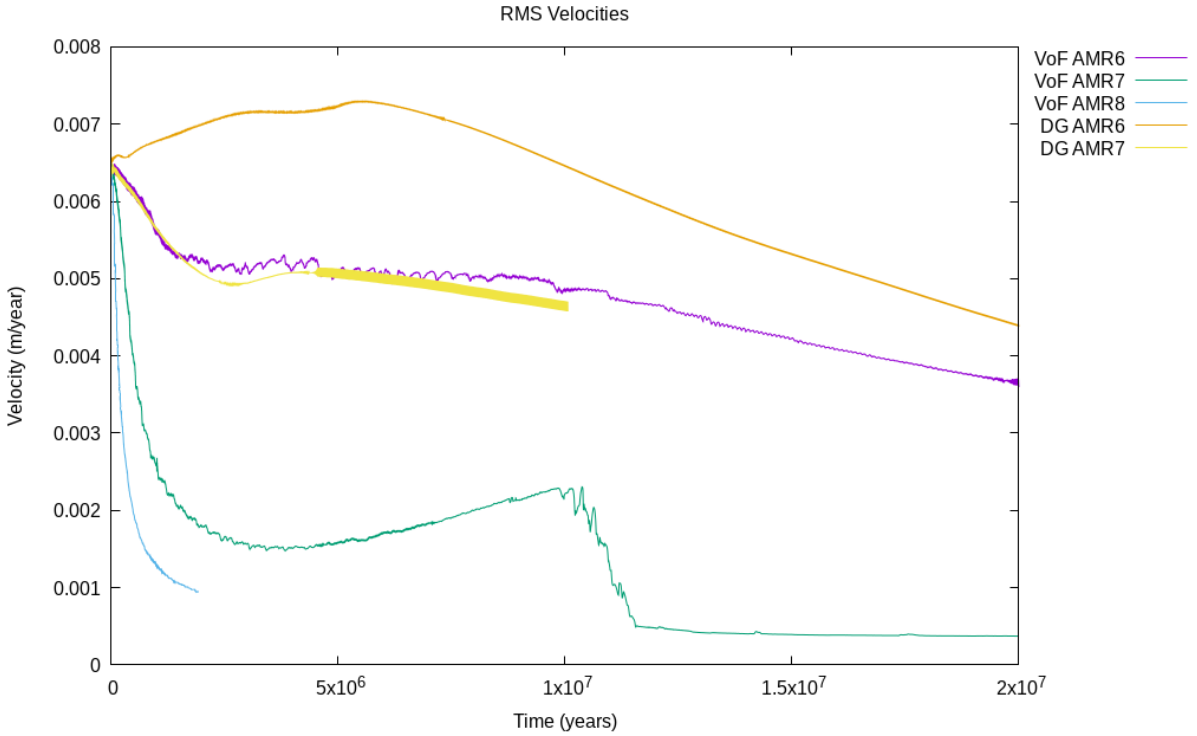
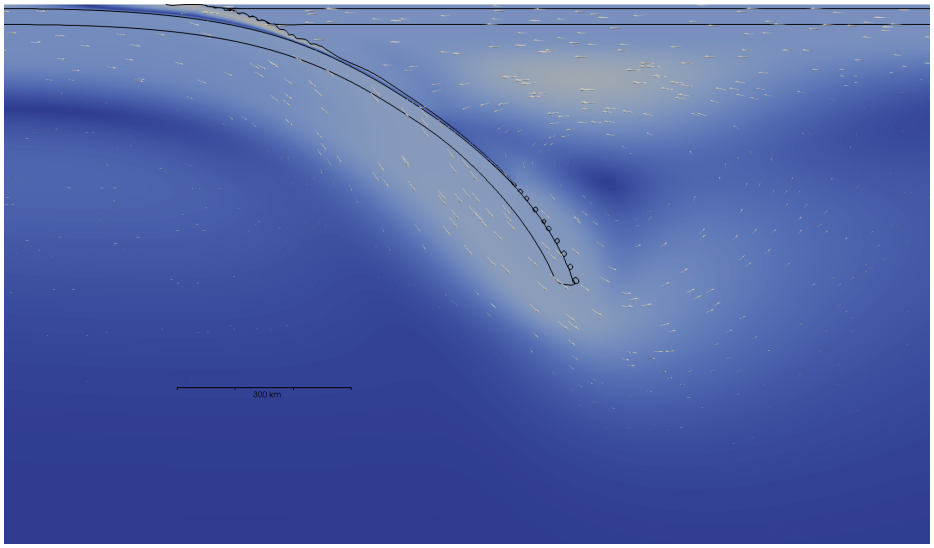


Figure 5.17: Comparison of root mean square (RMS) velocities for various computations organized by the method and the number of levels of refinement. We note that this includes computations that were not run to completion (i.e., time $t = 10^7$ or $t = 2 \times 10^7$years), since we did not have sufficient computational resources to make a longer VOF computation at a refinement level of 8 at the time. (This will be part of our future work.) This plot clearly shows that greater refinement results in lower velocities. This explains the difference in the depth of the subducting slabs in, for example, Figure 5.20 below.

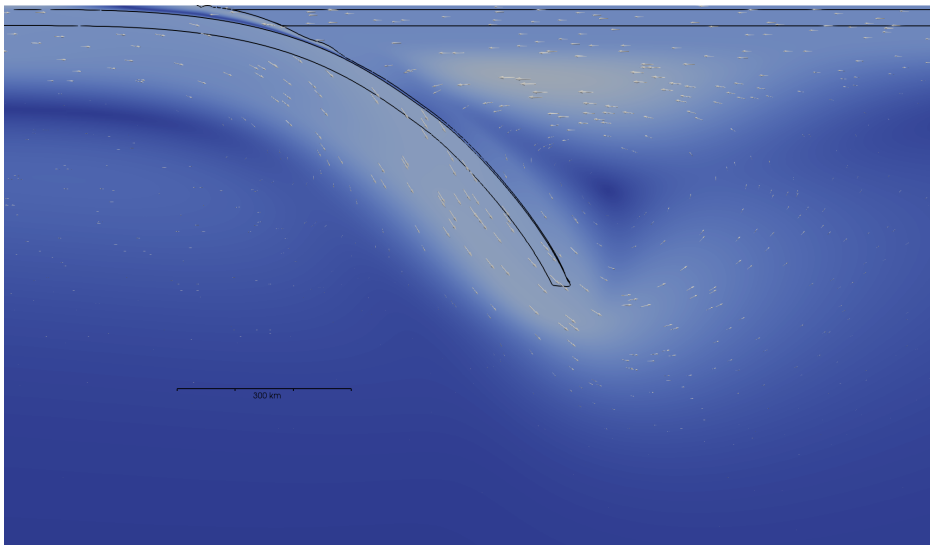
level of 7 is almost identical to that of the VOF method computed at a refinement level of 6. Making a visual comparison of the corresponding velocity fields in Figure 5.18 below suggests that this similarity is also true for the full velocity field.

Without AMR, the scale of this problem would make the computation prohibitively expensive, as some of the features of interest are prohibitively small relative to the scale of other necessary features. This is made even more clear when the need for the model to extend sufficiently deep that only the effects of the upper boundary condition on the solution is taken into account.

Another fact that must be emphasized is that anyone who uses *any* computational model, but in particular one with a VOF interface tracking or DG interface capturing



(a) State of VOF subducting slab with 6 levels of refinement at $t = 1 \times 10^7$ years.



(b) State of DG subducting slab with 7 levels of refinement at $t = 1 \times 10^7$ years.

