UC Merced

UC Merced Electronic Theses and Dissertations

Title

Characteristic Bending and Structure-Preserving Methods for Incompressible Flow

Permalink

https://escholarship.org/uc/item/20n0m25k

ISBN

9798197864314

Author

Blomquist, Matthew

Publication Date

2026-01-30

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, MERCED

Characteristic Bending and Structure-Preserving Methods for Incompressible Flow

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Matthew Blomquist

Committee in charge:

Shilpa Khatri, Chair

Ann S. Almgren

François Blanchette

Changho Kim

Maxime Theillard

2026

All chapters © 2026 Matthew Blomquist

The dissertation of Matthew Blomquist is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Ann S. Almgren)

(François Blanchette)

(Changho Kim)

(Maxime Theillard)

(Shilpa Khatri, Chair)

University of California, Merced

2026

TABLE OF CONTENTS

Signature Page . . . iii

List of Figures . . . vi

List of Tables . . . viii

Acknowledgements . . . ix

Curriculum Vitae . . . xi

Publications and Contributions . . . xii

Abstract . . . xiii

Chapter 1 Introduction . . . 1

Chapter 2 Stable Nodal Projection Method on Octree Grids . . . 4
2.1 Introduction . . . 4
2.2 Mathematical Model . . . 9
2.2.1 Incompressible Navier-Stokes Equations . . . 9
2.2.2 General Projection Method . . . 10
2.2.3 Properties of the Projection Operator . . . 10
2.2.4 Boundary Conditions . . . 11
2.2.5 Pressure Reconstruction . . . 12
2.3 Node-based Projection Operator on Cartesian Grids . . . 13
2.3.1 Definitions and Key Observations . . . 13
2.3.2 Uniform Cartesian Grids . . . 15
2.3.3 Extension to Adaptive Grids . . . 17
2.3.4 Computational Verification . . . 24
2.4 Nodal Navier-Stokes Solver . . . 27
2.4.1 Overview of the Time-Stepping Algorithm . . . 27
2.4.2 Verification - Analytic Vortex . . . 33
2.5 Validation Examples . . . 35
2.5.1 Driven Cavity . . . 36
2.5.2 Oscillating Cylinder . . . 37
2.5.3 Von Kármán Vortex Street . . . 40
2.5.4 Flow Past Beginnings . . . 45

2.6 Conclusions . . . 46

Chapter 3 Sharp Collocated Projection Method for Immiscible Two-Phase Flows 48
3.1 Introduction . . . 48
3.2 Background . . . 50
3.2.1 Governing Equations . . . 50
3.2.2 Interface Representation . . . 51
3.2.3 General Two-Phase Projection Method . . . 52
3.2.4 Quad/Octree Grids . . . 58
3.3 Numerical Discretizations . . . 60
3.3.1 Temporal Discretizations . . . 61
3.3.2 Spatial Discretizations . . . 64
3.4 Results . . . 71
3.4.1 Verification Examples . . . 71
3.4.2 Validation Examples . . . 77
3.5 Conclusions . . . 88

Chapter 4 Characteristic Bending for Incompressible Flows . . . 93
4.1 Introduction . . . 93
4.2 Preliminaries . . . 97
4.2.1 Governing Equations . . . 97
4.2.2 Semi-Lagrangian Method . . . 98
4.2.3 Reference Map Advection . . . 99
4.2.4 Volume Preserving Correction . . . 101
4.3 Characteristic Bending . . . 105
4.3.1 Overall Method . . . 105
4.3.2 Advection Step . . . 109
4.3.3 Correction Step . . . 110
4.3.4 Numerical Implementation . . . 111
4.4 Computational Results . . . 111
4.4.1 Advection Examples . . . 111
4.4.2 Incompressible Multiphase Navier-Stokes . . . 132
4.5 Conclusions . . . 138

Chapter 5 Conclusions . . . 142

Bibliography . . . 144

LIST OF FIGURES

Figure 2.1: Staggered and Collocated Data Layouts 5
Figure 2.2: Computational Domain in Two Dimensions 9
Figure 2.3: Discrete Divergence on Collocated and Staggered Grids 14
Figure 2.4: Spectrum of the Projection Operator 15
Figure 2.5: Ghost-Node Construction on Quadtree Grids 18
Figure 2.6: Relations Between Staggered and Nodal Operators 19
Figure 2.7: Randomized Non-Graded Quadtree 26
Figure 2.8: Projection Operator Accuracy 26
Figure 2.9: Stability of the Projection Operator 28
Figure 2.10: Collocated Projection Method for Navier-Stokes 29
Figure 2.11: Analytic Vortex Convergence 34
Figure 2.12: Driven Cavity Validation 36
Figure 2.13: Oscillating Cylinder Validation 38
Figure 2.14: Time Evolution of the Oscillating Cylinder 39
Figure 2.15: Drag and Lift Validation 41
Figure 2.16: Vorticity of the von Kármán Vortex 42
Figure 2.17: Drag and Lift Results with Improved Reconstruction 43
Figure 2.18: Von Kármán Vorticity with Improved Reconstruction 43
Figure 2.19: Three-dimensional von Kármán Vorticity 44
Figure 2.20: Beginnings Schematic 45
Figure 2.21: Beginnings Visualization 46

Figure 3.1: Problem schematic in 3D 51
Figure 3.2: Multiphase Projection Algorithm 53
Figure 3.3: Hybrid Discretization Schematic 67
Figure 3.4: Adaptive Grid for the Multiphase Analytic Vortex 72
Figure 3.5: Convergence of the Projection Operator 73
Figure 3.6: Parasitic Currents Interface and Velocity Convergence 76
Figure 3.7: Time Evolution of the Oscillating Bubble Radius 78
Figure 3.8: Velocity Profile of the Oscillating Bubble 79
Figure 3.9: Bhaga-Weber Example Final Profiles 81
Figure 3.10: Case e Rising Bubble Evolution 82
Figure 3.11: Case h Rising Bubble Evolution 82
Figure 3.12: Case f Rising Bubble Evolution 84
Figure 3.13: Case f Rising Bubble Rupture 85
Figure 3.14: Case f Rising Bubble Rupture - Background Grid 85

Figure 3.15: Case f Rising Bubble Rupture - Post Rupture 86
Figure 3.16: Case f Rising Bubble Rupture - Late Time Evolution 86
Figure 3.17: Multiple Rising Bubbles 87
Figure 3.18: Rising Bubbles in an Inverted Funnel 89
Figure 3.19: Rising Bubbles in an Inverted Funnel - Zoom 90
Figure 3.20: Rising Bubbles in an Inverted Funnel - Velocity Profiles 91
Figure 3.21: Rising Bubbles in an Inverted Funnel - Volume Loss 92

Figure 4.1: Semi-Lagrangian Schematic 98
Figure 4.2: Construction of the Reference Map 101
Figure 4.3: Reference Map Schematic 102
Figure 4.4: Relation Between Incompressibility and Volume Preservation 103
Figure 4.5: Basic Characteristic Bending Algorithm 106
Figure 4.6: Reference Map Characteristic Bending Algorithm 107
Figure 4.7: Comparison of semi-Lagrangian and Characteristic Bending Evolution 108
Figure 4.8: Comparison of Reference Map Construction Schemes 109
Figure 4.9: Solution Error - Gaussian Advection 114
Figure 4.10: Mass Loss - Gaussian Advection 115
Figure 4.11: Slotted Circle on a Quadtree Grid 116
Figure 4.12: Solution Error - Slotted Circle 119
Figure 4.13: Volume Loss - Slotted Circle 120
Figure 4.14: Effect of Reinitialization for the Slotted Circle 121
Figure 4.15: Interface Evolution Slotted Circle 122
Figure 4.16: Evolution of the Slotted Sphere 123
Figure 4.17: Comparison of Reinitialization Schemes 125
Figure 4.18: Solution Error - Single Vortex 127
Figure 4.19: Volume Loss - Single Vortex 128
Figure 4.20: Evolution of the Deformational Flow in 3D 129
Figure 4.21: Volume Loss - Deformational Flow 3D 130
Figure 4.22: Solution and Divergence Error - Incompressible Euler 131
Figure 4.23: Error Comparison Multiphase Analytic Vortex 134
Figure 4.24: Comparison of Methods for Rising Bubble Case e 136
Figure 4.25: Convergence of Methods for Rising Bubble Case e 137
Figure 4.26: Volume Loss for Rising Bubble Case a 138
Figure 4.27: Evolution of Rising Bubble Case f 139

LIST OF TABLES

Table 2.1: Projection Operator Convergence 26
Table 2.2: Analytic Vortex Convergence 34
Table 2.3: Impact of Successive Projections on Accuracy 35
Table 2.4: Semi-Lagrangian Convergence Comparison 35
Table 2.5: Drag and Lift Comparisons 40
Table 2.6: Drag Result Comparison 42

Table 3.1: Coupled Jump Solver Convergence 2D 69
Table 3.2: Coupled Jump Solver Convergence 3D 70
Table 3.3: Convergence for the Multiphase Analytic Vortex 74
Table 3.4: Convergence of the Parasitic Currents Example 76
Table 3.5: Oscillating Bubble Period of Oscillation 78
Table 3.6: Bhaga-Weber Parameters - Multiphase Flow 79
Table 3.7: Summary of Bhaga-Weber Examples 83

Table 4.1: List of Method Acronyms 113
Table 4.2: Bhaga-Weber Parameters - Characteristic Bending 135

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Maxime Theillard, for his guidance, patience, and perspective throughout my doctoral work. He played a central role in helping me transition from thinking like an engineer to communicating and working as an applied mathematician, and in doing so helped me feel that I truly belonged in the field. His mentorship emphasized mindset and clarity of thought over labels, and under his supervision I had the freedom to explore, get stuck, and ultimately find my own path forward. This experience shaped both this dissertation and my growth as an independent researcher. Just as importantly, he modeled a healthy approach to research and life. Work hard when it mattered, and step away when family and other priorities deserved attention.

I am also sincerely thankful to my new advisor, Shilpa Khatri, who was a steady voice of reason and consistently took the time to understand the details. She was always someone I could bounce ideas off of, and she was quick to help clarify subtle analysis techniques that I was unfamiliar with. Later, when she stepped in as my advisor, she became an important anchor as I finished my doctoral work. While there was limited time for her to shape my overall research direction, I deeply appreciate her guidance on how to present my work clearly and effectively to a broad audience.

I am particularly thankful to Changho Kim, who went far above and beyond to provide guidance, encouragement, and critical feedback that made me a better student, researcher, and person. He did not have to invest the time and effort that he did, and I am deeply grateful that he chose to do so. Changho helped me identify my first internship opportunity at Berkeley Lab, supported me as a GSR, and set clear expectations for the level of work required to pursue a postdoctoral position at the lab. I would not be where I am today without his support.

I am deeply grateful to Ann Almgren, who played an important mentoring role throughout my time at Berkeley Lab and beyond. She taught me the value of building on robust, high-performance, and well-documented code bases rather than reinventing tools from scratch, and emphasized the importance of being clear and direct. She also showed me why it is always good practice to write the paper alongside the work itself. Most importantly, she demonstrated why Berkeley Lab is such a special place to do science.

I am also grateful to Rob Saye and Pierson Guthrey for their mentorship during and after my time at the national labs. Rob showed me what high-order computational research looks like in a national lab setting, and why it is sometimes better to build tools from scratch. He pushed me to focus on details, to work through problems independently, and to take the time to do things correctly and rigorously. Pierson demonstrated that mission-driven research can be both impactful and enjoyable, emphasized the importance of writing code that works in production, and introduced me to tensor network methods. His guidance

helped me develop a skill set that prepared me well for postdoctoral research. Together, they showed me that serious scientific computing is best enjoyed with C++.

I would also like to thank François Blanchette and Arnold Kim for their guidance, support, and critical feedback throughout my doctoral studies. Both are outstanding teachers, and their insights and thoughtful questions helped strengthen my understanding of applied mathematics and improve my work.

I would also like to thank my friends and colleagues. Adam and Scott took the time to work through the collocated projection method with me, helped me make sense of what the mathematicians were saying in the early stages of the program, and patiently reviewed countless drafts and edits along the way. Alex, Matteo, and Hardeep became close friends throughout my time in Merced, offering support when I needed it most and giving me the space to vent through the frustrations of graduate school. I did not realize how much I needed that support at the time, but I am deeply grateful that they were there.

Finally, I thank my family, who made the greatest sacrifices so that I could pursue this work. Leaving a stable career brought uncertainty, but it also gave me the opportunity to be present with my daughters. I was able to watch Remi and Lila grow, and be there for the little moments that are easy to miss. Along the way, I may have forced them to learn more about projection methods than any children their age ever should. If they grow up to love mathematics, I will take full credit. If they want nothing to do with it, that is likely on me as well.

To my wife, Sarah, I cannot fully express how much your love, patience, and support have meant to me. This journey would not have been possible without you, even during the long stretches when my contribution was simply sitting and thinking in different chairs. I could not have done this without you.

This material is based on work supported by the National Science Foundation under Grant No. DMS-1840265 and by the UC Merced Graduate Dean's Dissertation Fellowship.

Matthew Blomquist
mblomquist@ucmerced.edu

Education:

2026	University of California, Merced, Ph.D. in Applied Mathematics
2018	California State University Northridge, M.S. in Mechanical Engineering
2011	University of California, Riverside, B.S. in Mechanical Engineering

Employment:

2021–Present	University of California, Merced, Graduate Student Researcher
Summer 2024	Lawrence Livermore National Laboratory, Graduate Student Intern
Summer 2023	Lawrence Berkeley National Laboratory, Graduate Student Researcher
Summer 2022	Lawrence Berkeley National Laboratory, Graduate Student Researcher
2011–2021	ITT Aerospace Controls, Senior Aerospace Engineer

Publications:

1. Matthew Blomquist, Stéphane Gaudreault, Maxime Theillard,"Characteristic Bending in Incompressible Flows". In: *arXiv:2512.06455, Under review (2025)*

2. Adam L. Binswanger, Matthew Blomquist, Scott R. West, Shilpa Khatri, Maxime Theillard, "Sharp Collocated Projection Method for Immiscible Two-Phase Flows". In: *arXiv:2508.11107, Under review (2025)*

3. Matthew Blomquist, Scott R. West, Adam L. Binswanger, Maxime Theillard, "Stable nodal projection method on octree grids". In: *Journal of Computational Physics 499* (2024), p. 112695

4. Matthew Blomquist, Abhijit Mukherjee, "Performance Improvements of Krylov Subspace Methods in Numerical Heat Transfer and Fluid Flow Simulations". In: *ASME International Mechanical Engineering Congress and Exposition* (2019), Vol. 59452, V008T09A061

5. Matthew Blomquist, Arman Khalighi, Abhijit Mukherjee, "Numerical Study of Bubble Instability During Microchannel Flow Boiling". In: *International Conference on Nanochannels, Microchannels, and Minichannels* (2015), Vol. 56871, V001T07A015

6. Arman Khalighi, Matthew Blomquist, Abhijit Mukherjee, "Numerical Simulation of Bubble Growth During Nanofluid Flow Boiling in a Microchannel". In: *International Conference on Nanochannels, Microchannels, and Minichannels* (2014), Vol. 46278, V001T03A009

PUBLICATIONS AND CONTRIBUTIONS

The research presented in this dissertation is adapted from published and submitted manuscripts that together form a unified body of work. Below, I identify the chapter associated with each manuscript and summarize my contributions using the CRediT authorship taxonomy.