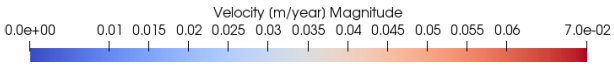
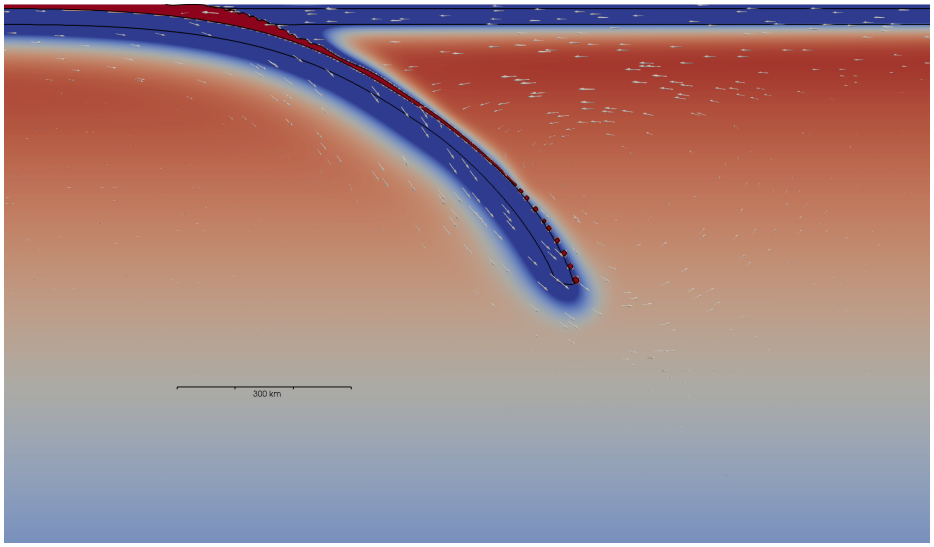
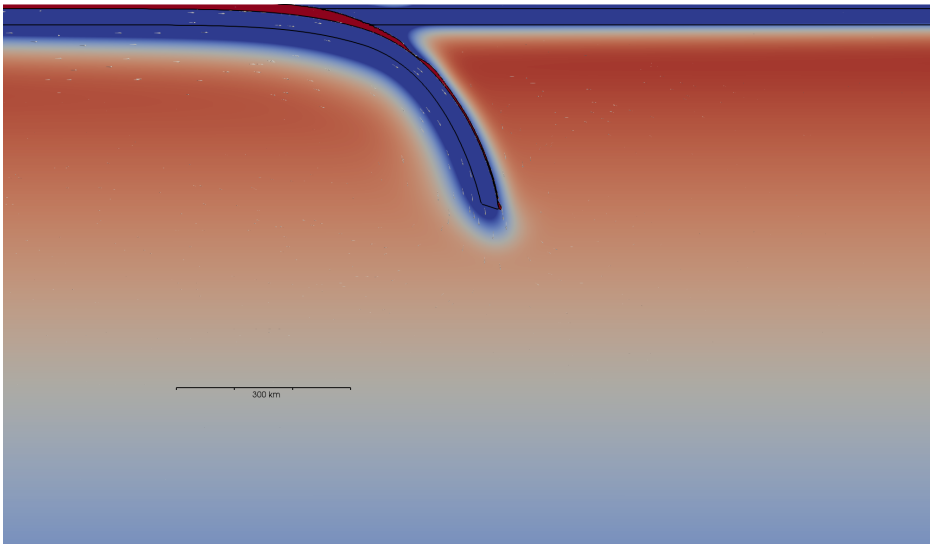


Figure 5.18: A comparison of the states for the subducting slab model at $t = 1 \times 10^7$ years when we used the VOF method with *6 levels of refinement* versus the DG method with *7 levels of refinement*. We note that in this computation, at this time, a visual comparison suggests that the computation we made with the VOF method with 6 levels of refinement is roughly equivalent to the computation we made of the same problem with the DG method and one additional level of refinement. However, we also note that the droplet-like features that appear near the end of the crustal slab in (a) is a numerical artifact that is caused by the VOF method being under-resolved. See Puckett (2014) for a proof of how many cells one needs to use with a VOF method to resolve a feature such as the end of the crustal slab in order for it to be fully resolved and second-order accurate. The colorscale is the magnitude of the velocity.



(a) State of VOF subducting slab with 6 levels of refinement at $t = 1 \times 10^7$ years.



(b) State of VOF subducting slab with 7 levels of refinement at $t = 1 \times 10^7$ years

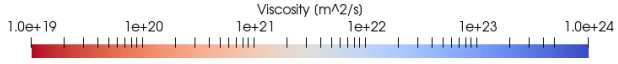
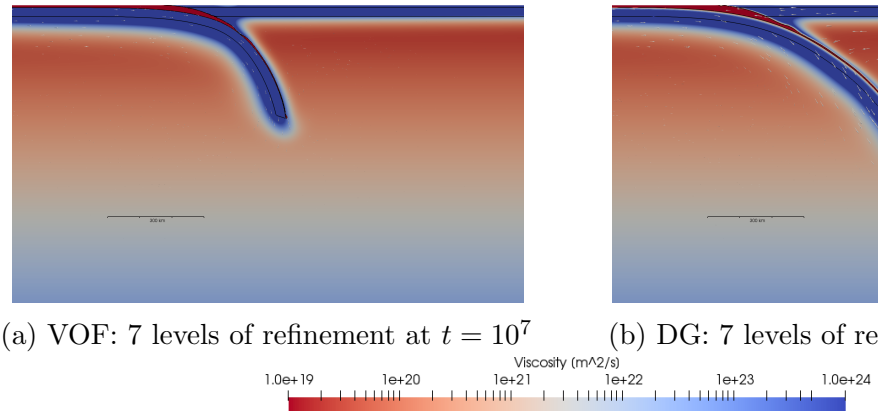


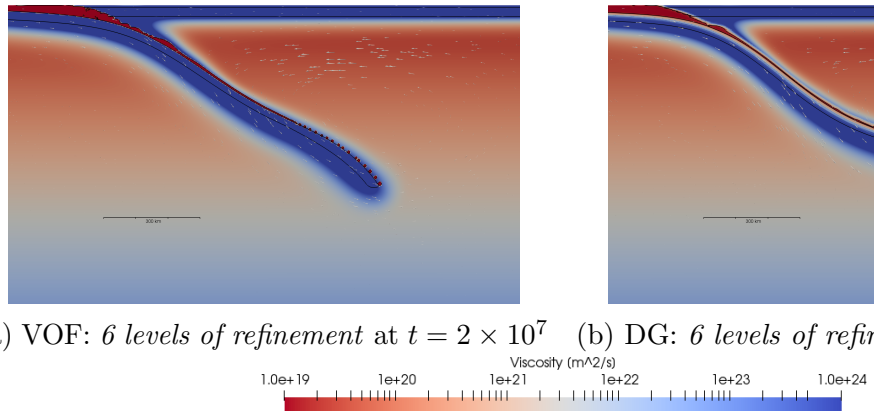
Figure 5.19: Comparison of the state of the subducting slab computed with the VOF method at refinement level 6 and refinement level 7 at $t = 1 \times 10^7$. The colorscale is viscosity. We have used the interface as reconstructed by the VOF method as described in Section 2.3 to show the boundaries in both (a) and (b). For refinement level 6 the features in red that look like droplets on the subducting crust in (a) are a numerical artifact, indicating the computation is under-resolved at this time. However, at this same time it appears that most of the crustal slab in (b) is reasonably well-resolved. In (b) one way to determine if the small piece of crust that is detaching from the tip of the crustal slab, as well as the thinning and separation further up the slab, is well-resolved at this time is to make at least one, and preferably several, increasingly refined computations of this particular problem and observe if the computational results converge to the same image, or if it changes with greater resolution. The arrow glyphs indicate the current velocity field.



(a) VOF: 7 levels of refinement at $t = 10^7$ (b) DG: 7 levels of refinement at $t = 10^7$

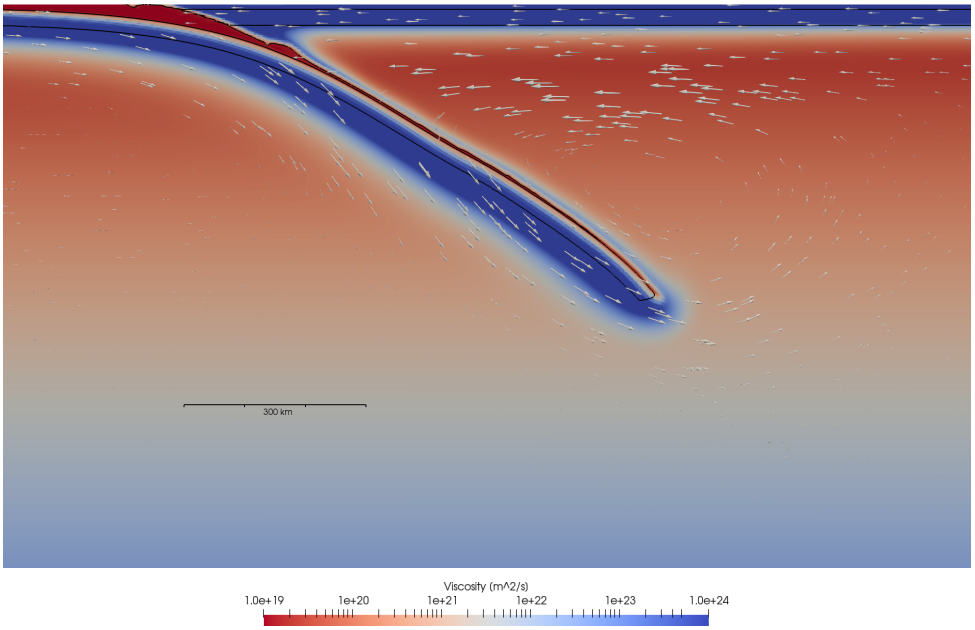
Figure 5.20: Comparison of the states of the subducting slab at time $t = 1 \times 10^7$ years, which we computed with (a) the VOF interface tracking method and (b) the DG advection method. As mentioned in the captions to Figures 5.15 and 5.16 above we have used the interface as computed by the VOF method to show the boundaries in (a) and contours to indicate all fluid boundaries computed with the DG method in (b). The colorscale is viscosity. The arrow glyphs indicate the current velocity field.

method is that a computation that is initially sufficiently well refined may require additional refinement at later times. This is amply illustrated in the figures from our computations of the subducting slab problem presented here, especially considering Figures 5.20 and 5.21.

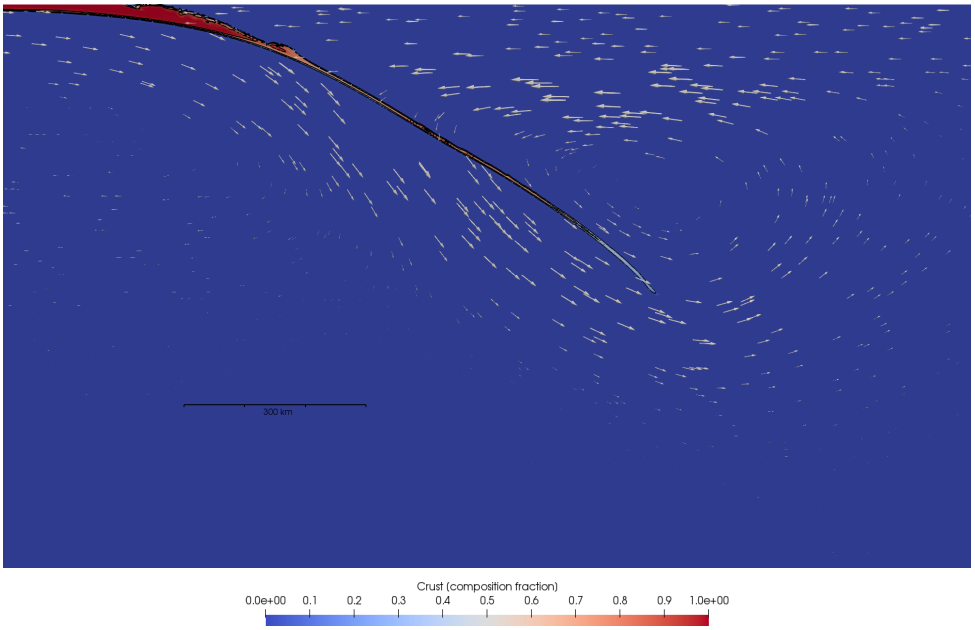


(a) VOF: *6 levels of refinement* at $t = 2 \times 10^7$ (b) DG: *6 levels of refinement* at $t = 2 \times 10^7$

Figure 5.21: Comparison of the state of the subducting slab computed with (a) VOF and (b) DG with *six levels of refinement at time* $t = 2 \times 10^7$ *years* The fluid boundaries. are determined using the methods for DG and VOF as described in the caption to Figures 5.15 and 5.16 above. The disconnected "bubble-like" contours for the subducting crust near the tip of the descending slab in (a) are a numerical artifact, which indicates that the model is *under-resolved* in that region (i.e., the tip of the subducting crustal slab) for only 6 levels of refinement. The colorscale is viscosity. The arrow glyphs indicate the current velocity field.



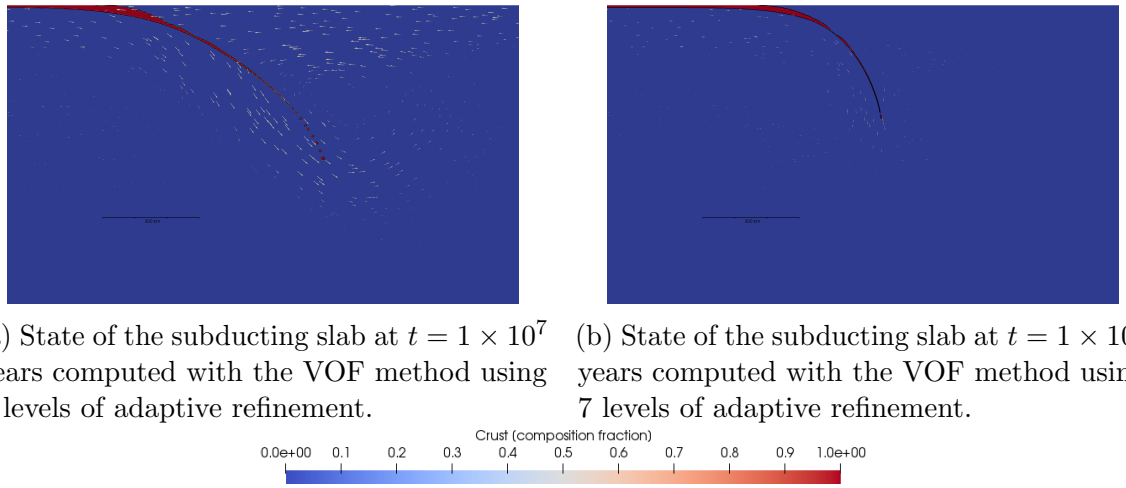
(a) DG w/6 levels of refinement at $t = 1 \times 10^7$



(b) Focus on the subducting crust slab. The colorscale is the subducting crust fraction, with contours to make regions of interest more clear. The contours for other compositions are not shown here to make the distinction as clear as possible.