Chapter 2: Stable Nodal Projection Method on Octree Grids
Based on:
Matthew Blomquist, Scott R. West, Adam L. Binswanger, Maxime Theillard, "Stable nodal projection method on octree grids." *Journal of Computational Physics 499* (2024), p. 112695.

CRediT Contributions:
Formal analysis, Investigation, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing.

Chapter 3: Sharp Collocated Projection Method for Immiscible Two-Phase Flows
Based on:
Adam L. Binswanger, Matthew Blomquist, Scott R. West, Shilpa Khatri, Maxime Theillard, "Sharp Collocated Projection Method for Immiscible Two-Phase Flows." *Under review (2026).*

CRediT Contributions:
Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing.

Chapter 4: Characteristic Bending for Incompressible Flows
Based on:
Matthew Blomquist, Stéphane Gaudreault, Maxime Theillard, "Characteristic Bending in Incompressible Flows." *Under review (2026).*

CRediT Contributions:
Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing.

ABSTRACT OF THE DISSERTATION

Characteristic Bending and Structure-Preserving Methods for Incompressible Flow

by Matthew Blomquist
Doctor of Philosophy in Applied Mathematics

University of California Merced, 2026

Committee Chair: Shilpa Khatri

This dissertation develops a family of projection-based numerical methods for incompressible flow on adaptive, collocated grids. Here, projection methods act as the primary computational mechanism through which numerical stability is enforced, physical features are represented sharply, and the geometric structure of incompressible flow is respected. In a broader sense, these methods constitute a conceptual framework for modeling incompressible flows, enabling the development of more accurate, flexible, and physically consistent simulations of complex fluid phenomena.

Chapter 2 develops a stable collocated adaptive mesh framework for the incompressible Navier-Stokes equations. While collocating variables simplifies algorithmic design and data structures, it disrupts the discrete representation of analytical operators that is typically automatic in staggered grid formulations. This challenge is addressed through the design of a stable nodal projection method tailored to node-based adaptive grids, demonstrating that collocated formulations are stable and can achieve high-order accuracy.

Chapter 3 extends this collocated projection framework to immiscible two-phase incompressible flows. Interfacial jump conditions are incorporated directly into the projection operator through a hybrid finite difference–finite volume formulation, enabling a sharp and stable treatment of interfacial stresses. Numerical results demonstrate accurate interface dynamics and strong volume conservation for complex two- and three-dimensional flows.

Chapter 4 introduces the characteristic bending method, a structure-preserving advection scheme that applies a volume-preserving projection in a Lagrangian frame to attenuate compressible modes. This projection can be viewed as bending characteristics toward the divergence-free space, using local information from the reference map to construct a global correction. The resulting method yields a consistent representation of incompressible advection and integrates naturally into the collocated framework of the previous chapters.

Together, these developments establish a cohesive and extensible, projection-based framework for enforcing incompressibility on adaptive, collocated grids, unifying stability, sharp feature representation, and structure preservation across single-phase fluid flow, multiphase fluid flow, and general advection problems.

Chapter 1

Introduction

Incompressible fluid flows are ubiquitous in science and engineering applications and lie at the heart of numerous research questions, from understanding the dynamics of active fluids [55] to designing hydrofoils [180] to modeling the retention of rising oil droplets in the ocean [107]. Accurately capturing these phenomena requires computational tools capable of resolving complex geometries, multiple spatial and temporal scales, and strongly coupled physics. Despite decades of advances in computational fluid dynamics (CFD), the development of robust and extensible numerical methods remains a challenge, particularly when adaptive mesh refinement (AMR) and complex interfaces are involved.

This dissertation addresses these challenges by developing a family of numerical methods for incompressible flows using adaptive quadtree and octree grids, where all of the variables are collocated at the nodes [20]. The overarching goal is to provide a unified framework for solving the incompressible Navier-Stokes equations in two and three dimensions that can easily be modified or extended for scientific and engineering applications with arbitrary geometries, coupled physics, or multiphase flows. The work presented herein showcases the evolution of this framework from single phase applications, to multiphase flows, and finally to a structure-preserving method that filters spurious compressible modes from incompressible fields. Across these settings, projection methods are not merely viewed as an algorithmic step, but rather as a unifying mechanism for enforcing physical and geometric constraints in both Eulerian and Lagrangian formulations.

At the foundation of each of the methods developed herein is the idea of using a projection to enforce the physical or geometric constraints inherent to incompressible flows. In the context of the incompressible Navier-Stokes equations, projection methods were originally introduced by Chorin [35] and leverage the Helmholtz-Hodge decomposition to build a two-step numerical scheme for evolving the fluid velocity. In the

first step, an intermediate velocity field is computed without considering pressure and, in the second step, this intermediate velocity field is projected onto the divergence-free subspace. When performed using a uniform staggered grid, such as the Marker-and-Cell (MAC) layout [69], the projection method preserves the analytic properties of the gradient, divergence, and Laplacian operators discretely, resulting in a numerical velocity field that remains incompressible to machine precision.

Extending standard projection methods to adaptive grids, particularly tree-based grids, introduces significant complexity. Staggered layouts require multiple independent data structures, specialized interpolation schemes, and often unique discretization stencils for each velocity component, all of which can make implementation cumbersome and limit extensibility. In contrast, a collocated variable arrangement stores all quantities at the same spatial locations (in this case the nodes), provides a simpler and more flexible framework that is well-suited for tree-based adaptive mesh refinement. The cost of this simplicity is the loss of the exact analytical relationships among the discrete gradient, divergence, and Laplacian operators. This leads to an approximate projection, in which the numerical velocity field is only approximately divergence-free. Such approximations can introduce instability, spurious compressible modes, and a loss of accuracy in the solution of the Navier-Stokes equations.

Chapter 2 addresses these challenges through the development of a Stable Nodal Projection Method on Octree Grids [20]. This chapter introduces the collocated adaptive mesh framework that serves as the computational backbone for the remainder of this dissertation. It details the construction of the spatial differential operators at the core of the method, provides analytic justification for the stability of the nodal projection, and validates the solver through canonical verification and benchmark problems in both two and three dimensions. The chapter concludes with a demonstration of a moderate-Reynolds number flow past the University of California, Merced *Beginnings* sculpture, showcasing how arbitrary computer-aided design (CAD) data can be integrated within this framework to simulate real-world flows. This chapter is based on work originally published in the Journal of Computational Physics [20], with minor revisions for clarity and continuity.

Building on this collocated projection framework, Chapter 3 extends the method to develop a Sharp Collocated Projection Method for Immiscible Two-Phase Flows [17]. Multiphase systems introduce additional complexity through discontinuities in density and viscosity, as well as the need to track and evolve the interface between phases, while accurately imposing interfacial stress jump conditions arising from surface tension. These challenges are particularly demanding when the interface is treated in a sharp manner, where discontinuities are enforced directly at the interface rather than diffused over a transition region. The projection approach presented herein addresses these issues by

coupling the velocity components through a hybrid finite-volume/finite-difference formulation that treats the full stress tensor monolithically. This eliminates the iterative velocity-correction schemes required in previous formulations while retaining the simplicity and generality of the collocated adaptive framework introduced in Chapter 2. Verification and validation examples, including analytic tests and benchmarks against experimental data, demonstrate the accuracy of the method, its ability to capture highly deforming interfaces, and its robustness in simulating interactions between rising bubbles and solid obstructions. This chapter is based on work currently under review [17].

While Chapters 2 and 3 focus on enforcing incompressibility through projection in the traditional context of the Navier-Stokes equations, Chapter 4 extends this idea to general advection under incompressible velocity fields [18]. Numerical advection schemes are subject to several sources of error, including interpolation and time-integration inaccuracies, as well as error in the advecting velocity field itself that may arise from the use of approximate projections. These errors can lead to numerical instabilities, loss of geometric structure in the advected fields, and, in the context of multiphase flows, notable losses in the mass of bubble or droplets. Chapter 4 introduces a novel structure-preserving method, Characteristic Bending, to address these challenges. The key insight of this method is to reinterpret the classical semi-Lagrangian advection as the construction of a reference map, allowing us to leverage a volume-preserving projection to bend the characteristics of the velocity field toward the divergence-free space. The resulting scheme systematically filters compressible modes from the advecting velocity and provides a natural mechanism for correcting errors introduced by approximate projection methods. This work serves as a natural capstone to the dissertation, demonstrating how projection schemes can be used not only to enforce incompressibility, but also to preserve the geometric structure of incompressible flows. This chapter is based on work currently under review [18].

Together, the methods developed herein form a cohesive framework for simulating incompressible flows on adaptive, collocated grids. We begin by relaxing the traditional projection approach for incompressible flows to create a stable numerical method for solving the Navier-Stokes equations on adaptive collocated grids that prioritizes simplicity in implementation. We then extend this method to demonstrate how the framework provides a natural foundation for incorporating additional physical complexity, such as interfacial dynamics in multiphase systems. Finally, we show how errors introduced by approximate projection methods can be corrected using the Characteristic Bending approach, which applies the projection concept in a geometric setting to preserve the structure of incompressible flow. Collectively, these methods illustrate how projections can be used not only as a numerical tool for enforcing constraints, but also as a broader philosophy for developing accurate and extensible solvers for complex fluid systems.

Chapter 2

Stable Nodal Projection Method on Octree Grids

2.1 Introduction

Incompressible fluid flows are ubiquitous in science and engineering applications and lie at the heart of numerous research questions. Developing control strategies to minimize drag [52], understanding arterial wall deformation in the human heart [77], and even optimizing the energy consumption of automotive spray painting operations [148] all require a detailed understanding of incompressible flows. For the majority of these phenomena, analytical solutions do not exist, and experimental approaches can be difficult and costly to create. Numerical simulations are the natural choice for studying these problems. Still, despite decades of computational advancement, their development remains challenging, especially when irregular geometries, adaptive grids, or complex boundary conditions are involved. For this reason, it is essential to develop computational fluid dynamics tools that are accessible and straightforward to implement, which can be achieved, for example, by minimizing the number of unique data structures and simplifying data access patterns.

The first step in developing a numerical simulation for incompressible flows is typically to discretize the fluid domain using a numerical mesh. This mesh can be represented as a structured set of elements or nodes, as in the Cartesian [69] and curvilinear [94] styles, or by an unstructured mesh [129]. Structured meshes are usually easy to generate and can sometimes yield additional accuracy (*e.g.*, supra-convergence for nodal finite differences), but special care is needed to handle irregular geometries. Unstructured meshes, conversely, can tessellate irregular geometries, but their generation can be computationally expensive

[118]. With problems that involve moving geometries or in the context of adaptive grids, unstructured grids often require that computational costs be paid at each time step. For that reason, it is often preferable to use structured meshes, such as non-graded quad/octrees, and develop the tools to handle irregular geometries. This is the approach we take herein.

The primary challenge with numerically solving the incompressible Navier-Stokes equations is related to the coupling between the mass and momentum equations, which manifests as an incompressibility constraint on the velocity field that must be satisfied at each time step. One method for enforcing this constraint is to solve a monolithic system of coupled equations for the velocity and pressure unknowns [141]. This approach can often be beneficial when additional physical constraints are coupled with an incompressible flow solver, such as in two-phase flows [1, 31] and fluid-structure interaction problems [60, 79, 32, 161]. However, the monolithic systems may not be diagonally dominant and thus require the use of expensive solvers (*e.g.*, GMRES). For this reason, most computational frameworks for solving the incompressible Navier-Stokes equations use splitting methods.

The projection method, pioneered by Chorin [35] and later proposed as a fractional step method by Temam [162, 163], decouples the momentum equation and the incompressibility condition by leveraging the Helmholtz-Hodge decomposition. This creates a two-step time-stepping procedure for solving the Navier-Stokes equations, where an intermediate velocity field is first computed with the pressure omitted. Then, the intermediate velocity field is projected onto the divergence-free subspace to recover the incompressible velocity at

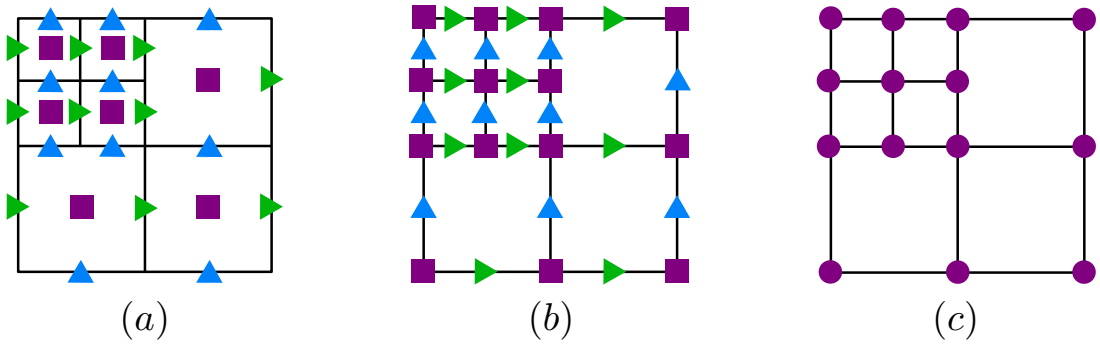


Figure 2.1: Data layouts - (a) Within the Marker and Cell representation the pressure components (■) are stored at the cell centers, the x-velocity (▶) are stored at the vertical faces, and the y-velocity components (▲) are stored at the horizontal faces. (b) In [66], Gomez *et al.* proposed a different staggered storage, where the pressure components are now stored at the nodes, and the velocity components are stored along the edges. (c) In our approach, all quantities (●) are collocated at the nodes.

the new time step. In this procedure, the pressure is never directly computed but can be reconstructed from the curl-free component of the intermediate velocity field (see *e.g.*, 2.2.5). The decoupling of the momentum equation and the incompressibility condition via this projection method introduces temporal splitting errors that manifest on the boundary condition and can limit the order of accuracy of numerical methods in both space and time. However, the development of high-order projection methods has been an active area of research for the last three decades, and a number of standard approaches exist (see [124, 87, 27, 67, 167, 106]).

The overall stability of the projection method relies on the stability of each step. In the first step, the computation of the intermediate velocity field requires the solution of an advection-diffusion problem. The stability of this step is dictated by the properties of the temporal integration scheme and is thus easily assessed. The stability of the projection step, however, is linked to the spatial discretization of the gradient, divergence, and Laplacian operators. The natural way to ensure that the projection step is stable is to preserve the analytical properties of these operators and impose the divergence-free constraint at the discrete level. Unfortunately, this creates complications when using a collocated arrangement. For example, the discrete Laplacian is no longer a composition of the discrete gradient and divergence when using collocated variables with standard central difference formulas on Cartesian grids. These complications can be overcome by using interpolation procedures (*e.g.*, see [115]), but these procedures can themselves become challenging in the presence of irregular boundaries or adaptive grids.

An alternate approach is to use a staggered layout, such as the Marker and Cell (MAC) representation introduced by Harlow and Welch [69], and illustrated in Figure 2.1. In this staggered arrangement, pressure data is located at the cell centers, and the velocities are located at the cell faces. In this context, the analytical properties of the operators are preserved, and the projection step enforces the divergence-free constraint with an orthogonal projection of the intermediate velocity field onto the divergence-free space. For this reason, the MAC layout has long been recognized as an ideal choice for solving the incompressible Navier-Stokes equations [65, 90, 177].

A different approach is to relax the requirement that the operators are preserved exactly at the discrete level and approximate the projection step. This notion of using an approximate projection was first introduced by Almgren, Bell, and Szymczak [6] as a means of circumventing the numerical difficulties associated with exact discrete projections. In their approach, the discrete Laplacian operator is consistent, but not exactly the composition of the divergence and gradient, and the divergence of the resulting velocity field is only approximately zero up to second order in the mesh spacing. This approximate projection uses velocities collocated at the cell centers with pressure data stored at the nodes. This design choice provides a symmetric discretization and generates

a well-behaved linear system suitable for a multigrid solver. Additionally, the collocation of the velocities greatly simplifies the application of high-order upwind techniques and enables the overall time-stepping algorithm to achieve second-order convergence in both space and time. For a detailed analysis of approximate projection methods, we refer the reader to [5] and the references therein, which cover the use of approximate projections for a variety of incompressible flow problems and the combination of approximate projection methods with adaptive spatial and temporal meshes.

The inherent multiscale nature of the incompressible Navier-Stokes equations calls for adaptive mesh refinement (AMR) to optimize computation and memory overhead. Typically, only small localized regions of the domain need high grid resolution, for example, where high vorticity is present, such as in boundary layers or vortex cores. One of the earliest examples of using AMR is from Berger and Oliger [15], where finer grids were adaptively placed over a coarse grid covering the domain. In [4], Almgren *et al.* combined an approximate projection method with an adaptive refinement strategy to solve the three-dimensional variable density incompressible Navier-Stokes equations. This block-structured approach was comprised of a nested hierarchy of logically-rectangular girds with refinement in both space and time. This AMR approach has been extended to numerous applications, and we refer the interested reader to the survey of [44] and to [149] for more details. For a modern block-structured AMR framework, we refer the reader to [191, 192].

Tree-based approaches to AMR [142] are an alternative to block-structured AMR and the strategy used herein. They combine efficiency with simplicity by using recursive splitting schemes with non-overlapping regions. One of the earliest uses of graded octrees (a tree in which the size ratio between adjacent cells is at most two) for solving the incompressible Euler equations can be seen from Popinet in [133], where finite volume discretizations are used on collocated cell-based quantities. This work uses the same approach as Almgren *et al.* in [6] to approximate the Laplacian but also requires an approximation for the gradient operator. This leads to a nonuniform stencil and a non-symmetric system of linear equations for the pressure. In [104], Losasso *et al.* proposed a symmetric discretization of the Poisson equation on octrees for free surface flows. This discretization resulted in a symmetric linear system, solved using a standard preconditioned conjugate gradient method. Their approach was only first-order accurate in the case of a non-graded mesh but was later extended to second-order accuracy [103], and later employed in the context of single phase [10, 67, 46], multiphase [167], and free surface [48, 7] applications.

In [112], Min *et al.* developed a collocated projection method for the incompressible Navier–Stokes equations on non-graded adaptive grids. Their method utilized the Poisson solver of [114], which produced a non-symmetric but diagonally dominant linear system

where the gradients of the solution were also second-order accurate. To ensure stability, instead of using an approximate projection, the authors manually enforced the orthogonality property between the divergence-free velocity field and the gradient of the Hodge variable when computing the updated velocity field. While this orthogonalization procedure results in an exact projection, this method is only stable if the normal velocity is null along the boundary.

For a more versatile approach, Guittet *et al.* [67] developed a stable projection method for the incompressible Navier-Stokes equations on non-graded octree grids using a MAC layout. While this layout is a natural framework for a stable projection, the poor alignment between the pressure and velocity variables necessitated the use of complicated discretizations for the momentum equation and the projection step. The authors had to develop a Voronoi-based finite volume approach to treat the viscous terms implicitly and an expensive least squares interpolation was required to implement the semi-Lagrangian scheme for the advective terms. Though complicated discretizations were required, this framework was later extended to simulate active [165, 169] and interfacial [36, 167] flows.

An alternative staggered layout was proposed by Gomez *et al.* [66], in which the pressure was stored at the nodes and velocity stored along the edges (see Figure 2.1). This configuration of the staggered layout offers several advantages over the traditional MAC layout. The pressure gradient and velocity are naturally aligned, simplifying the construction of finite difference operators and granting higher accuracy. Unfortunately, this staggered layout, like the MAC configuration, requires multiple data structures and solvers.

The method we present here is entirely collocated at the nodes of an arbitrary octree. It is carefully designed to achieve stability while reproducing solid interfaces and accompanying boundary layers with high fidelity. The overall stability of our method relies on the properties of our projection operator, which we analyze by relating the staggered and collocated approaches. We use a second-order semi-Lagrangian Backward Difference Formula scheme to update the momentum equation, treating the advective term explicitly and the viscous one implicitly. Arbitrary interface and boundary conditions are treated in a sharp manner using a level-set representation and the hybrid Finite-Volume/Finite-Difference discretizations from [169].

This chapter is organized as follows. In Section 2.2, we begin by recalling the incompressible Navier-Stokes equations and the general projection method used to solve them numerically. In Section 2.3, we present the principal result of this work: our collocated projection operator on Cartesian grids. We prove its stability on uniform grids, construct a sufficient stability condition for adaptive grids, and provide numerical evidence of its stability for various boundary conditions and grid configurations. In Section 2.4, we integrate our projection operator into a complete solver for the

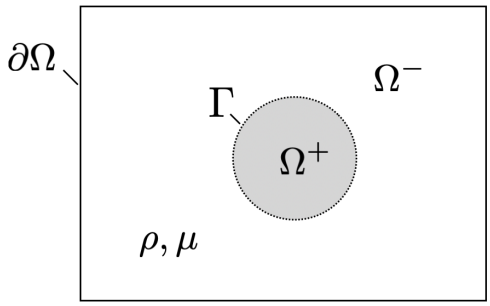


Figure 2.2: Computational domain shown in two dimensions. The fluid domain, Ω^-, is enclosed by the domain boundary, $\partial\Omega$, and the interface, Γ. Fluid properties, ρ and μ, are constant throughout the fluid domain. An arbitrary solid domain, Ω^+ is shown as a shaded region.

incompressible Navier-Stokes equations and observe overall second-order convergence. In Section 2.5, we illustrate the robustness and efficiency of our solver by using it to simulate several common validation problems of incompressible fluid flows in two and three spatial dimensions. We conclude in Section 2.6.

2.2 Mathematical Model

2.2.1 Incompressible Navier-Stokes Equations

We consider a fluid set in a computational domain $\Omega = \Omega^+ \cup \Omega^- \subset \mathbb{R}^{2,3}$, where Ω^- represents the fluid phase and Ω^+ represents all solid objects present in the fluid (see Figure 2.2), whose dynamics are modeled by the incompressible Navier-Stokes equations

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\Delta\mathbf{u} + \mathbf{f} \qquad \forall \mathbf{x} \in \Omega^-, \tag{2.1}$$

$$\nabla\cdot\mathbf{u} = 0 \qquad \forall \mathbf{x} \in \Omega^-, \tag{2.2}$$

where $\mathbf{u}$ is the fluid velocity, p is the pressure, ρ is the constant density, μ is the constant viscosity, and $\mathbf{f}$ are external forces such as the gravitational force. We denote the boundary of Ω^- by Γ and the boundary of the computational domain Ω by $\partial\Omega$.

2.2.2 General Projection Method

The classical projection method is a fractional-step scheme for solving the incompressible Navier-Stokes equations. In the first step, often referred to as the viscosity or advection step, we advance the velocity field $\mathbf{u}^n$ at time step t_n to an intermediate velocity field $\mathbf{u}^*$ by solving the momentum equation (2.1) where the pressure term is omitted. For example, using a standard first-order semi-explicit scheme, $\mathbf{u}^*$ is constructed as the solution of

$$\rho\left(\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} + \mathbf{u}^n \cdot \nabla \mathbf{u}^n\right) = \mu \Delta \mathbf{u}^* + \mathbf{f}. \tag{2.3}$$

Next, we project $\mathbf{u}^*$ onto the divergence-free space to enforce the incompressibility condition (2.2). To do so, we use the Helmholtz-Hodge decomposition to separate the intermediate velocity into curl-free and divergence-free components as follows:

$$\mathbf{u}^* = \mathbf{u}^{n+1} + \nabla \phi, \tag{2.4}$$

where $\mathbf{u}^{n+1}$ is the divergence-free velocity field at time t_{n+1} and ϕ is the Hodge variable. Taking the divergence of the above equation (2.4), we obtain a Poisson equation for the Hodge variable,

$$\Delta \phi = \nabla \cdot \mathbf{u}^*. \tag{2.5}$$

Using the appropriate boundary conditions, the above equation is solved to compute the Hodge variable, and the divergence-free velocity is recovered as

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \nabla \phi. \tag{2.6}$$

Note that equation (2.6) can be rewritten as an operator applied to $\mathbf{u}^*$,

$$\mathbf{u}^{n+1} = \left(\mathcal{I} - \nabla \Delta^{-1} \nabla \cdot\right) \mathbf{u}^*. \tag{2.7}$$

Thus, we define the generic projection operator P as,

$$P = \mathcal{I} - \nabla \Delta^{-1} \nabla \cdot . \tag{2.8}$$

2.2.3 Properties of the Projection Operator

The analytic operator P, defined by equation (2.8), is indeed a projection (*i.e.*, $P^2 = P$) as

$$P^2 = \left(\mathcal{I} - \nabla \Delta^{-1} \nabla \cdot\right)^2 = \mathcal{I} - 2\nabla \Delta^{-1} \nabla \cdot + \nabla \Delta^{-1} \nabla \cdot \nabla \left(\nabla \cdot \nabla\right)^{-1} \nabla \cdot, \tag{2.9}$$

$$= \mathcal{I} - \nabla \Delta^{-1} \nabla \cdot . \tag{2.10}$$

The projection property ensures that the projected field is exactly divergence-free and, thus, that the incompressibility condition is exactly satisfied. Moreover, since the gradient and divergence are the negative transpose of each other ($\nabla^T = -\nabla\cdot$), the projection is symmetric $P^T = P$, orthogonal *i.e.*,

$$\|\mathbf{u}^{n+1}\|^2 = \|P\mathbf{u}^*\|^2 = \|\mathbf{u}^*\|^2 - 2 < \mathbf{u}^*|\nabla\phi > + \|\nabla\phi\|^2, \tag{2.11}$$

$$= \|\mathbf{u}^*\|^2 + 2 < \nabla\cdot\mathbf{u}^*|\phi > + \|\nabla\phi\|^2, \tag{2.12}$$

$$= \|\mathbf{u}^*\|^2 + 2 < \nabla\cdot\nabla\phi|\phi > + \|\nabla\phi\|^2, \tag{2.13}$$

$$= \|\mathbf{u}^*\|^2 - \|\nabla\phi\|^2, \tag{2.14}$$

and therefore contracting,

$$\|P\mathbf{u}^*\| \leq \|\mathbf{u}^*\|. \tag{2.15}$$

In the discrete case, if the composition (*i.e.* $\Delta = \nabla\cdot\nabla$) and negative transpose properties are preserved, P remains contracting, and, provided that the temporal integrator is stable, the entire update from t_n to t_{n+1} is stable.

2.2.4 Boundary Conditions

We focus on single-phase flows around solid objects and explicitly restrict attention to the no-slip boundary condition at fluid–solid interfaces. Solid boundaries are represented implicitly using a level-set description and enforced sharply in the discretization, following [165]. In addition, standard inflow and outflow boundary conditions are applied at the outer boundaries of the computational domain.

The no-slip boundary condition is a Dirichlet boundary condition on the velocity, $\mathbf{u}|_\Gamma$, where $\mathbf{u}|_\Gamma$ is zero if the interface is static and equal to the velocity of the interface if it is moving. This, along with equation (2.4), implies that the boundary condition of the intermediate velocity field $\mathbf{u}^*$ is

$$\mathbf{u}^*|_\Gamma = \mathbf{u}|_\Gamma + \nabla\phi^{n+1}|_\Gamma. \tag{2.16}$$

When solving for the Hodge variable in the projection step, we prescribe the homogeneous Neumann boundary conditions, *i.e.*,

$$(\nabla\phi\cdot\mathbf{n})|_\Gamma = 0, \tag{2.17}$$

where $\mathbf{n}$ is the outward-facing normal vector to the boundary. Note that the value of ϕ is not computed when finding the intermediate velocity field $\mathbf{u}^*$. To keep the viscosity and projection decoupled, we can use the Hodge variable at the previous time step, ϕ^n, to serve as an initial guess to the correct boundary condition of $\mathbf{u}^*$ and iterate until we have

convergence in both $\mathbf{u}^*$ and ϕ. In [67, 46, 106], it was observed that using this initial guess led to very few iterations being needed to reach convergence. Other iterative corrections can be designed, (see for example [169]). The $\nabla\phi^{n+1}$ term in the above (2.16) should be seen as a splitting error. It can be shown to be first-order in time. Even in the absence of iterative correction, the presence of this term will only introduce converging errors on the boundary conditions [27].

Potential flux boundary conditions on the walls of the computational domain $\partial\Omega$ correspond to Neumann boundary conditions. This means that the boundary condition for the intermediate velocity field $\mathbf{u}^*$ is

$$(\nabla\mathbf{u}^* \cdot \mathbf{n})|_{\partial\Omega} = (\nabla\mathbf{u} \cdot \mathbf{n})|_{\partial\Omega} + (\nabla\nabla\phi \cdot \mathbf{n})_{\partial\Omega}. \tag{2.18}$$

The boundary condition for the Hodge variable is a Dirichlet boundary condition. Again the last term in the above equation should be seen as a splitting and converging ($\mathcal{O}(\Delta t)$) error. In this case, an iterative correction for this term would require differentiating the Hodge variable twice, which may produce noisy and inaccurate results. Therefore, this term is often disregarded. If boundary conditions are prescribed on the pressure directly, the corresponding boundary conditions on the Hodge variable are obtained from the pressure reconstruction formula.