Figure 5.22: State of the subducting slab at $t = 1 \times 10^7$ years computed with the DG method and 6 levels of refinement. The figures show contours at $C = 0.1, 0.5$ and 0.9 in black. The arrow glyphs indicate the velocity field.

We continue our analysis of the difference between the computations made with the VOF interface tracking method and with the DG advection method by highlighting those aspects of the problem that make the use of the VOF algorithm more desirable. In particular, consider Figure 5.22, where it is apparent that the inherent smoothing of the subducting crust composition interface by the DG advection method has resulted in a region of lower viscosity, which may extend beyond the correct region of influence. We note that this is especially apparent in Figure 5.22a, where we can see that a visibly lower viscosity region extends halfway through the harzburgite layer of the subducting slab. We also note that in Figure 5.22b it is apparent that the majority of the subducted crust layer is at a composition fraction $C \leq 0.9$, and the tip has a composition fraction $C \leq 0.5$. Among other concerns, given that the low viscosity of the subducting crust layer is a major factor in the time evolution of the subducting slab, this suggests that the numerical smoothing effect of the DG computation may be generating a qualitative error in the computational result. This computation at time $t = 2 \times 10^7$ years is clearly under-resolved as is apparent from the balls that have formed on the crustal slab. Looking at the result at this time leads us to conjecture that the almost all of this computation is under-resolved.



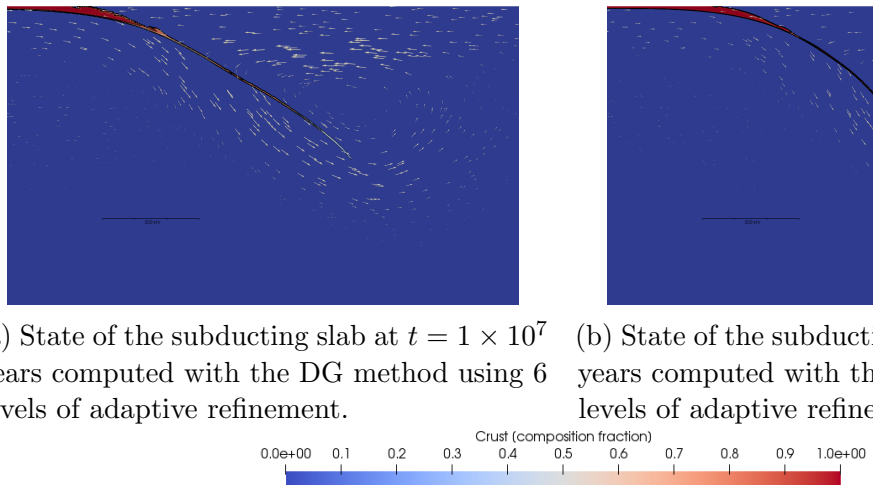
(a) State of the subducting slab at $t = 1 \times 10^7$ years computed with the VOF method using 6 levels of adaptive refinement.

(b) State of the subducting slab at $t = 1 \times 10^7$ years computed with the VOF method using 7 levels of adaptive refinement.

Figure 5.23: State of the subducting slab using the VOF advection algorithm for comparison with Figure 5.22. In both (a) and (b), only the subducting crustal slab is shown, and the retention of the sharp interface that is a design goal of the VOF algorithm may be observed. In (a), we note the existence of the "droplet" artifact characteristic of under-resolution, which we discussed in the caption to Figure 5.19.

In analyzing the results, it is necessary to note that in the coarsest computation the chosen level of refinement produces a crust depth that is less than three maximally refined cells in size, and therefore near the limit of the VOF method to resolve detailed structural features. This requirement does not differ fundamentally from other advection algorithms, as we can observe the same inability to resolve detailed structure with the DG advection method in Figure 5.20. (See also the detailed comparison of four advection methods in Puckett et al. (2018)). The need for greater refinement is reinforced by the observation of the droplet artifact discussed in Section 5.2, which is especially noticeable in Figure 5.20a, suggesting that the first priority for any further investigation into this problem should be to run another set of models with at least one additional level of refinement near the regions of interest. For both the DG and the VOF implementations, we can expect this additional level of refinement to be most influential on the behavior of the computation near the tip of the descending slab. Thus, a significant change in the qualitative behavior may suggest an investigation into sensitivity to the conditions to additional further refinement in that region in particular, without additional refinement of other parts of the slab. A comparison of the subducting crust configuration under additional refinement is show for both the VOF and DG methods in Figures 5.23 and 5.24.

We note that the VOF model produces a much lower velocity for both the surface trench



(a) State of the subducting slab at $t = 1 \times 10^7$ years computed with the DG method using 6 levels of adaptive refinement.

(b) State of the subducting slab at $t = 1 \times 10^7$ years computed with the DG method using 7 levels of adaptive refinement.

Figure 5.24: State of the subducting slab using the DG advection algorithm for comparison to Figure 5.22.

Component	VOF	DG
Total	$9.709\,96 \times 10^1$ s	$1.047\,74 \times 10^2$ s
Stokes solve	$8.070\,62 \times 10^1$ s	$9.482\,76 \times 10^1$ s
Advection update	5.435 06 s	4.476 12 s

Table 5.2: A comparison of the computation time required for the various component computations of an average single time step at 7 levels of adaptive refinement. This comparison of the time required for a single time step was chosen over the time required for a full computation since the two different algorithms compute the CFL condition in significantly different ways.

movement and the slab descent rate. The median velocities noted in the introduction of Billen and Arredondo (2018) for the physical system are approximately $0.9\,\mathrm{cm\,yr^{-1}}$ to $1.3\,\mathrm{cm\,yr^{-1}}$. That this does not match the values observed in the problem we have computed here, may indicate that additional features and / or physics need to be added to the model.

Comparing the initial behavior seen in Figure 5.17 suggests that the values produced by the VOF algorithm are most likely more accurate than those in the DG computation, more precisely, by one level of refinement. In other words, computing with the VOF interface tracking algorithm at a given level of refinement (say 6) may save one level of refinement over computing with the DG advection algorithm (say 7). This can be seen in the figure by noting that the DG RMS velocity with refinement level 7 and the VOF RMS velocity with refinement level 6 almost entirely overlap for the first six hundred thousand years, and the first major departure is at 1.6 million years. This may be attributed to the fact that the VOF algorithm does not permit numerical smearing of the initial sharp interface, while this is unavoidable for the DG algorithm, resulting in a broader region with a lower viscosity.

In Table 5.2, we see that the VOF algorithm takes slightly less time to complete a time step overall, though the main advection update takes slightly longer. However, we also note that the profile of the RMS velocity for the VOF method in Figure 5.17 matches that of the more refined DG method, suggesting a superior rate of convergence and thus

requiring less computational resources for a converged computation.

We also note that our use of a single VOF method to model the interfaces between three materials, rather than a more general n-fluid model (e.g., Hill and Shashkov (2013); Jemison et al. (2015); Anbarlooei and Mazaheri (2011)) permits the (three) different material types in our computations to overlap somewhat, especially near sharp bends in the initial state, such as near the tip of the subducting slab, as well as in other regions in Figure 5.21a. This overlap currently appears to be at an acceptable level relative to that expected for other advection algorithms, but is also an area that is open to additional improvement with future work, especially if further investigation suggests a sensitivity to such conditions exists in the problems under consideration. See Chapter 7 for a discussion and references regarding this issue.