2.2.5 Pressure Reconstruction

In our formulation of the projection method, pressure is never directly computed. If desired, we can compute it using (2.3) and (2.4) by asking that the discrete momentum equation must be satisfied for $\mathbf{u}^{n+1}$ and p, and obtaining

$$\rho\left(\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \mathbf{u}^n \cdot \nabla\mathbf{u}^n\right) = -\nabla p + \mu\Delta\mathbf{u}^{n+1}. \tag{2.19}$$

Since $\mathbf{u}^{n+1} = \mathbf{u}^* - \nabla\phi$ from the Helmholtz-Hodge decomposition

$$\rho\left(\frac{\mathbf{u}^* - \nabla\phi - \mathbf{u}^n}{\Delta t} + \mathbf{u}^n \cdot \nabla\mathbf{u}^n\right) = -\nabla p + \mu\Delta(\mathbf{u}^* - \nabla\phi), \tag{2.20}$$

and because $\mathbf{u}^*$ is defined as the solution of (2.3), this simplifies into

$$-\rho\frac{\nabla\phi}{\Delta t} = -\nabla p - \mu\Delta\nabla\phi, \tag{2.21}$$

or equivalently,

$$\nabla p = \frac{\rho}{\Delta t}\nabla\phi - \mu\nabla\Delta\phi, \tag{2.22}$$

and so, up to an additive constant, we arrive at our pressure reconstruction formula

$$p = \frac{\rho}{\Delta t}\phi - \mu\Delta\phi. \tag{2.23}$$

The pressure is reconstructed at time level t^{n+1}. We point out that using a different time integrator may alter the above formula (see *e.g.*, [67, 167]).

2.3 Node-based Projection Operator on Cartesian Grids

In this section, we introduce our collocated projection method, analyze its properties on periodic uniform and adaptive Cartesian grids, and verify the stability of the method numerically.

We find that the composition projection property is lost due to the collocation, and thus our operator is not a projection. However, we see that our operator can be iterated to recover the canonical projection onto the divergence-free space. We derive a technical sufficient condition for these iterations to converge and, in the periodic uniform case, are able to prove that our operator is, in fact, contracting.

The convergence analysis relies on the connection between the staggered and collocated frameworks. To this effect, we show that the collocated projection can be related to the staggered one through interpolation procedures and leverage important facts about the discrete projection operator on staggered grids. The interpolations are solely introduced for the purpose of proving the stability of the collocated projection; they do not appear in the numerical implementation.

2.3.1 Definitions and Key Observations

Our nodal projection operator $\mathcal{P}_N$ is defined from the nodal gradient $\mathcal{G}_N$, divergence $\mathcal{D}_N$ and Laplacian $\mathcal{L}_N$ as

$$\mathcal{P}_N = \mathcal{I} - \mathcal{G}_N\mathcal{L}_N^{-1}\mathcal{D}_N, \tag{2.24}$$

where $\mathcal{I}$ is the appropriate identity operator and $\mathcal{L}_N^{-1}$ is the operator that for any vector X, returns Y the solution of $\mathcal{L}_N Y = X$ with problem-dependent homogeneous boundary conditions [1]. Again, the nodal Laplacian is not the composition of the nodal divergence and nodal gradient (*i.e.*, $\mathcal{L}_N \neq \mathcal{G}_N\mathcal{D}_N$) and our nodal projection operator is not a true projection (*i.e.*, $\mathcal{P}_N^2 \neq \mathcal{P}_N$). This means that the projected velocity field is not guaranteed to be divergence-free (*i.e.*, $\mathcal{D}_N\mathcal{P}_N X \neq 0$). However, we can show that our projection

[1] If the boundary conditions are chosen so that the solution is not uniquely defined, we will enforce that the solution is zero at one specific location.

operator only preserves the incompressible modes of the velocity field. Therefore, if we iterate our projection operator and if the iterated operator converges, then it must converge to the canonical orthogonal projection.

To demonstrate this, we assume that $\mathcal{P}_N^k \to \mathcal{P}_N^\infty$ as $k \to \infty$, and remark that if λ_i are the eigenvalues of $\mathcal{P}_N$, the eigenvalues of $\mathcal{P}_N^\infty$ are $\lim_{k\to\infty} \lambda_i^k$, and since they are finite, they can only be 1 or 0. In addition, the eigenvectors of $\mathcal{P}_N^\infty$ corresponding to the eigenvalue 1 are the eigenvectors of $\mathcal{P}_N$ for the same eigenvalue. Therefore $\mathcal{P}_N^\infty$ is the projection on the eigenspace of $\mathcal{P}_N$ corresponding to the eigenvalue 1. Clearly, any divergence-free modes X belong to this eigenspace since

$$\mathcal{P}_N^\infty X = \lim_{k\to\infty} \mathcal{P}_N^k X = X. \tag{2.25}$$

Now, if X is an eigenvector associated to the eigenvalue 1, $\mathcal{P}_N X = X$, and so

$$\mathcal{G}_N \mathcal{L}_N^{-1} \mathcal{D}_N X = 0, \tag{2.26}$$

meaning that $\mathcal{L}_N^{-1}\mathcal{D}_N X$ must be constant. Assuming that a homogeneous Dirichlet boundary condition is enforced somewhere, this constant must be zero

$$\mathcal{L}_N^{-1} \mathcal{D}_N X = 0, \tag{2.27}$$

and thus $\mathcal{D}_N X = 0$, indicating that X is incompressible. The eigenspace for the eigenvalue 1 is the incompressible space, and therefore P_N^∞, if it exists, is the canonical projection on the incompressible space.

In the subsequent sections, we construct a sufficient condition for this limit to exist. However, we note that in practice, this limit may not be reached as only a small number of

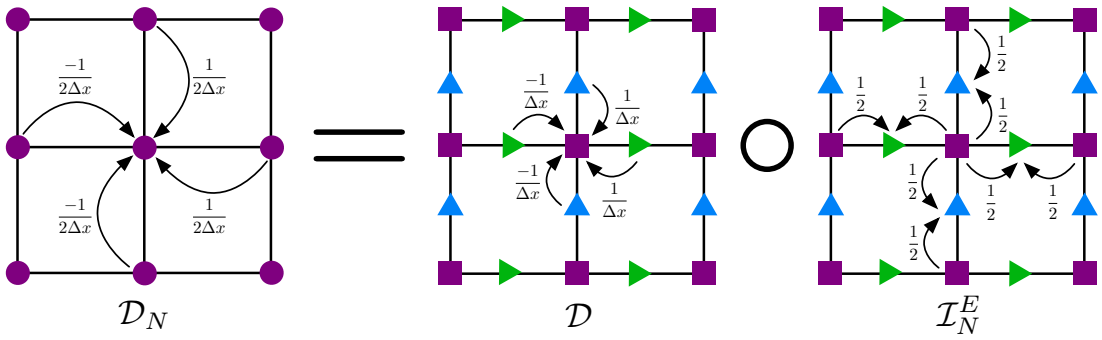


Figure 2.3: Connection between the discrete divergence on collocated ($\mathcal{D}_N$) and staggered grids ($\mathcal{D}$). The arrows represent the finite difference coefficients used in each contributing term.

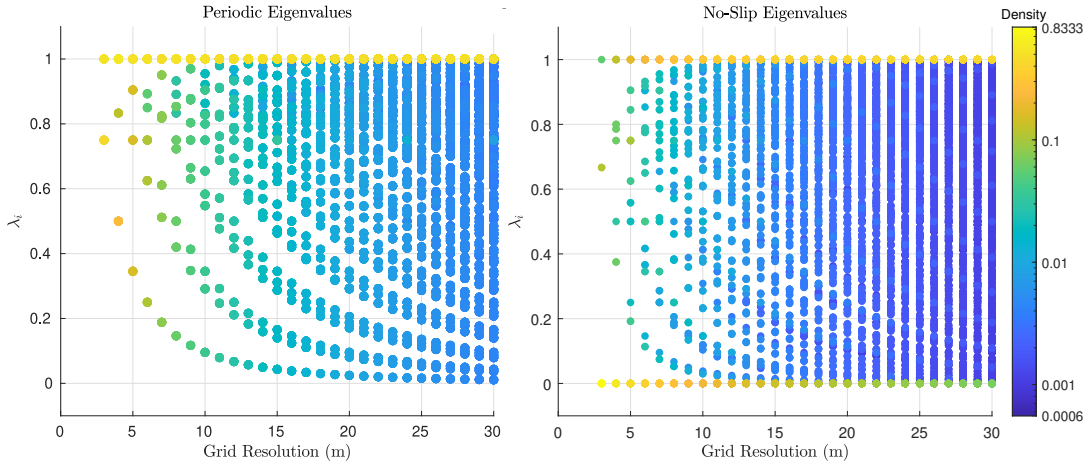


Figure 2.4: Spectrum of the projection operator with periodic boundary conditions (left) and no-slip boundary conditions (right) on a uniform grid. We observe that the eigenvalues are in the interval [0,1], which suggests that the operator is stable. The density of the eigenvalues for each grid resolution is shown.

iterations will be performed, and the advanced velocity field may not be exactly divergence-free. These deviations can be kept arbitrarily small by adjusting the stopping criteria, and in practice, a few iterations are needed to achieve a reasonably small error tolerance. In fact, a single iteration may be enough to obtain an overall second-order solution (see Section 2.4.2).

2.3.2 Uniform Cartesian Grids

Periodic Domains

In this case, we define $\mathcal{G}_N$, $\mathcal{L}_N$, $\mathcal{D}_N$ to be the standard periodic second-order finite difference gradient, Laplacian, and divergence. As Figure 2.3 illustrates, the nodal divergence operator $\mathcal{D}_N$ can be seen as the composition of the staggered discrete divergence $\mathcal{D}$ with the linear interpolation operator from the set of nodes N to the set of edges E, $\mathcal{I}_N^E : N \to E$

$$\mathcal{D}_N = \mathcal{D}\mathcal{I}_N^E. \tag{2.28}$$

Similarly, the discrete nodal gradient $\mathcal{G}_N$ is related to the staggered operator and the interpolation from the edges to the nodes $\mathcal{I}_E^N$ as

$$\mathcal{G}_N = \mathcal{I}_E^N \mathcal{G}. \tag{2.29}$$

The staggered operators satisfy the negative transpose property

$$\mathcal{D}^T = -\mathcal{G}, \tag{2.30}$$

and the two interpolations are transpose of each other

$$(\mathcal{I}_E^N)^T = \mathcal{I}_N^E. \tag{2.31}$$

Finally, we observe that the nodal Laplacian $\mathcal{L}_N$ is identical to the staggered one $\mathcal{L}$

$$\mathcal{L}_N = \mathcal{L}. \tag{2.32}$$

Using these observations, we can rewrite the projection operator (2.24) as

$$\mathcal{P}_N = \mathcal{I} - I_E^N \mathcal{G}\mathcal{L}^{-1}\mathcal{D}I_N^E. \tag{2.33}$$

All these ingredients allow us to follow the contraction's proof (see Section 2.2.3): the norm of the projected velocity is

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 - 2 < X|I_E^N \mathcal{G}\mathcal{L}^{-1}\mathcal{D}I_N^E X > + \|I_E^N \mathcal{G}\mathcal{L}^{-1}\mathcal{D}I_N^E X\|^2, \tag{2.34}$$

setting $\mathcal{L}^{-1}\mathcal{D}I_N^E X = \Phi$, and using the transpose properties of the divergence and interpolation we get

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 + 2 < \mathcal{D}I_N^E X|\Phi > + \|I_E^N \mathcal{G}\Phi\|^2, \tag{2.35}$$

which is identical to

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 + 2 < \mathcal{L}\mathcal{L}^{-1}\mathcal{D}I_N^E X|\Phi > + \|I_E^N \mathcal{G}\Phi\|^2, \tag{2.36}$$

giving us

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 + 2 < \mathcal{L}\Phi|\Phi > + \|I_E^N \mathcal{G}\Phi\|^2, \tag{2.37}$$

and since $\mathcal{L} = \mathcal{D}\mathcal{G}$ and $\mathcal{G} = -\mathcal{D}^T$

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 - 2 < \mathcal{G}\Phi|\mathcal{G}\Phi > + \|I_E^N \mathcal{G}\Phi\|^2, \tag{2.38}$$

we have

$$\|\mathcal{P}_N X\|^2 = \|X\|^2 - 2\|\mathcal{G}\Phi\|^2 + \|I_E^N \mathcal{G}\Phi\|^2, \tag{2.39}$$

and so

$$\|\mathcal{P}_N X\|^2 \leq \|X\|^2 - \left(2 - \|I_E^N\|^2\right) \|\mathcal{G}\Phi\|^2. \tag{2.40}$$

Thus, if $\|I_E^N\| < \sqrt{2}$, we have $\|\mathcal{P}_N X\| \leq \|X\|$ and $\mathcal{P}_N$ is contracting, and therefore converging. Here, the interpolations are averaging operators, so their L^∞-norms are less than or equal to one, and the projection is contracting.

In Figure 2.4, we display the spectrum of the projection operator, computed directly from the discrete numerical operator and for increasing grid resolution. Since the operator is contracting, we expect its spectrum to lie in the interval $[0, 1]$, and indeed it does. We color the eigenvalues by the density of the eigenvalue for that grid resolution. Note that for all grid resolutions, the eigenvalue with the highest density is the $\lambda = 1$ eigenvalue, which corresponds to an eigenvector with an incompressible mode.

No-slip Boundary Conditions

For a more realistic context, we turn our attention to no-slip boundary conditions. On rectangular domains, this means that we set the x-velocity $u = 0$ along the horizontal walls, the y-velocity $v = 0$ along the vertical walls, and the normal gradient of the Hodge variable to be zero on all walls. Currently, we do not have proof of the stability of this operator, as a general formulation of this operator with these boundary conditions, unlike in the periodic case, is not easily known. Instead, we reconstruct the numerical operator by projecting the canonical basis vectors of $\mathbb{R}^n$, where n is the dimension of the square matrix operator, with the boundary condition set beforehand. The resulting spectrum for increasing grid resolution is depicted in Figure 2.4, and is contained in $[0, 1]$. The addition of the boundary condition introduces an extra constraint on the velocity field, causing the projection to be even more restrictive and thus the admissible incompressible modes (*i.e.*, eigenvector associated to $\lambda = 1$) to be rarer. In addition, the no-slip boundary condition introduces a 0 eigenvalue to the operator, which corresponds to the canonical basis vectors that are non-zero only along the walls with the prescribed boundary condition.

2.3.3 Extension to Adaptive Grids

The negative transpose property is lost on adaptive grids, so the projected velocity norm cannot easily be bounded. Instead, we focus on the spectrum of the projection, and since we know that the incompressible space is the eigenspace associated with the eigenvalue 1 (see Section 2.3.1), we only need to find a sufficient condition for all eigenvalues associated to compressible eigenvectors to be strictly less than 1 in absolute value. In other words, a condition under which our projection is critically stable.

We start our study by formally defining all involved operators and proving the stability of the staggered projection. We then leverage this property to find a sufficient convergence condition. For the sake of clarity, this entire theoretical study is done on two-dimensional

periodic grids. The extension to three dimensions is tedious but straightforward.

Construction of the Discrete Differential Operators

To discretize our differential operators, we follow the work of Min and Gibou [114]. The central idea is to define ghost values at T-junctions (often called hanging nodes) to circumvent the lack of direct neighbors. For the construction of the gradient and Laplacian operators, we define the ghost values using standard Taylor analysis and derivative approximation. Referring to Figure 2.5, node n_0 does not have a direct neighbor to the right, so we introduce a ghost node n_r on the face delimited by nodes n_{r_t} and n_{r_b}. For any nodal quantity ϕ, sampled at the existing nodes, we calculate a third-order accurate ghost value ϕ_r using the information at n_0, its direct neighbors (*i.e.* n_t, n_b), and at the neighboring nodes n_{r_t} and n_{r_b} as

$$\phi_r = \frac{r_b\phi_{r_t} + r_t\phi_{r_b}}{r_t + r_b} - \frac{r_t r_b}{t+b}\left(\frac{\phi_t - \phi_0}{t} - \frac{\phi_0 - \phi_b}{b}\right). \tag{2.41}$$

As we will see in the following sections, the interpolation methodology used to construct the ghost nodes for the divergence operator is key to the stability of the nodal projection. We found that using the same third-order interpolation to construct the ghost

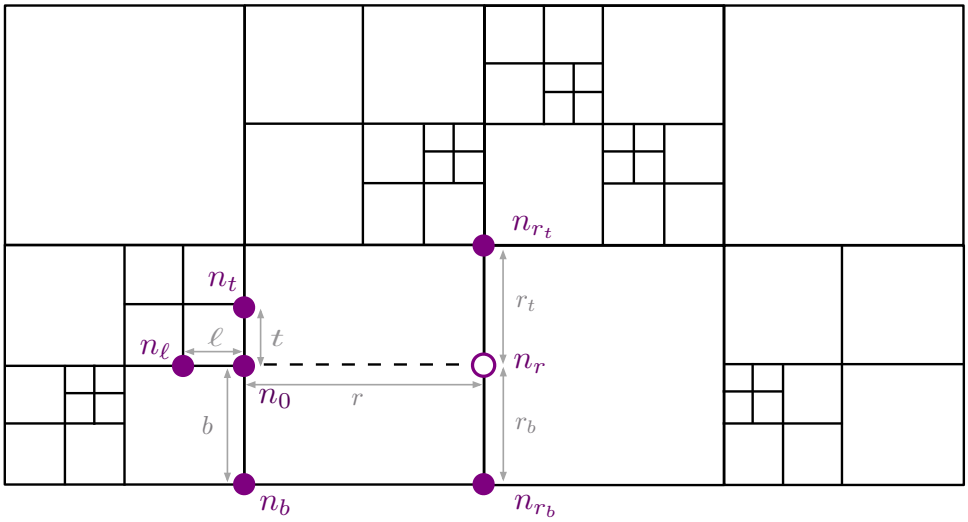


Figure 2.5: Finite difference discretization on quadtree grids. Here, node n_0 has no direct neighbor to the right, and thus a ghost node n_r (○) must be constructed using the existing neighboring nodes (●). Standard central discretizations can then be constructed using this ghosted neighborhood.

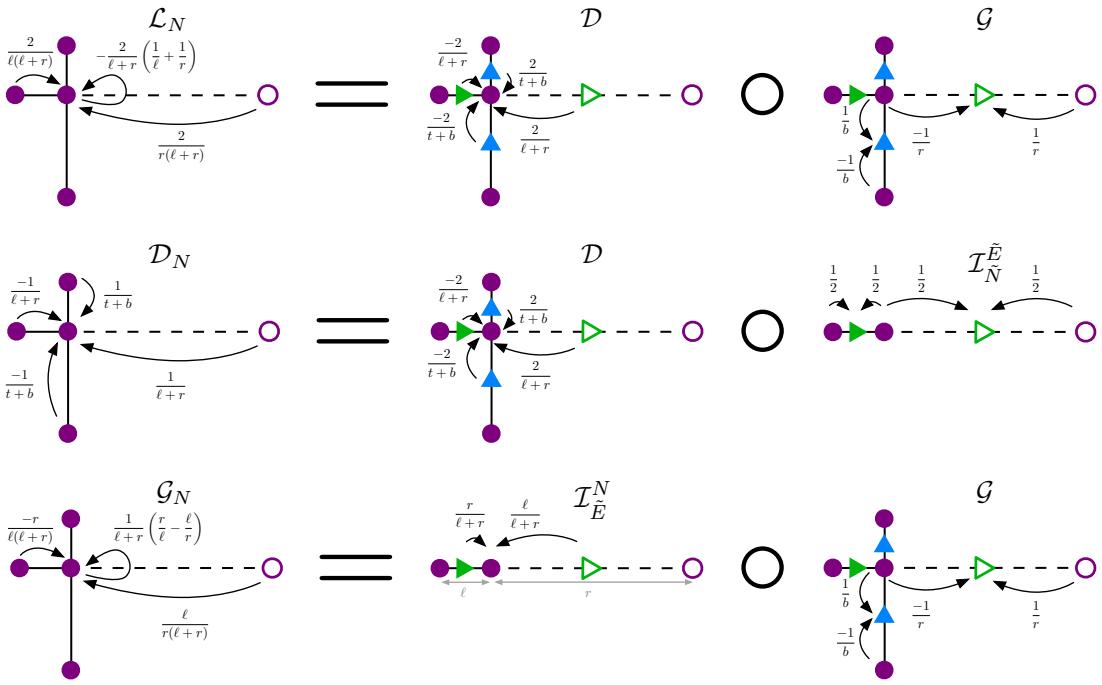


Figure 2.6: Connections between the staggered and collocated operators on periodic quadtree grids. The nodal Laplacian $\mathcal{L}_N$ is the composition of the staggered divergence $\mathcal{D}$ and gradient $\mathcal{G}$. The nodal divergence $\mathcal{D}_N$ and gradient $\mathcal{G}_N$ are related to their staggered counterpart through linear interpolation. Ghost variables are depicted as empty symbols (▷ and ○). As in Figure 2.5, arrows represent the finite difference coefficients. For the gradient and Laplacian, only the x-contribution is illustrated. The y-contribution is computed similarly.

nodes for the divergence operator, for example, is unstable. Therefore, we use a third-order quadratic interpolation scheme to define the ghost node values at T-junctions for the divergence operator. In Section 2.3.4, we show that this choice of interpolation yields a stable projection operator for a variety of boundary and interface conditions. We note that this is not the only interpolation scheme that produces a stable projection operator. In our exploration, we found that linear, ENO [70], and WENO [101] interpolation schemes are also numerically stable. These interpolations are computed as follows: if the position $\mathbf{x} = (x, y)$ belongs to the cell $\mathcal{C}$, which we assume to be $[0, 1]^2$ for

the sake of presentation, the interpolated value is

$$\phi(x,y) = \phi^{00}(1-x)(1-y) + \phi^{01}(1-x)y + \phi^{10}x(1-y) + \phi^{11}xy - \phi_{xx}\frac{x(1-x)}{2} - \phi_{yy}\frac{y(1-y)}{2}. \tag{2.42}$$

For each interpolation, the second-order derivatives are computed as

$$\begin{aligned}
\text{Linear} \qquad & \phi_{xx} = 0, \\
\text{Quadratic} \qquad & \phi_{xx} = \phi^{00}_{xx}(1-x)(1-y) + \phi^{01}_{xx}(1-x)y + \phi^{10}_{xx}x(1-y) + \phi^{11}_{xx}xy, \\
\text{ENO} \qquad & \phi_{xx} = \underset{\text{nodes}(\mathcal{C})}{\text{minmod}}\ \phi^n_{xx}, \\
\text{WENO} \qquad & \phi_{xx} = \frac{1}{W_x}\sum_{\text{n}\in\text{nodes}(\mathcal{C})} \phi^n_{xx} w^n_x, \qquad w^n_x = \frac{1}{|\phi^n_{xx}|^2}, \qquad W_x = \sum_{n\in\text{nodes}(\mathcal{C})} w^n_x,
\end{aligned}$$

where ϕ^n_{xx} are the discrete second-order derivatives at node n, approximated using the finite-differences method presented in [114]. As with the gradient and Laplacian, this definition of the ghost node values results in a stencil with four direct neighbors, with at most one ghost neighbor.

Using these new neighborhoods, we can construct discretizations for the standard differential operators, such as the nodal Laplacian $\mathcal{L}_N$, divergence $\mathcal{D}_N$ and gradient $\mathcal{G}_N$. As Figure 2.6 illustrates, these operators evaluated at any node n_0 for any given sampled Hodge variable ϕ and any given velocity field $\mathbf{u} = (u, v)$ are expressed in terms of their neighborhood information as

$$\mathcal{L}_N\phi\Big|_0 = \frac{2}{r+\ell}\left(\frac{\phi_r - \phi_0}{r} - \frac{\phi_0 - \phi_\ell}{\ell}\right) + \frac{2}{t+b}\left(\frac{\phi_t - \phi_0}{t} - \frac{\phi_0 - \phi_b}{b}\right), \tag{2.43}$$

$$\mathcal{D}_N(u,v)\Big|_0 = \frac{u_r - u_\ell}{r+\ell} + \frac{v_t - v_b}{t+b}, \tag{2.44}$$

$$\mathcal{G}_N\phi\Big|_0 = \left(\frac{\ell}{r+\ell}\frac{\phi_r - \phi_0}{r} + \frac{r}{r+\ell}\frac{\phi_0 - \phi_\ell}{\ell}, \frac{b}{t+b}\frac{\phi_t - \phi_0}{t} + \frac{t}{t+b}\frac{\phi_0 - \phi_b}{b}\right). \tag{2.45}$$

Note that our definitions of the Laplacian and gradient are identical to those in previous studies (see *e.g.*, [114, 112, 166, 168, 169, 93]), and in particular, they have been shown to yield supra convergence, in the sense that both the solution and its gradient converge with second-order accuracy [114]. The divergence operator, however, does not follow the construction proposed by Min and Gibou [114, 112]. Here, we use the standard central difference formula to construct a first-order accurate approximation of the divergence operator instead of creating weighted averages of the forward and backward

derivative and constructing a second-order centered difference approximation. It was observed in [112] that using the second-order divergence in a purely nodal context leads to an unstable projection operator.

To establish the connection between the staggered and collocated projection, we must formally define all discrete differential operators and the interpolations needed to bridge between them, as well as the spaces they act on. Using ghost nodes allows us to see these operators as natural extensions of their uniform counterparts.

Interpolation: Notation and Definitions

From the set of nodes and edges N we define $\tilde{N}$ as the set of all grid nodes and ghost neighbors (depicted as empty circles in Figure 2.6), and refer to it as the set of ghosted nodes. For each ghost node, we define a ghost edge (depicted as empty triangles in Figure 2.6), centered between the ghost node and the corresponding hanging node. We denote by $\tilde{E}$ the set of ghosted edges, *i.e.* grid, and ghost edges. As we did before, for any two discrete spaces A and B, defined from the grid structure, we will define $\mathcal{I}_A^B : A \to B$ as the interpolation from A to B.

As Figure 2.6 illustrates, we define the interpolation from the set of ghosted nodes to ghosted edges, $\mathcal{I}_{\tilde{N}}^{\tilde{E}} : \tilde{N} \to \tilde{E}$, and the interpolation from the set of ghosted edges to the set of nodes, $\mathcal{I}_{\tilde{E}}^{N} : \tilde{E} \to N$, as the standard linear interpolations from the nearest neighbors. We also define the ghost operators $\mathcal{I}_{\mathcal{G}}, \mathcal{I}_{\mathcal{D}} : N \to \tilde{N}$ which return the ghost values needed for the discrete gradient and divergence from the nodal values. Similarly, we define $\mathcal{I}_{E}^{\tilde{E}} : E \to \tilde{E}$. We define the operator $\mathcal{I}_{\tilde{E}}^{E}$ as the canonical restriction from $\tilde{E}$ to E. The staggered gradient operator, $\mathcal{G} : \tilde{N} \to \tilde{E}$, is constructed using the standard second-order central difference scheme and the staggered divergence operator, $\mathcal{D} : \tilde{E} \to N$ is constructed using a weighted first-order central difference scheme (see Figure 2.6).

Staggered Projection on Quadtree Grids

We start by observing that the nodal Laplacian is obtained by taking the divergence of the gradient computed for the ghosted values, *i.e.* $\mathcal{L} = \mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}$, and so the projection operator $\mathcal{P} : E \to E$ has the form

$$\mathcal{P} = \mathcal{I} - \mathcal{I}_{\tilde{E}}^{E}\mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}\mathcal{I}_{E}^{\tilde{E}}. \tag{2.46}$$

Because $\mathcal{I}_{\tilde{E}}^{E}\mathcal{I}_{E}^{\tilde{E}} = \mathcal{I}$ (i.e un-ghosting the ghosted edges' values returns the edges' values), we can rewrite the above definition by defining the operator $\mathcal{P}_{\tilde{E}}$ as

$$\mathcal{P} = \mathcal{I}_{\tilde{E}}^{E} \underbrace{\left(\mathcal{I} - \mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}\right)}_{\mathcal{P}_{\tilde{E}}} \mathcal{I}_{E}^{\tilde{E}}. \tag{2.47}$$

$\mathcal{P}_{\tilde{E}}$ should be interpreted as the projection on the space of ghosted edges. Squaring it, we see that

$$\mathcal{P}_{\tilde{E}}^2 = \mathcal{I} - 2\mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D} + \mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}, \tag{2.48}$$

$$\mathcal{P}_{\tilde{E}}^2 = \mathcal{I} - \mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}, \tag{2.49}$$

$$\mathcal{P}_{\tilde{E}}^2 = \mathcal{P}_{\tilde{E}}, \tag{2.50}$$

and so $\mathcal{P}_{\tilde{E}}$ is indeed a projection[2].

Stability of the Collocated Projection

To connect the staggered and collocated projections, we start by recognizing that the nodal Laplacian $\mathcal{L}_N$ is the composition of the staggered divergence with the staggered gradient (see Figure 2.6)

$$\mathcal{L}_N = \mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}. \tag{2.51}$$

As it was the case on uniform grids, the nodal divergence and gradient remain related to their staggered counterparts through linear interpolations, only now these relationships involve the ghost interpolation operators $\mathcal{I}_{\mathcal{G}}$ and $\mathcal{I}_{\mathcal{D}}$

$$\mathcal{D}_N = \mathcal{D}\mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}, \tag{2.52}$$

$$\mathcal{G}_N = \mathcal{I}_{\tilde{E}}^{N}\mathcal{G}\mathcal{I}_{\mathcal{G}}. \tag{2.53}$$

Using the above expression, the nodal projection operator

$$\mathcal{P}_N = \mathcal{I} - \mathcal{G}_N \left(\mathcal{L}_N\right)^{-1} \mathcal{D}_N, \tag{2.54}$$

can be expressed as

$$\mathcal{P}_N = \mathcal{I} - \mathcal{I}_{\tilde{E}}^{N}\mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}\mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}. \tag{2.55}$$

We consider a compressible eigenpair (λ, X) such that

$$\mathcal{P}_N X = \lambda X, \qquad \mathcal{D}_N X \neq 0, \qquad \text{and} \qquad \lambda \neq 1. \tag{2.56}$$

We multiply the whole equation on the left by the operator $\mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}$, and define $Y = \mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}X$, to obtain

$$\mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}\mathcal{P}_N X = Y - \underbrace{\mathcal{I}_{\tilde{N}}^{\tilde{E}}\mathcal{I}_{\mathcal{D}}\mathcal{I}_{\tilde{E}}^{N}}_{\mathcal{I}_p} \underbrace{\mathcal{G}\mathcal{I}_{\mathcal{G}} \left(\mathcal{D}\mathcal{G}\mathcal{I}_{\mathcal{G}}\right)^{-1} \mathcal{D}}_{\mathcal{Q}_{\tilde{E}} = \mathcal{I} - \mathcal{P}_{\tilde{E}}} Y = \lambda Y, \tag{2.57}$$

[2]Note that, as it is the case in the continuous world, this property only relies on the fact that the staggered Laplacian is the composition of the divergence and gradient operators. We, for example, did not have to assume that these two operators are the negative transpose of each other.

therefore $(\mathcal{I} - \mathcal{I}_p \mathcal{Q}_{\tilde{E}})\, Y = \lambda Y$, and λ is also an eigenvalue for the operator $\mathcal{M} = \mathcal{I} - \mathcal{I}_p \mathcal{Q}_{\tilde{E}}$.