5.4 Applicability of the Volume-of-Fluid Method

Since one of the goals of this dissertation is to explain how one can make use of a VOF method in their work to those not already familiar with the algorithm, here we present a brief overview of the types of problems it is applicable to.

The primary characteristic of the VOF algorithm is that it is a specialized algorithm, and therefore requires that certain assumptions hold in order to provide the greatly increased accuracy that is possible with an interface tracking algorithm. In particular, the algorithm is designed for cases where we are distinguishing between types of fluid, with the types both being mutually exclusive and do not mix; i.e. they are immiscible. Because of this specialization, use of the VOF algorithm enforces the existence of sharp boundaries between the fluids.

The primary benefit the volume of fluid method is that the interfaces between the tracked regions will remain sharp and, assuming the flow incompressible flow, volume will be conserved. Note that incompressible flow is not a necessary condition for one to employ a volume of fluid method, since it is possible to design VOF methods to model compressible flow, such as the equations of gas dynamics (Henderson et al., 1991; Miller and Puckett, 1996) As a non-rigorous description of the differences between the approaches,

we can describe the volume of fluid algorithm as having a numerical "surface tension" where more general FEM methods have a numerical "diffusion". It is necessary to note that these descriptions are intended more as intuitive labels for the categories rather than a mathematical or physical description of the behavior. In particular, the volume of fluid algorithm will tend to round off sharp corners, where general FEM methods will tend to spread out any sharp discontinuity at least to some extent (this also loses any sharp corners). In Section 5.2, we discuss some of the manners in which the volume of fluid algorithm may fail in cases where the interface is under-resolved, a concern which is not limited to the volume of fluid method. Arguably, the distinctive nature of the behavior of an under-resolved volume of fluid algorithm may be an advantage due to making it clear when a region of the model is under-resolved.

The primary concern unique to underresolved VOF codes is the existence of "jetsam", or small disconnected fluid volumes that effectively only occupy 1 or 2 model cells. As noted, these are a symptom of a severely underresolved region, as a more severe and less well behaved counterpart to the droplet stream noted in section 5.2. However, due to the nature of the VOF algorithm, they will display non-physical behavior almost entirely dependent on the implementation of the reconstruction and advection algorithms. The non-physical behavior they display may also render some assumptions invalid such as the interface mesh velocity limit based on the CFL condition used in the AMR strategy described in section 2.8, permitting cases that were intended to be handled elsewhere to occur. It is necessary to note that the simulation states most likely to induce such behavior are also likely to induce corresponding under-resolution symptoms with other advection algorithms. Therefore, if additional resolution is not possible or unreasonably costly, the decision is likely to be a choice of which approach is most resilient or degrades most gracefully with respect to the primary region of interest for the model.[5]

[5]Under some circumstances, it may be desirable to make use of an algorithm specifically due to the fact that it produces behaviors obviously caused by under-resolution due to this making it easier to diagnose when the current parameters are forcing an under-resolved solution

Chapter 6

CONCLUSIONS

We have implemented a Volume-of-Fluid (VOF) interface tracking method in the open source finite element code ASPECT, which is designed to model convection and other processes in the Earth's mantle. Our VOF method works efficiently and effectively in ASPECT's parallel environment and with it's adaptive mesh refinement (AMR) algorithm. We show that the VOF method reproduces linear interfaces in a constant flow to machine precision and is second-order accurate when we use it to compute a standard, smooth, interface tracking benchmark problem. We also demonstrate that the method shows excellent agreement with two benchmark problems from the computational mantle convection literature. In particular, in the second of these benchmarks we use AMR to allow us to compute at a much higher effective resolution at lower computational cost than would otherwise be possible.

We then use the new interface tracking methodology to study a problem involving thermochemical convection in density stratified flow. This model problem is relevant to the study of structures at the core mantle boundary known as Large Low Shear Velocity Provinces (LLSVPs). Recent studies utilizing seismic imaging have revealed large regions with anomalous seismic properties in the lower mantle. There are two dome-like regions beneath Africa and the Pacific with low shear-wave velocities that extend some 1000 km above the core-mantle boundary and have horizontal dimensions of several thousand kilometers (Cottaar and Romanowicz, 2012; French and Romanowicz, 2015). Most interpretations propose that the heterogeneities are compositional in nature, differing

from the surrounding mantle, an interpretation that would be consistent with chemical geodynamic models. Based on geological and geochemical studies it has been argued that LLSVPs have persisted for billions of years (Burke et al., 2008).

The model problem is designed to study the basic physics underlying the formation of thermal plumes that bring some of this material to the Earth's surface. In our computations we use AMR to obtain an effective grid resolution of 768×256 square cells overlaying the fluid interface on an underlying grid of 192×64 square cells. This increase in resolution confirms that for a certain range of the nondimensional buoyancy parameter B at Rayleigh number $\mathrm{Ra} = 10^5$ our computations of the interface have converged well enough to interpret with confidence the large scale dynamics of the two regions of differing densities.

Finally, we apply the new interface tracking methodology to a basic 2D slab subduction problem. This model problem involves complex interactions between more than two distinct materials, with significant nonlinearities. The results suggest that the use of a VOF interface tracking algorithm may allow some additional insights into the underlying physical processes, and possibly allow longer-term models.

In conclusion, the results of the work presented here demonstrate that our VOF interface tracking method should perform well on a number of problems of interest to the computational mantle convection community.

Chapter 7

FUTURE WORK

The primary target for future improvements to the volume of fluid implementation in ASPECT is to implement ability to compute 3D models on more general grids. There are a number of modifications that we will employ to make this possible, some of which are fundamentally required and others which may significantly improve the maintainability of the code. The three elements which must be updated to handle 3D are the volume calculation, the reconstruction step, and the advection step. The volume calculation is the most straightforward, and does not require any novel elements, instead only requiring effort to ensure that the selected approach is both efficient and accurate. The reconstruction and advection algorithm can be extended to 3D, but as many of the likely 3D target problems are partial spheres, alteration to handle non-Cartesian meshes is a simultaneous requirement. This would require either confirming that a simple extension remains valid and implementing the correct extension, or designing a new algorithm for this type of mesh. In the case of the advection algorithm, the addition of another dimensional split is relatively straightforward. However, we also plan to consider unsplit methods as well as increasing the efficiency of the current implementation.

The first of the above modifications to be discussed in more depth is the consideration of an unsplit advection update. Given our current design and some of our other goals, this work will be focused on selecting or designing an approach that will both maintain the required volume fraction bounds, and minimize the dependence on external real space computations. One of the core design requirements for this work is that our approach be

extensible to a 3D implementation. Current speculation on likely approaches suggest that the final implementation will consist of an approximation based flux calculation combined with some form of flux or slope limiter, for example one based on the Flux Corrected Transport algorithm (Zalesak, 1979) or a van Leer style slope limiter (Colella, 1990).

Another modification is to redesign the reconstruction algorithm to implement a standard least squares volume of fluid reconstruction (Puckett, 1991; Pilliod and Puckett, 1997, 2004), a coupled level set method (Sussman and Puckett, 2000a) or a moment of fluid reconstruction (Jemison et al., 2015). It may be suitable to include a configuration option for selection between the implemented versions of these due to some of them being more suitable for some problems than others, especially as the latter two approaches would require at least one matrix solve, and all would include a less constrained iterative minimization than the current reconstruction algorithm. In addition to being desirable for some applications of the volume of fluid interface tracking extension, the change in reconstruction algorithm may reduce some difficulties in implementation of the other improvements.

Finally, another important improvement is to allow variations in the refinement level of the interface. This will require modifications of both the reconstruction and advection steps, and should be included as part of the implementation of the above modifications considered for inclusion in the 3D algorithm.

References

Agrusta, R., Goes, S., van Hunen, J., 2017. Subducting-slab transition-zone interaction: Stagnation, penetration and mode switches. Earth and Planetary Science Letters 464, 10 – 23.

Anbarlooei, H. R., Mazaheri, K., 2011. Moment of fluid interface reconstruction method in axisymmetric coordinates. International Journal for Numerical Methods in Biomedical Engineering 27 (10), 1640–1651.

Arndt, D., Bangerth, W., Davydov, D., Heister, T., Heltai, L., Kronbichler, M., Maier, M., Pelteret, J.-P., Turcksin, B., Wells, D., 2017a. The `deal.II` library, version 8.5. Journal of Numerical Mathematics 25 (3), 137–146.

Arndt, D., Bangerth, W., Davydov, D., Heister, T., Heltai, L., Kronbichler, M., Maier, M., Pelteret, J.-P., Turcksin, B., Wells, D., 2017b. The `deal.II` library, version 8.5. Journal of Numerical Mathematics.

Arredondo, K. M., Billen, M. I., 2016. The effects of phase transitions and compositional layering in two-dimensional kinematic models of subduction. Journal of Geodynamics 100, 159 – 174, 200 years of geodynamic modelling.

Aulisa, E., Manservisi, S., Scardovelli, R., Zaleski, S., aug 2007. Interface reconstruction with least-squares fit and split advection in three-dimensional cartesian geometry. Journal of Computational Physics 225 (2), 2301–2319.

Bangerth, W., Dannberg, J., Gassmöller, R., Heister, T., et al., 2019. ASPECT: Advanced Solver for Problems in Earth's ConvecTion User Manual. Computational Infrastructure for Geodynamics, https://geodynamics.org/cig/software/aspect/aspect-manual.pdf.

Bangerth, W., Hartmann, R., Kanschat, G., 2007. deal.II – a general purpose object oriented finite element library. ACM Trans. Math. Softw. 33 (4), 24/1–24/27.

Benson, D. J., January 1998. Eulerian finite element methods for the micromechanics of heterogeneous materials: Dynamic prioritization of material interfaces. Computer Methods in Applied Mechanics and Engineering 151 (3-4), 343–360.

Billen, M. I., Arredondo, K. M., 2018. Decoupling of plate-asthenosphere motion caused by non-linear viscosity during slab folding in the transition zone. Physics of the Earth and Planetary Interiors 281, 17 – 30.

Burke, K., Steinberger, B., Torsvik, T. H., Smethurst, M. A., 2008. Plume generation zones at the margins of large low shear velocity provinces on the core–mantle boundary. Earth and Planetary Science Letters 265 (1), 49–60.

Burstedde, C., Wilcox, L. C., Ghattas, O., 2011a. `p4est`: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. SIAM Journal on Scientific Computing 33 (3), 1103–1133.

Burstedde, C., Wilcox, L. C., Ghattas, O., 2011b. `p4est`: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. SIAM Journal on Scientific Computing 33 (3), 1103–1133.

Chandrasekhar, S., 1961. Hydrodynamic and Hydromagnetic Stability. Dover, New York.

Chorin, A. J., 1985. Curvature and solidification. J. Comput. Phys. 57, 472–490.

Chorin, A. J., Marsden, J. E., 1993. A Mathematical Introduction to Fluid Mechanics, 4th Edition. No. 4 in Texts in Applied Mathematics. Springer-Verlag, New York, qA901.C53 1992.

Colella, P., 1990. Multidimensional upwind methods for hyperbolic conservation laws. J. Comput. Phys. 87, 171–200.

Cottaar, S., Romanowicz, B., 2012. An unusually large ULVZ at the base of the mantle near Hawaii. Earth and Planetary Science Letters 355, 213–222.

Courant, R., Friedrichs, K. O., Lewy, H., Mar. 1967. On the Partial Difference Equations of Mathematical Physics. IBM J. Res. Dev. 11 (2), 215–234, (English translation of the orignal work, "Uber die Partielien Differenzenglrichungen der Mathematischen Physik", *Math. Ann.* **100**, 32-74 (1928)).

Dannberg, J., Eilon, Z., Faul, U., Gassmller, R., Moulik, P., Myhill, R., 2017. The importance of grain size to mantle dynamics and seismological observations. Geochemistry, Geophysics, Geosystems 18 (8), 3034–3061.

Dannberg, J., Heister, T., 2016. Compressible magma/mantle dynamics: 3d, adaptive simulations in ASPECT. Geophysical Journal International 207 (3), 1343–1366.

Davaille, A., 1999. Two-layer thermal convection in miscible viscous fluids. J. Fluid Mech. 379, 223253.

Donea, J., Huerta, A., 2005. Steady Transport Problems. John Wiley and Sons.

Elgeti, S., Sauerland, H., February 2015. Deforming fluid domains within the finite element method: Five mesh-based tracking methods in comparison. Archives of Computational Methods in Engineering 23 (2), 323–361.

Enright, D., Fedkiw, R., Ferziger, J., Mitchell, I., 2002. A hybrid particle level set method for improved interface capturing. Journal of Computational Physics 183 (1), 83 – 116.

French, S. W., Romanowicz, B., 2015. Broad plumes rooted at the base of the Earth's mantle beneath major hotspots. Nature 525 (7567), 95–99.

Gassmoeller, R., 2016. Open source support for massively parallel, generic finite element methods. Poster presented at the 2016 NSF SI2 PI Workshop.

Gassmöller, R., Lokavarapu, H., Heien, E., Puckett, E. G., Bangerth, W., 2018. Flexible and scalable particle-in-cell methods with adaptive mesh refinement for geodynamic computations. Geochem. Geophy. Geosy. 19, 472–490.

Gerya, T. V., Yuen, D. A., 2003. Characteristics-based marker-in-cell method with conservative finite-differences schemes for modeling geological flows with strongly variable transport properties. Physics of the Earth and Planetary Interiors 140 (4), 293–318.

Guermond, J.-L., Pasquetti, R., Popov, B., 2011. Entropy viscosity method for nonlinear conservation laws. J. Comput. Phys. 230 (11), 4248 – 4267, Special issue High Order Methods for CFD Problems.

Hager, B., O'Connell, R., 1981. A simple global model of plate dynamics and mantle convection. Journal of Geophysical Research: Solid Earth 86 (B6), 4843–4867.

Hager, B. H., Clayton, R. W., 1989. Constraints on the structure of mantle convection using seismic observations, flow models, and the geoid. In: Mantle Convection, Plate Tectonics and Global Dynamics. Gordon and Breach Science Publishers, pp. 657–763.

He, Y., Billen, M. I., Puckett, E. G., Kellogg, L. H., 2016. A Discontinuous Galerkin method for solving time-dependent convection-diffusion temperature equations : Demonstration and comparison with other methods in the mantle convection code ASPECT. Talk given at the annual AGU Fall Meeting.

He, Y., Puckett, E. G., Billen, M. I., 2017. A discontinuous Galerkin method with a bound preserving limiter for the advection of non-diffusive fields in solid Earth geodynamics. PEPI 263, 23–37.

Heister, T., Dannberg, J., Gassmöller, R., Bangerth, W., 2017. High accuracy mantle convection simulation through modern numerical methods. II: Realistic models and problems. Geophysical Journal International 210 (2), 833–851.

Helmsen, J. J., Colella, P., Puckett, E. G., 1997. Non-convex profile evolution in two dimensions using volume of fluids. Technical Report LBNL-40693, Lawrence Berkeley National Laboratory.