To conclude, we will find a condition under which $\mathcal{M}^k$ converges as $k \to \infty$, which is a sufficient condition for the eigenvalues of $\mathcal{M}$, other than one, to be strictly less than one in absolute value. To do this, we define $\epsilon = \mathcal{I} - \mathcal{I}_p$, and rewrite $\mathcal{M}$ as

$$\mathcal{M} = \mathcal{I} - (\mathcal{I} - \epsilon)\, \mathcal{Q}_{\tilde{E}}, \tag{2.58}$$

and

$$\mathcal{M} = \mathcal{P}_{\tilde{E}} + \epsilon \mathcal{Q}_{\tilde{E}}. \tag{2.59}$$

Since $\mathcal{P}_{\tilde{E}}$ is a projection,

$$\mathcal{Q}_{\tilde{E}} \mathcal{P}_{\tilde{E}} = (\mathcal{I} - \mathcal{P}_{\tilde{E}}) \mathcal{P}_{\tilde{E}} = 0. \tag{2.60}$$

Squaring $\mathcal{M}$, we get

$$\mathcal{M}^2 = \mathcal{P}_{\tilde{E}}^2 + \mathcal{P}_{\tilde{E}} \epsilon \mathcal{Q}_{\tilde{E}} + \epsilon \mathcal{Q}_{\tilde{E}} \mathcal{P}_{\tilde{E}} + (\epsilon \mathcal{Q}_{\tilde{E}})^2, \tag{2.61}$$

which, using the fact that $\mathcal{P}_{\tilde{E}}^2 = \mathcal{P}_{\tilde{E}}$ and the orthogonality condition (2.60), we simplify into

$$\mathcal{M}^2 = \mathcal{P}_{\tilde{E}} (\mathcal{I} + \epsilon \mathcal{Q}_{\tilde{E}}) + (\epsilon \mathcal{Q}_{\tilde{E}})^2. \tag{2.62}$$

By recurrence, we obtain that

$$\mathcal{M}^k = \mathcal{P}_{\tilde{E}} \left(\sum_{i=0}^{k-1} (\epsilon \mathcal{Q}_{\tilde{E}})^i \right) + (\epsilon \mathcal{Q}_{\tilde{E}})^k, \tag{2.63}$$

from which we conclude that the limit of $\mathcal{M}^k$ as $k \to \infty$ exists if and only if $\rho(\epsilon \mathcal{Q}_{\tilde{E}})$, the spectral radius of $\epsilon \mathcal{Q}_{\tilde{E}}$, is less than 1, *i.e.*

$$\rho(\epsilon \mathcal{Q}_{\tilde{E}}) < 1 \qquad \Leftrightarrow \qquad \rho((\mathcal{I} - \mathcal{I}_p)\, \mathcal{Q}_{\tilde{E}}) < 1. \tag{2.64}$$

Implications and Key Role of the Construction of Ghost Nodes

To interpret the above condition, recall that

$$\mathcal{I}_p = \mathcal{I}_{\tilde{N}}^{\tilde{E}} \mathcal{I}_{\mathcal{D}} \mathcal{I}_{\tilde{E}}^{N} \qquad \text{and} \qquad \epsilon = \mathcal{I} - \mathcal{I}_p. \tag{2.65}$$

Our proof does not rely on a specific form for the divergence ghost node interpolation $\mathcal{I}_{\mathcal{D}}$; rather, it tells whether, for a given interpolation, the projection is stable. Furthermore looking at the expression of $\mathcal{I}_p$ and ϵ, we see that $\mathcal{I}_{\mathcal{D}}$ is the only adjustable parameter for the

stability of the projection. Hence, the construction of the ghost values for the divergence dictates the stability of the projection operator.

We should note that the above condition is satisfied if $\|\epsilon\mathcal{Q}_{\tilde{E}}\| < 1$, and so for the condition (2.64) to be satisfied, it is sufficient to have

$$\|\epsilon\| < \frac{1}{\|\mathcal{Q}_{\tilde{E}}\|}. \tag{2.66}$$

Since the staggered operators $\mathcal{P}_{\tilde{E}}$ and $\mathcal{Q}_{\tilde{E}}$ on uniform grids are orthogonal with norm equal to one, we expect their norms on adaptive grids to be close to one (*i.e.* $\|\mathcal{Q}_{\tilde{E}}\| \approx 1$), and so we expect the above stability condition to be met if $\|\epsilon\| \lessapprox 1$. In other words, we expect that if the norm of the interpolation error introduced by $\mathcal{I}_p$ is less than one, which is a realistic expectation as $\mathcal{I}_p$ is the product of consistent interpolations, and therefore consistent itself, then the projection is converging.

Intuitively, one could think that for the iterated projection to converge, $\mathcal{I}_p$ must be stable and that, therefore, its norm must be less than one. Our proof suggests that this intuition is wrong and that, somewhat surprisingly, an unstable interpolation can lead to a converging algorithm as long as (2.66) is satisfied. This is essential as $\mathcal{I}_P$ involves interpolating from the nodes to the ghost nodes, which, as formula (2.41) illustrates, is done using quadratic interpolating polynomials and can virtually introduce instabilities. The other two interpolations ($\mathcal{I}_{\tilde{N}}^{\tilde{E}}$ and $\mathcal{I}_{\tilde{E}}^{N}$) involved in $\mathcal{I}_P$ are constructed by taking weighted averages, and thus stable.

Unfortunately, our general ghost node construction is too complicated for us to prove whether condition (2.64) is met or not. Even for simpler constructions, such as a linear one, where the ghost values are obtained as the weighted average of existing values, $\mathcal{I}_P$ is now the product of three averaging stochastic matrices, and thus itself a square stochastic matrix with positive coefficients, we can only conclude that its eigenvalues are in the disk $D(0,1)$, which is insufficient.

Overall, the condition (2.64) feels achievable and, in fact, less restrictive than one would have naively thought, and so we are reasonably hopeful that our method is stable. Yet, the constructions of our interpolation operators, even in the low-order scenario, are too complicated for us to prove it formally. In addition, we should point out that boundary effects are left out of our theoretical study. We, therefore, resort to an in-depth computational verification.

2.3.4 Computational Verification

In this section, we verify the stability of our nodal projection operator by computing its convergence properties using a standard example with homogeneous Neumann boundary

conditions and by verifying the stability of our nodal projection operator in the presence of different combinations of boundary and interface conditions. For the examples in this section and subsequent tests, wall and boundary conditions are treated using the second-order hybrid Finite-Volume/Finite Difference approach presented in [169], itself based on previous studies [168]. For interface conditions, the fluid quantities in Ω^- are systematically extended to Ω^+ using the PDE-based approach proposed by Aslam [8]. All other problem-specific treatments are described in the appropriate sections.

Projection Test - Supra-convergence of the Hodge Variable

We verify the convergence and stability of our projection operator by repeatedly applying the operator to an initially non-divergence-free velocity field and computing the error between the resulting velocity field and the exact divergence-free velocity field, as was done in previous studies [112, 121, 67]. Doing so, we test the stability of our projection and the convergence of the Hodge variable as the grid resolution goes to zero. We consider the two-dimensional velocity field:

$$u^*(x,y) = \sin(x)\cos(y) + x(\pi - x)y^2\left(\frac{y}{3} - \frac{\pi}{2}\right), \tag{2.67}$$

$$v^*(x,y) = -\cos(x)\sin(y) + y(\pi - y)x^2\left(\frac{x}{3} - \frac{\pi}{2}\right), \tag{2.68}$$

in the domain $\Omega = [0,\pi]^2$ with Neumann boundary conditions. The divergence-free velocity field corresponding to u^* and v^* above is:

$$u(x,y) = \sin(x)\cos(y), \tag{2.69}$$

$$v(x,y) = -\cos(x)\sin(y). \tag{2.70}$$

We perform this test by starting with a randomized quadtree with 240 initial splits. Our goal here is to verify the order of accuracy for our projection operator in extreme scenarios where the grid is highly non-graded. We then successively refine the grid by further splitting each leaf, each time splitting the local spatial resolution in half, and monitor the error. The results of this test are shown in Figure 2.8 and Table 2.1, along with the initial random quadtree in Figure 2.7. As expected, the projection is stable, and we see second-order convergence in the velocity field.

Projection Test - Stability

In this test, we specifically look at the stability of our projection operator in the presence of a variety of boundary and interface conditions. Using the same initial velocity

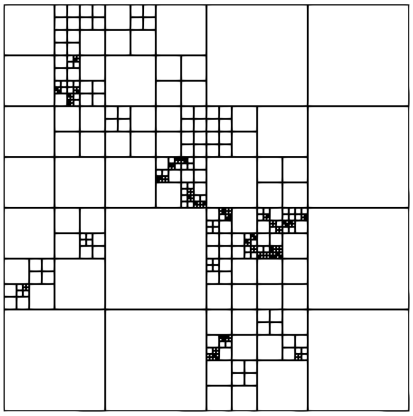


Figure 2.7: Initial highly non-graded quadtree used for the Projection Test initialized with 240 random splits

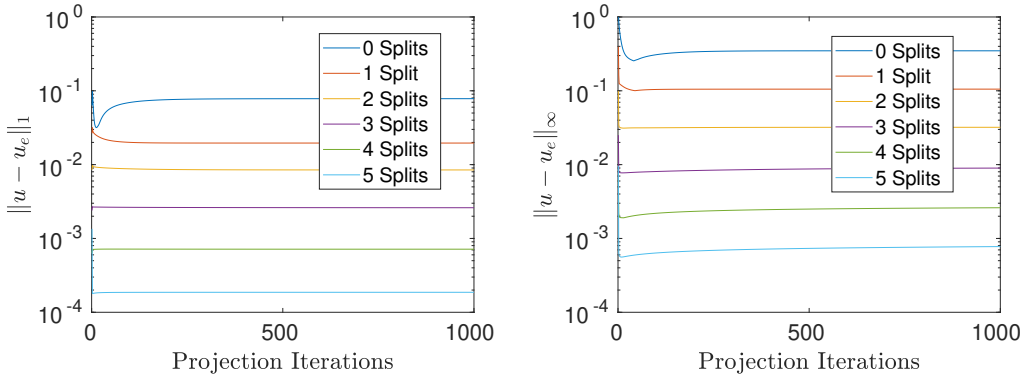


Figure 2.8: The L^1 (left) and L^∞ (right) errors for the projection test on a randomly generated quadtree with increasing number of splits.

Table 2.1: The final error and order of convergence for the projection test on a randomly generated quadtree with an increasing number of splits.

Splits	L^1	Order	L^∞	Order
0	7.82e-02	-	3.48e-01	-
1	1.96e-02	2.00	1.05e-01	1.73
2	8.47e-03	1.21	3.20e-02	1.72
3	2.61e-03	1.70	8.99e-03	1.83
4	7.16e-04	1.87	2.61e-03	1.79
5	1.87e-04	1.94	7.76e-04	1.75

field as in the previous example, (2.69) and (2.70), we successively apply the projection operator to this field and monitor the norm of the variation between the velocity at the current projection iteration and the velocity after the final projection iteration. For a stable operator, this variation will tend to zero, up to the solver tolerance.

The results of this test are shown in Figure 2.9. As expected, the variation in the norm of the velocity decreases as we successively apply our projection operator. For these examples, the tolerance of our linear solver was set to 10^{-12}. As expected, we see that our collocated projection operator is numerically stable for all boundary and interface conditions tested. We also note that in practice, only a small number of iterations of the projection operator will be used and, as we later show, second-order accuracy in both the velocity and Hodge variable can be achieved with only a single projection applied.

2.4 Nodal Navier-Stokes Solver

In this section, we integrate our nodal projection operator into a general framework for solving the incompressible Navier-Stokes equations. We begin by presenting an overview of our time-stepping algorithm and then we provide a brief introduction of the specific numerical techniques used to compute the intermediate velocity field. We conclude this section by solving an analytical solution to the Navier-Stokes equations with our nodal solver and comparing these results with the MAC-based solver presented in [67]. Furthermore, we use this example to discuss the practical number of projection iterations that need to be applied when using our nodal solver.

2.4.1 Overview of the Time-Stepping Algorithm

We advance our velocity field from $\mathbf{u}^n$ to $\mathbf{u}^{n+1}$ using a standard two-step projection method. In the Viscosity step, we compute the intermediate velocity field, $\mathbf{u}^*$, using a semi-Lagrangian Backward Difference Formula (SLBDF) scheme for the temporal integration of the momentum equation where the viscous terms are treated implicitly. In the Projection step, we repeatedly apply our nodal projection operator with the stopping criteria described below. To account for the splitting error on the boundary (see Section 2.2.4), we iterate the Viscosity and Projection steps following a boundary correction procedure also described below. A summary of our time-stepping algorithm can be seen in Figure 2.10 and details of the specific techniques used can be found in the subsequent sections.