Helmsen, J. J., Colella, P., Puckett, E. G., Dorr, M., January 1996. Two new methods for simulating photolithography development in three dimensions. In: Proceeedings of the

10th SPIE Optical/Laser Microlithography Conference. Vol. 2726. SPIE, San Jose, CA, pp. 253–261.

Henderson, L. F., Colella, P., Puckett, E. G., March 1991. On the refraction of shock waves at a slowfast gas interface. J. Fluid Mech. 224, 1–27.

Hill, R. N., Shashkov, M., 2013. The symmetric moment-of-fluid interface reconstruction algorithm. J. Comput. Phys. 249, 180 – 184.

Hirt, C. W., Nichols, B. D., 1981. Volume of Fluid (VOF) method for the dynamics of free boundaries. J. Comput. Phys. 39, 201–225.

Huber, R., Helmig, R., 1999. Multiphase flow in heterogeneous porous media: A classical finite element method versus an implicit pressure–explicit saturation-based mixed finite element–finite volume approach. International Journal for Numerical Methods in Fluids 29 (8), 899–920.

Jemison, M., Sussman, M., Shashkov, M., 2015. Filament capturing with the multimaterial moment-of-fluid method. J. Comput. Phys. 285, 149–172.

John, F., 1978. Partial Differential Equations, 4th Edition. Vol. 1 of Applied Mathematical Sciences. Springer-Verlag, New York.

Kellogg, L. H., Hager, B. H., van der Hilst, R. D., 1999. Compositional stratification in the deep mantle. Science 283 (5409), 1881–1884.
URL https://science.sciencemag.org/content/283/5409/1881

King, S., Raefsky, A., Hager, B., 1990. ConMan: Vectorizing a finite element code for incompressible two-dimensional convection in the Earth's mantle. Physics of the Earth and Planetary Interiors 59 (3), 195–207.

Korzekwa, D. R., Kothe, D. B., Lam, K. L., Puckett, E. G., Tubesing, P. K., Williams, M. W., 1999. A second-order accurate, linearity-preserving volume tracking algorithm for free surface flows on 3-d unstructured meshes. In: Proceedings of the 3rd ASME /JSME

Joint Fluids Engineering Conference. FEDSM99-7109. American Society of Mechanical Engineers, San Francisco, CA, pp. 1–6.

Kronbichler, M., Bangerth, W., 2011. Numerical experiments to determine optimal parameters. Reference documentation for deal.II, The step-31 tutorial program.

Kronbichler, M., Heister, T., Bangerth, W., 2012. High accuracy mantle convection simulation through modern numerical methods. Geophysical Journal International 191 (1), 12–29.

Lax, P. D., Mar. 1967. Hyperbolic Difference Equations: A Review of the Courant-Friedrichs-Lewy Paper in Light of Recent Developments. IBM J. Res. Dev. 11 (2), 235–238.

Le Bars, M., Davaille, A., 2004. Large interface deformation in two-layer thermal convection of miscible viscous fluids. J. Fluid Mech. 499, 75110.

Le Bars, M., Davaille, A., 2005. Thermochemical convection in two superimposed miscible viscous fluids. In: Gutkowski, W., Kowalewski, T. (Eds.), Mechanics of the 21st Century, Proceedings of the 21st International Congress of Theoretical and Applied Mechanics. Springer Verlag, pp. FM7–12126.

LeVeque, R. J., April 1996. High-resolution conservative algorithms for advection in incompressible flow. SIAM J. Numer. Anal. 33 (2), 627–665.

López, J., Hernández, J., Gómez, P., Faura, F., 2016. A new volume conservation enforcement method for PLIC reconstruction in general convex grids. J. Comput. Phys. 316, 338–359.

Manga, M., 1996. Mixing of heterogeneities in the mantle: Effect of viscosity differences. Geophysical Research Letters 23 (4), 403–406.

Manga, M., Stone, H., 1994. Interactions between bubbles in magmas and lavas: effects of bubble deformation. Journal of Volcanology and Geothermal Research 63 (3), 267–279.

Manga, M., Stone, H. A., O'Connell, R. J., 1993. The interaction of plume heads with compositional discontinuities in the earth's mantle. Journal of Geophysical Research: Solid Earth 98 (B11), 19979–19990.

Maronnier, V., Picasso, M., Rappaz, J., November 1999. Numerical simulation of free surface flows. Journal of Computational Physics 155 (2), 439–455.

Mashayek, F., Ashgriz, N., June 1995. A hybrid finite-element-volume-of-fluid method for simulating free surface flows and interfaces. International Journal for Numerical Methods in Fluids 20 (12), 1363–1380.

McNamara, A., Zhong, S., 2004. Thermochemical structures within a spherical mantle: Superplumes or piles?: Thermochemical structures. Journal of Geophysical Research: Solid Earth 109 (B7), n/a–n/a.

Miller, G. H., Puckett, E. G., February 1994. Edge effects in molybdenum-encapsulated molten silicate shock wave targets. J. Appl. Phys. 75 (3), 1426–1434.

Miller, G. H., Puckett, E. G., August 1996. A high-order Godunov method for multiple condensed phases. J. Comput. Phys. 128 (1), 134–164.

Moresi, L., Gurnis, M., 1996. Constraints on the lateral strength of slabs from three-dimensional dynamic flow models. Earth and Planetary Science Letters 138 (1), 15–28.

Nagrath, S., Jansen, K. E., Lahey, R. T., oct 2005. Computation of incompressible bubble dynamics with a stabilized finite element level set method. Computer Methods in Applied Mechanics and Engineering 194 (42-44), 4565–4587.

National Geographic Society, 2019. The Earth's Mantle.
URL https://www.nationalgeographic.org/encyclopedia/mantle/

Nichols, B. D., Hirt, C. W., Hotchkiss, R. S., August 1980. SOLA-VOF: a solution algorithm for transient fluid flow with multiple free boundaries. Technical Report LA-8355, Los Alamos National Laboratory.

Noh, W. F., Woodward, P. R., June 28–July 3 1976. SLIC (Simple Line Interface Calculation). In: van de Vooren, A. I., Zandbergen, P. J. (Eds.), Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics. Vol. 59 of Lecture Notes in Physics. Springer-Verlag, Twente University, Enschede, pp. 330–340.

Osher, S., Fedkiw, R., 2003. Level Set Methods and Dynamic Implicit Surfaces, 1st Edition. Vol. 153 of Applied Mathematical Sciences. Springer-Verlag, New York.

Parker, B. J., Youngs, D. L., February 1992. Two and three dimensional eulerian simulation of fluid flow with material interfaces. Technical Report 01/92, UK Atomic Weapons Establishment, Aldermaston, Berkshire.

Pilliod, J. E., September 1992. An analysis of piecewise linear interface reconstruction algorithms for volume-of-fluid methods. MS Thesis, Graduate Group in Applied Mathematics, University of California, Davis.

Pilliod, J. E., Puckett, E. G., Aug. 1997. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. Technical Report LBNL-40745, Lawrence Berkeley National Laboratory.

Pilliod, J. E., Puckett, E. G., 1998. An unsplit, second-order accurate Godunov method for tracking deflagrations and detonations. In: Houwing, A. F. P., Paull, A., Boyce, R. R., Danehy, P. M., Hannemann, H., Kurtz, J. J., McIntyre, T. J., McMahon, S. J., Mee, D. J., Sandeman, R. J., Tanno, H. (Eds.), Proceedings of the 21st International Symposium on Shock Waves. Vol. II. Panther Publishing, Fyshwick, Australia, pp. 1053–1058.