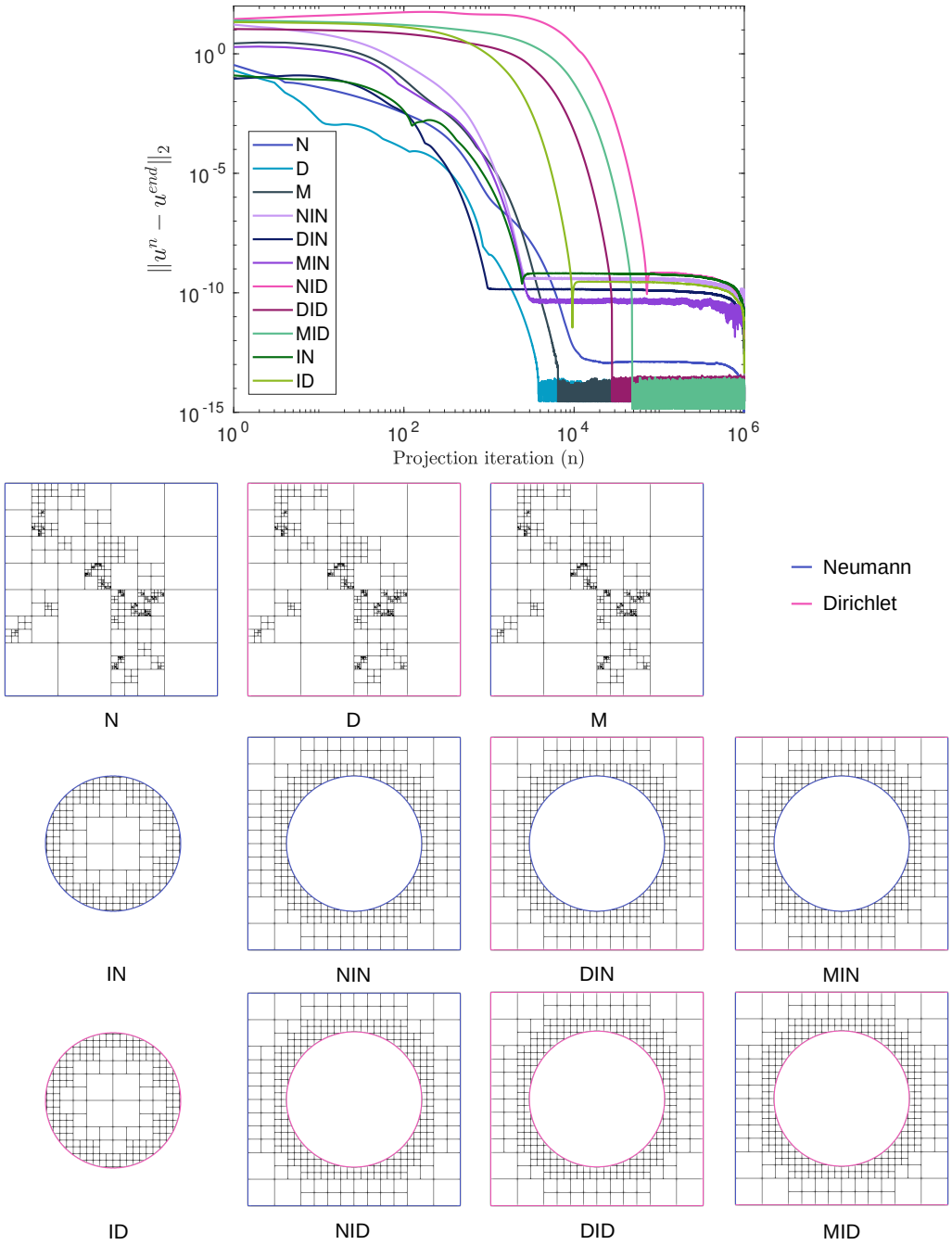


Figure 2.9: Variation in L^2 norm of the velocity after successive iterations of the projection operator for different sets of boundary conditions. Wall boundary conditions sets are denoted N for Neumann, D for Dirichlet, and M for mixed (a combination of Dirichlet and Neumann). When an interface is present, the wall boundary condition sets are augmented with IN for a Neumann interface and ID for a Dirichlet interface.

1 – Initialization

Initialize the corrective velocity boundary condition $\mathbf{c}$ as the final one from the previous iteration.

2 – Repeat until $\|\mathbf{u}^{n+1} - \mathbf{u}|_\Gamma\| < \epsilon_o$

2a – Viscosity step

Compute the intermediate velocity field $\mathbf{u}^*$ as the numerical solution of

$$\rho \frac{D\mathbf{u}^*}{Dt} = \mu \triangle \mathbf{u}^* + \mathbf{f} \qquad \forall \mathbf{x} \in \Omega^-/\Gamma, \tag{2.71}$$

$$\mathbf{u}^* = \mathbf{u}|_\Gamma + \mathbf{c} \qquad \forall \mathbf{x} \in \Gamma, \tag{2.72}$$

Initialize $\mathbf{u}^{n+1} = \mathbf{u}^*$

2b – Projection step

Repeat until $\|\mathbf{u}^{n+1} - \mathcal{P}_N \mathbf{u}^{n+1}\| < \epsilon_i \|\mathbf{u}^{n+1}\|$

Project the velocity field

$$\mathbf{u}^{n+1} = \mathcal{P}_N \mathbf{u}^{n+1}. \tag{2.73}$$

Compute the new correction

$$\mathbf{c} = \mathbf{c} - \omega \left(\mathbf{u}^{n+1} - \mathbf{u}|_\Gamma \right). \tag{2.74}$$

3 – Update

Adapt the mesh to $\mathbf{u}^{n+1}$ and update all the variables accordingly.

Figure 2.10: Outline of the algorithm for the construction of the solution $\mathbf{u}^{n+1}$ at time t_{n+1} from the solution $\mathbf{u}^n$ at the previous time step t_n.

Viscosity step - SLBDF integrator

To solve for the auxiliary velocity field $\mathbf{u}^*$, we start by integrating the momentum equation (2.1) using a semi-Lagrangian backward difference (SLBDF) scheme [102, 67, 167] with an adaptive time-step $\Delta t_n = t_{n+1} - t_n$, and implicit treatment of the viscous term, namely

$$\rho\left(\alpha\frac{\mathbf{u}^* - \mathbf{u}_d^n}{\Delta t_n} + \beta\frac{\mathbf{u}_d^n - \mathbf{u}_d^{n-1}}{\Delta t_{n-1}}\right) = \mu\triangle\mathbf{u}^* + \mathbf{f}, \tag{2.75}$$

where

$$\alpha = \frac{2\Delta t_n + \Delta t_{n-1}}{\Delta t_n + \Delta t_{n-1}},\ \beta = -\frac{\Delta t_n}{\Delta t_n + \Delta t_{n-1}}. \tag{2.76}$$

The departing velocities $\mathbf{u}_d^n$ and $\mathbf{u}_d^{n-1}$ are obtained by interpolating the velocity field at the departure points $\mathbf{x}_d^n$ and $\mathbf{x}_d^{n-1}$ and corresponding time steps. The adaptive time-step, Δt_n is determined by pre-setting the CFL number, CFL $= \max\|\mathbf{u}^n\|\Delta t/\Delta x_{\text{min}}$, where Δx_{min} corresponds to the maximum level of refinement. To find the departure points, we follow the characteristic curve backward using a second-order Runge-Kutta method (RK2, midpoint):

$$\hat{\mathbf{x}} = \mathbf{x}^{n+1} - \frac{\Delta t_n}{2}\cdot\mathbf{u}^{n+1}(\mathbf{x}^{n+1}), \tag{2.77}$$

$$\mathbf{x}_d^n = \mathbf{x}^{n+1} - \Delta t_n\cdot\mathbf{u}^{n+\frac{1}{2}}(\hat{\mathbf{x}}), \tag{2.78}$$

and

$$\bar{\mathbf{x}} = \mathbf{x}^{n+1} - \frac{(\Delta t_n + \Delta t_{n+1})}{2}\cdot\mathbf{u}^{n+1}(\mathbf{x}^{n+1}), \tag{2.79}$$

$$\mathbf{x}_d^{n-1} = \mathbf{x}^{n+1} - (\Delta t_n + \Delta t_{n+1})\cdot\mathbf{u}^{\text{mid}}(\bar{\mathbf{x}}). \tag{2.80}$$

We interpolate the intermediate velocities, $\mathbf{u}^{n+\frac{1}{2}}(\hat{\mathbf{x}})$ and $\mathbf{u}^{\text{mid}}(\bar{\mathbf{x}})$, from the velocity fields at t_{n-1} and t_n as

$$\mathbf{u}^{n+\frac{1}{2}}(\hat{\mathbf{x}}) = \frac{2\Delta t_{n-1} + \Delta t_n}{2\Delta t_{n-1}}\mathbf{u}^n(\hat{\mathbf{x}}) - \frac{\Delta t_n}{2\Delta t_{n-1}}\mathbf{u}^{n-1}(\hat{\mathbf{x}}), \tag{2.81}$$

$$\mathbf{u}^{\text{mid}}(\bar{\mathbf{x}}) = \frac{\Delta t_n + \Delta t_{n-1}}{2\Delta t_{n-1}}\mathbf{u}^n(\bar{\mathbf{x}}) + \frac{\Delta t_{n-1} - \Delta t_n}{2\Delta t_{n-1}}\mathbf{u}^{n-1}(\bar{\mathbf{x}}). \tag{2.82}$$

Viscosity step - improved trajectory reconstruction

As the semi-Lagrangian method shown above relies on knowing what $\mathbf{u}^{n+1}$ is, we must look to other explicit methods to find the departure point. Here, we discuss two departure point methods that utilize an RK2 scheme and approximate $\mathbf{u}^{n+1}$.

The first and most obvious choice for utilizing an RK2 method with known information is to replace $\mathbf{u}^{n+1}$ in (2.77) and (2.79) with $\mathbf{u}^n$. This midpoint rule formulation was popularized by Xiu and Karniadakis [183] and has been a popular integration choice for projection methods that utilize a semi-Lagrangian discretization of the advection term (see, *e.g.*, [67, 112]). Using $\mathbf{u}^n$ is akin to a constant Taylor approximation to $\mathbf{u}^{n+1}$.

This constant approximation is often a sufficient approximation for lower CFL numbers, but for larger CFL numbers, it can cause the computed departure point to lie too far from the characteristic prior to interpolation and can lead to less accurate velocity terms. By doing a first-order Taylor expansion of $\mathbf{u}^{n+1}(\mathbf{x}^{n+1})$ around t_n and using an Euler approximation for any derivative terms, we get

$$\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) = \mathbf{u}^n(\mathbf{x}^{n+1}) + \frac{\Delta t_n}{\Delta t_{n-1}} \left(\mathbf{u}^n(\mathbf{x}^{n+1}) - \mathbf{u}^{n-1}(\mathbf{x}^{n+1}) \right) + \mathcal{O}(\Delta t^2), \tag{2.83}$$

which we substitute into (2.77) and (2.79). By having a higher-order approximation for $\mathbf{u}^{n+1}(\mathbf{x}^{n+1})$, we expect our numerical simulations to have more accurate results, especially if a large time-step is used.

While we can mitigate this issue with a higher-order integration scheme, such as a fourth-order Runge-Kutta method, the order of accuracy of the semi-Lagrangian method is determined by both the integration scheme used to find the departure point and the interpolation scheme used to evaluate $\mathbf{u}$ at the departure point. Since we are using a quadratic interpolation scheme, using an integration scheme that is higher than second-order accurate is superfluous.

We use both integrating schemes for our numerical validation in Section 2.5. The classical scheme is used to directly compare the nodal projection method with previous numerical studies and experiments, and the new improved integrating scheme is used to compare the accuracy of the solver when implementing each integrating scheme.

Stopping criteria for iterative procedures

As our projection operator is not an exact orthogonal projection, we use an iterative procedure to create an approximately divergence-free velocity field. We perform successive projections until

$$\|\mathbf{u}^{n+1} - \mathcal{P}_N \mathbf{u}^{n+1}\| < \epsilon_i \|\mathbf{u}^{n+1}\|, \tag{2.84}$$

or a predefined maximum number of iterations, K_{max}, has been reached. Typically, we choose $\epsilon_i = 10^{-3}$, set $K_{max} = 5$, and only a few $(1-3)$ iterations are required to reach convergence.

The boundary correction $\mathbf{c}$ is designed to compensate for the splitting errors (see 2.2.4) in a similar fashion to what was proposed in [167]. It is designed so that when the correction reaches convergence (see Eq. (2.74)), the boundary condition on the solid object is satisfied (*i.e.*, $\mathbf{u}^{n+1} = \mathbf{u}$). The parameter ω controls the convergence rate and must be chosen in the range $0 < \omega < 1$. Similar corrections are performed on the wall of the computational domain $\partial\Omega$ but are not explicitly stated here for the sake of concision. Typically, we terminate the outer iterations when the error in the interface's velocity ($\|\mathbf{u}^{n+1} - \mathbf{u}|_\Gamma\|$) is less than $\epsilon_o = 10^{-3}$.

Refinement criteria

As it was done in previous studies [67, 167, 164], the mesh is dynamically refined near the solid interface and where high gradients of vorticity occur. At each iteration, we recursively apply the following splitting criterion at each cell.

We split each cell $\mathcal{C}$ if

$$\min_{n \in \text{nodes}(\mathcal{C})} |\phi(n)| \leq B \cdot \text{Lip}(\phi) \cdot \text{diag}(\mathcal{C}) \quad \text{and} \quad \text{level}(\mathcal{C}) \leq \text{max}_{\text{level}}, \tag{2.85}$$

or

$$\min_{n \in \text{nodes}(\mathcal{C})} \text{diag}(\mathcal{C}) \cdot \frac{\|\nabla \mathbf{u}(n)\|}{\|\mathbf{u}_\infty\|} \geq T_V \quad \text{and} \quad \text{level}(\mathcal{C}) \leq \text{max}_V, \tag{2.86}$$

or

$$\text{level}(\mathcal{C}) \leq \text{min}_{\text{level}}. \tag{2.87}$$

If none of these criteria are met, we merge $\mathcal{C}$ by removing all its descendants.

Here $\text{Lip}(\phi)$ is an upper estimate of the minimal Lipschitz constant of the level set function ϕ. Since the level set used to create the mesh will be reinitialized (*i.e.*, $|\nabla\phi| = 1$), we use $\text{Lip}(\phi) = 1.2$. $\text{diag}(\mathcal{C})$ is the length of the diagonal of cell $\mathcal{C}$. B is the user-specified width of the uniform band around the interface. T_V is the vorticity threshold and $\text{max}_{\text{level}}$ is the maximum grid level allowed for the vorticity-based refinement. Condition (3.41) ensures that a minimum resolution of $\text{min}_{\text{level}}$ is maintained.

As we use dynamic mesh refinement, it may be possible for the head and root of the characteristic to be located on different levels of adaptive mesh refinement. In general, we attempt to avoid this by adjusting thresholds for refinement or by increasing the uniform band around the fluid-solid interface. Nevertheless, this can occur and in that event the departure point is evaluated using Equation (2.42), with values from the coarser grid.

2.4.2 Verification - Analytic Vortex

Convergence study

We consider the following analytical solution to the Navier-Stokes Equations,

$$u(x, y, t) = \sin(x)\cos(y)\cos(t), \tag{2.88}$$
$$v(x, y, t) = -\cos(x)\sin(y)\cos(t), \tag{2.89}$$
$$p(x, y, t) = 0. \tag{2.90}$$

We set $\mu = 1$, $\rho = 1$, and define the forcing terms as

$$f_x = \sin(x)\cos(y)\left(2\mu\cos(t) - \rho\sin(t)\right) + \rho\cos^2(t)\sin(x)\cos(x), \tag{2.91}$$
$$f_y = \cos(x)\sin(y)\left(\rho\sin(t) - 2\mu\cos(t)\right) + \rho\cos^2(t)\sin(y)\cos(y). \tag{2.92}$$

We monitor the convergence of our solver as our mesh is refined and compare the results against the MAC-based solver of [67]. Each of the simulations is run until a final time of $t_f = \pi/3$ is reached with an adaptive time step defined by $\Delta t_n = \Delta x / \max \|u_n\|$. We set the maximum number of projections, K_{max}, to 5, with an error tolerance of 10^{-3} (see Section 2.4.1), and use the boundary correction procedure explained in Section 2.4.1. The results of this example are shown in Table 2.2 and in Figure 2.11.