Pilliod, J. E., Puckett, E. G., September 2004. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. J. Comput. Phys. 199 (2), 465–502.

Puckett, E. G., 1991. A volume-of-fluid interface tracking algorithm with applications to computing shock wave refraction. In: Proceedings of the Fourth International Symposium on Computational Fluid Dynamics. pp. 933–938.

Puckett, E. G., February 2010a. On the second-order accuracy of volume-of-fluid interface reconstruction algorithms: Convergence in the max norm. CAMCoS 5 (1), 99–148.

Puckett, E. G., October 2010b. A volume-of-fluid interface reconstruction algorithm that is second-order accurate in the max norm. CAMCoS 5 (2), 199–220.

Puckett, E. G., January 2014. On the second-order accuracy of volume-of-fluid interface reconstruction algorithms II: An improved constraint on the cell size. CAMCoS 8 (1), 123–158.

Puckett, E. G., Almgren, A. S., Bell, J. B., Marcus, D. L., Rider, W. J., January 1997. A high-order projection method for tracking fluid interfaces in variable density incompressible flows. J. Comput. Phys. 130 (2), 269–282.

Puckett, E. G., Miller, G. H., 1996. The numerical computation of jetting impacts. In: Sturtevant, B., Shepherd, J. E., Hornung, H. (Eds.), Proceedings of the 20th International Symposium on Shock Waves. Vol. II. World Scientific, New Jersey, pp. 1467–1472.

Puckett, E. G., Turcotte, D. L., He, Y., Lokavarapu, H., Robey, J. M., Kellogg, L. H., 2018. New numerical approaches for modeling thermochemical convection in a compositionally stratified fluid. Phys. Earth Planet. Inter 276, 10–35, special Issue:15th SEDI conference.

Rider, W. J., Kothe, D. B., Apr. 1998. Reconstructing volume tracking. J. Comput. Phys. 141 (2), 112–152.

Robey, J. M., Puckett, E. G., 2019. Implementation of a volume-of-fluid method in a finite element code with applications to thermochemical convection in a density stratified fluid in the earths mantle. Computers & Fluids 190 (2), 217–253.

Samuel, H., Evonuk, M., 2010. Modeling advection in geophysical flows with particle level sets. Geochemistry, Geophysics, Geosystems 11 (8).

Scardovelli, R., Zaleski, S., 2000. Analytical relations connecting linear interfaces and volume fractions in rectangular grids. Journal of Computational Physics 164 (1), 228 – 237.

Scardovelli, R., Zaleski, S., 2003. Interface reconstruction with least-square fit and split eulerian-lagrangian advection. International Journal for Numerical Methods in Fluids 41 (3), 251–274.

Schubert, G., Turcotte, D. L., Olson, P., 2001. Mantle convection in the Earth and planets. Cambridge University Press.

Sethian, J. A., June 1999. Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences, 2nd Edition. Vol. 3 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge, U.K. ; New York.

Sheldon, J., Cardwell Jr, W., et al., 1959. One-dimensional, incompressible, noncapillary, two-phase fluid flow in a porous medium. Petroleum Transactions, AIME 216, 290–296.

Shin, S., Lee, W. I., 2000. Finite element analysis of incompressible viscous flow with moving free surface by selective volume of fluid method. International Journal of Heat and Fluid Flow 21 (2), 197 – 206.

Steinberger, B., 2000. Plumes in a convecting mantle: Models and observations for individual hotspots. Journal of Geophysical Research: Solid Earth 105 (B5), 11127–11152.

Strang, W. G., 1968. On the construction and comparison of difference schemes. SIAM J. Numer. Anal. 5 (3), 506–517.

Strang, W. G., 2016. Introduction to Linear Algebra, 5th Edition. Cambridge Wellsely Press.

Sussman, M., Puckett, E. G., Aug 2000a. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. Journal of Computational Physics 162 (2), 301–337.

Sussman, M., Smereka, P., Osher, S., 1994. A level set approach for computing solutions to incompressible two-phase flow. Journal of Computational Physics 114 (1), 146 – 159.

Sussman, M. S., Puckett, E. G., 2000b. A coupled level set and volume of fluid method for computing 3D and axisymmetric incompressible two-phase flows. J. Comput. Phys. 162, 301–337.

Tan, E., Choi, E., Thoutireddy, P., Gurnis, M., Aivazis, M., 2006. Geoframework: Coupling multiple models of mantle convection within a computational framework: Geoframework-mantle convection models. Geochemistry, Geophysics, Geosystems 7 (6), n/a–n/a.

Tornberg, A.-K., Engquist, B., May 2000. A finite element based level-set method for multiphase flow applications. Computing and Visualization in Science 3 (1), 93–101.

Torrey, M. D., Cloutman, L. D., Mjolsness, R. C., Hirt, C. W., December 1985. NASA-VOF2D: A computer program for incompressible flows with free surfaces. Technical Report LA-10612-MS, Los Alamos National Laboratory.

Torrey, M. D., Mjolsness, R. C., Stein, L. R., July 1987. NASA-VOF3D: A three-dimensonal computer program for incompressible flows with free surfaces. Technical Report LA-11009-MS, Los Alamos National Laboratory.

Tryggvason, G., Scardovelli, R., Zaleski, S., 2011. Direct Numerical Simulations of Gas-Liquid Multiphase Flows, 1st Edition. Cambridge Monographs on Applied & Computational Mathematics. Cambridge University Press, Cambridge.

Turcotte, D. L., Schubert, G., 2014. Geodynamics, 3rd Edition. Cambridge University Press.

van Keken, P. E., King, S. D., Schmeling, H., Christensen, U. R., Neumeister, D., Doin, M.-P., 1997. A comparison of methods for the modeling of thermochemical convection. Journal of Geophysical Research: Solid Earth 102 (B10), 22477–22495.

Ventura, G., 2006. On the elimination of quadrature subcells for discontinuous functions in the eXtended finite-element method. International Journal for Numerical Methods in Engineering 66 (5), 761–795.

Wanner, G., Hairer, E., 1991. Solving ordinary differential equations II. Vol. 14 of Springer Series in Computational Mathematics. Springer-Verlag Berlin Heidelberg.

Weymouth, G., Yue, D. K.-P., Apr. 2010. Conservative volume-of-fluid method for free-surface simulations on Cartesian-grids. J. Comput. Phys. 229 (8), 2853–2865.

Williams, M. W., Kothe, D. B., Puckett, E. G., 1999a. Approximating interfacial topologies with applications for interface tracking algorithms. In: Proceedings of the 37th Aerospace Sciences Meeting and Exhibit. American Institute of Aeronautics and Astronautics, Reno, NV, pp. 1–9, aIAA-99-1076.

Williams, M. W., Kothe, D. B., Puckett, E. G., Nov. 1999b. Robust finite volume modeling of 3-D free surface flows on unstructured meshes. In: Proceedings of the 14th AIAA Computational Fluid Dynamics Conference. American Institute of Aeronautics and Astronautics, Norfolk, VA, pp. 1–6, a99-993567.

Zalesak, S. T., 1979. Fully multidimensional flux-corrected transport algorithms for fluids. Journal of computational physics 31 (3), 335–362.

Zhong, S., 2006. Constraints on thermochemical convection of the mantle from plume heat flux, plume excess temperature, and upper mantle temperature. Journal of Geophysical Research 111 (B4).

Zhong, S., Zuber, M., Moresi, L., Gurnis, M., 2000. Role of temperature-dependent viscosity and surface plates in spherical shell models of mantle convection. Journal of Geophysical Research: Solid Earth 105 (B5), 11063–11082.