We see that our nodal projection method achieves second-order convergence in both the L^1 and L^∞ norm for velocity as expected based on our discretization. For the Hodge variable, we also see second-order convergence in both L^1 and L^∞. If we compare the results of our nodal solver with the MAC-based solver of [67], we notice a distinct improvement in the L^∞ norm for velocity.

Impact of successive projections - determining K_{max}

In order to develop a practical guideline for the number of projection iterations that should be used in our nodal solver, we again consider the analytic vortex example from the previous section. We continue using the same adaptive time step, grid refinement criteria, and boundary correction procedures previously described. However, instead of using the error threshold described in 2.4.1, we explicitly define the number of projections to be performed at each time step. The results of this test are shown in Table 2.3 for the explicit number of projections set to 1, 5, and 10.

We emphasize that for this example, only a single projection is required to achieve second-order accuracy in the L^∞ norm for both velocity and the Hodge variable. Increasing the number of projections does not significantly change the solution or the order of accuracy. For this reason, we do not set a minimum number of projections for our

nodal solver. Instead, we specify a maximum number of projections, K_{max}, and typically use a default value of 5 projections. In practice and using an error threshold of 10^{-3} (see Section 2.4.1), we observe that a single projection is often sufficient.

Verification of the improved departure point reconstruction scheme

Finally, we use the analytic vortex example to verify that our improved departure point reconstruction scheme remains second-order accurate. Using an identical setup as above, we monitor the convergence of our solver using the original SLBDF scheme presented

Table 2.2: Convergence of x-component velocity and Hodge variable ϕ for the Nodes and MAC [67] implementations.

Velocity

	Nodes				MAC			
Level (max:min)	L^1	Order	L^∞	Order	L^1	Order	L^∞	Order
7:3	6.84e-04	-	2.34e-02	-	5.48e-03	-	2.18e-02	-
8:4	1.25e-04	2.46	2.72e-03	3.11	1.76e-03	1.64	8.65e-03	1.33
9:5	2.33e-05	2.42	9.56e-04	1.51	5.56e-04	1.66	3.28e-03	1.40
10:6	5.61e-06	2.05	2.19e-04	2.12	1.61e-04	1.79	1.64e-03	1.00
11:7	1.36e-06	2.05	4.44e-05	2.30	3.62e-05	2.15	3.84e-04	2.09
12:8	4.15e-07	1.71	9.55e-06	2.22	8.59e-06	2.08	2.45e-04	0.65

Hodge

	Nodes				MAC			
Level (max:min)	L^1	Order	L^∞	Order	L^1	Order	L^∞	Order
7:3	4.43e-05	-	4.16e-04	-	3.17e-04	-	1.62e-03	-
8:4	1.67e-05	1.41	9.48e-05	2.13	3.47e-05	3.19	2.47e-04	2.71
9:5	2.55e-06	2.71	1.44e-05	2.72	4.48e-06	2.95	5.85e-05	2.08
10:6	3.07e-07	3.05	2.45e-06	2.55	7.20e-07	2.64	1.44e-05	2.02
11:7	3.12e-08	3.30	5.49e-07	2.16	7.41e-08	3.28	2.81e-06	2.36
12:8	3.67e-09	3.09	1.05e-07	2.39	1.14e-08	2.70	6.72e-07	2.06

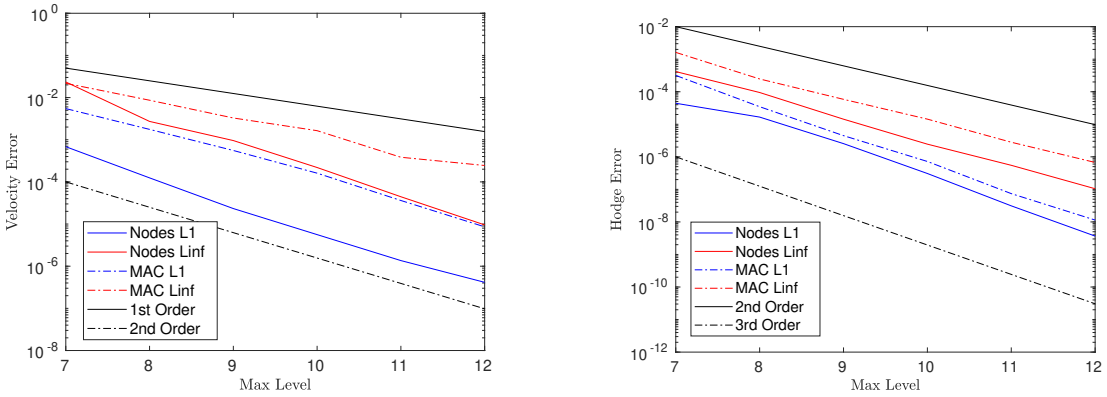


Figure 2.11: Visualization of the error convergence for the velocity and Hodge variable.

Table 2.3: Impact of the number of projections on the L^∞ error for the velocity and Hodge variable ϕ for the analytic vortex example.

	Projections = 1				Projections = 5				Projections = 10			
	Velocity		Hodge		Velocity		Hodge		Velocity		Hodge	
Level	L^∞	Order	L^∞	Order	L^∞	Order	L^∞	Order	L^∞	Order	L^∞	Order
7:3	2.36e-02	-	4.16e-04	-	4.19e-02	-	9.65e-05	-	5.44e-02	-	7.01e-05	-
8:4	2.79e-03	3.08	9.49e-05	2.13	6.86e-03	2.61	3.24e-05	1.58	8.74e-03	2.64	1.44e-05	2.28
9:5	9.74e-04	1.52	1.44e-05	2.72	2.32e-03	1.56	5.81e-06	2.48	3.03e-03	1.53	2.89e-06	2.32
10:6	2.25e-04	2.12	2.45e-06	2.56	4.92e-04	2.24	9.52e-07	2.61	6.38e-04	2.25	6.31e-07	2.20
11:7	4.58e-05	2.29	5.49e-07	2.16	9.92e-05	2.31	2.11e-07	2.17	4.58e-05	3.80	5.48e-07	0.20
12:8	9.96e-06	2.20	1.05e-07	2.38	1.82e-05	2.45	4.24e-08	2.32	2.45e-05	0.90	3.47e-08	3.98

in [67] and compare the results against the improved scheme presented herein. Again, the maximum number of projections, K_{max}, is set to 5, with an error tolerance for the projection set to 10^{-3}. The L^1 and L^∞ errors for the x-component velocity and the Hodge variable are shown in Table 2.4. As expected, we see that the improved reconstruction scheme maintains second-order accuracy.

Table 2.4: Convergence of the x-component velocity and Hodge variable Φ using the original and improved departure point reconstruction schemes.

Velocity

	Original Reconstruction				Improved Reconstruction			
Level (max:min)	L^1	Order	L^∞	Order	L^1	Order	L^∞	Order
7:3	6.84e-04	-	2.34e-02	-	9.08e-04	-	2.90e-02	-
8:4	1.25e-04	2.46	2.72e-03	3.11	1.13e-04	3.0.	2.73e-03	3.41
9:5	2.33e-05	2.42	9.56e-04	1.51	2.38e-05	2.24	9.58e-04	1.51
10:6	5.61e-06	2.05	2.19e-04	2.12	5.58e-06	2.09	2.20e-04	2.12
11:7	1.36e-06	2.05	4.44e-05	2.30	1.30e-06	2.10	4.47e-05	2.30
12:8	4.15e-07	1.71	9.55e-06	2.22	3.95e-07	1.72	9.55e-06	2.23

Hodge

	Original Reconstruction				Improved Reconstruction			
Level (max:min)	L^1	Order	L^∞	Order	L^1	Order	L^∞	Order
7:3	4.43e-05	-	4.16e-04	-	3.59e-05	-	3.18e-04	-
8:4	1.67e-05	1.41	9.48e-05	2.13	1.69e-05	1.09	9.72e-05	1.71
9:5	2.55e-06	2.71	1.44e-05	2.72	2.57e-06	2.72	1.47e-05	2.72
10:6	3.07e-07	3.05	2.45e-06	2.55	3.10e-07	3.05	2.44e-06	2.59
11:7	3.12e-08	3.30	5.49e-07	2.16	3.20e-08	3.28	5.59e-07	2.13
12:8	3.67e-09	3.09	1.05e-07	2.39	3.83e-09	3.06	1.08e-07	2.37

2.5 Validation Examples

In this section, we validate our node-based Navier-Stokes solver using several canonical problems in two and three spatial dimensions. We compare the results of our solver with

previous numerical studies [57, 50, 67, 121, 26, 28, 47, 116, 108, 82, 46] and experimental results [45] demonstrating the robustness of our solver. Furthermore, we use the Kármán vortex example to highlight the advantages of our improved departure point reconstruction scheme and the efficiency of our solver. Finally, we conclude this section by simulating a high Reynolds number flow past the UC Merced Beginnings sculpture to demonstrate that our solver can be used to simulate flows with non-trivial geometries.

2.5.1 Driven Cavity

The lid-driven cavity problem is a canonical example for validating an incompressible flow solver. Following the numerical experiments of Ghia *et al.* [57] and Erturk *et al.* [50], we consider the domain $\Omega = [0, 1]^2$ with the top wall moving at a unit velocity and no-slip boundary conditions on the velocity for all four walls. We set the density $\rho = 1$ and the Reynolds number to $Re = 1\ 000$.

The numerical simulations are performed using minimum and maximum quadtree levels of 8 and 10, respectively. We set the CFL number to 1.0 and dynamically refine the mesh based on the procedure discussed in Section 2.4.1. In Figure 2.12 we plot the results of our simulation with data from the numerical experiments of Ghia *et al.* [57] and Erturk *et al.* [50]. We see excellent agreement between our simulation and the previous studies.

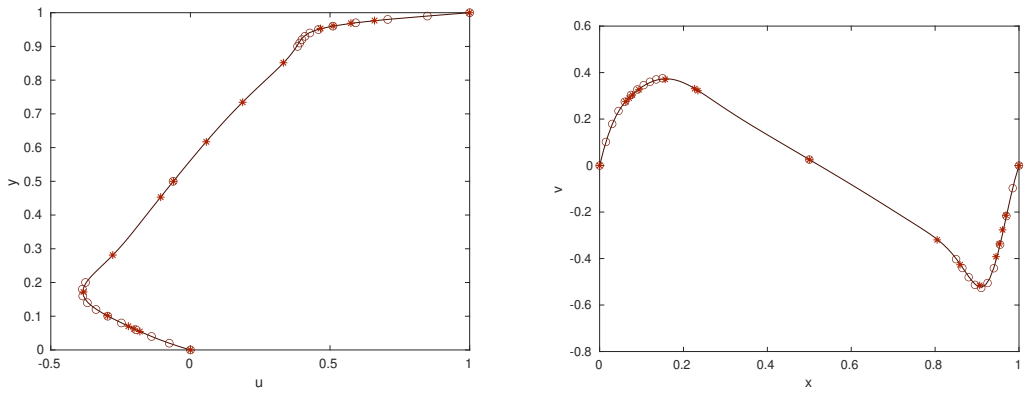


Figure 2.12: x- and y-components of the velocity field in the driven cavity example. The black circles are the results from Ghia *et al.* [57] and the red stars are from Erturk *et al.* [50]. The line results are taken from our solver. This simulation was run with a minimum and maximum quadtree level of 8 and 10, respectively, and a CFL of 1.0.

2.5.2 Oscillating Cylinder

We demonstrate the ability of our solver to handle moving interfaces by considering an oscillating cylinder example. The oscillating cylinder is a popular example of a one-way fluid-structure interaction problem that has been studied in both experimental and numerical contexts [45, 80, 150, 67]. In this test, oscillatory motion is imposed on a cylinder in a quiescent fluid and the flow around the cylinder is characterized by the Reynolds number and the Keulegan–Carpenter number defined as

$$KC = \frac{u_{max}}{2rf}.$$

We follow the problem definition of [45] and [67] and set these dimensionless parameters to $Re = 100$ and $KC = 5$. We set the radius of the cylinder to $r = 0.05$, our frequency to $f = 1$, and the amplitude of oscillation to $x_0 = 1.5914r$. We consider the computational domain of $\Omega = [-1, 1]^2$ with homogeneous Dirichlet boundary conditions and define the center position of our cylinder as

$$x_c = x_0(1 - \cos(2\pi f t)).$$

We compare the results of our solver with those of [45] and [67] by computing the drag coefficient. We compute the drag and lift forces on the disk by integrating

$$\mathbf{F} = \int_{\Gamma} (-p\mathbf{I} + 2\mu\sigma) \cdot \mathbf{n} \tag{2.93}$$

where Γ is the boundary of the cylinder, σ is the symmetric stress tensor and $\mathbf{n}$ is the outward facing normal vector to the cylinder. The drag and lift coefficients, C_d and C_l, are computed by re-scaling the components of $\mathbf{F}$ by $\rho r \|\mathbf{u}\|_\infty^2$. Numerically, the integral is computed using the quadrature rules of [113] on third-order smooth extensions [8] of the integrands.

For this example, we are able to use an adaptive mesh with minimum and maximum quadtree levels of 4 and 11, respectively. Additionally, we see good quantitative agreement with both studies for CFL numbers up to 10. Figure 2.13 shows the time history of the drag coefficient for $Re = 100$ and $KC = 5$ and a comparison with previous studies using a CFL of 10. In Figure 2.14, we show the time evolution of the vorticity and the adaptive grid refinement of the oscillating cylinder.

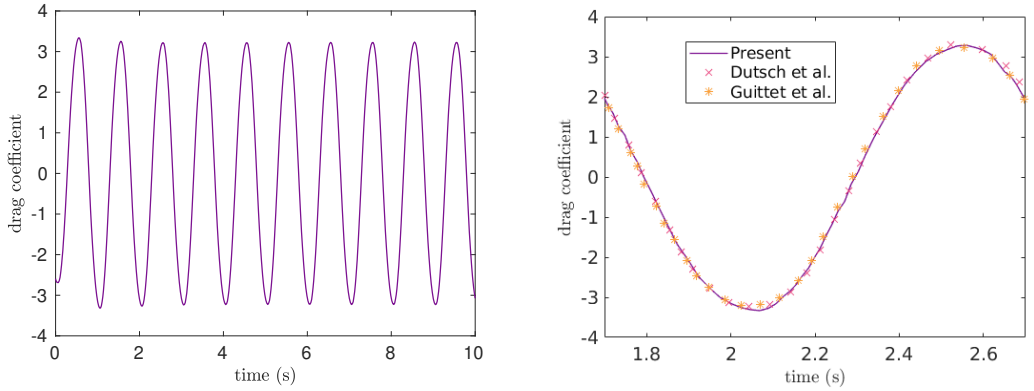


Figure 2.13: History of the drag coefficient for an oscillating cylinder with Re = 100 and KC = 5. The red crosses are the original experimental data from Dutsch et al. [45] and the orange stars were computed by Guittet *et al.* [67]. Data points were digitized using the original figures. This simulation was run with a minimum and maximum quadtree level of 4 and 11, respectively, and a CFL of 10.0.

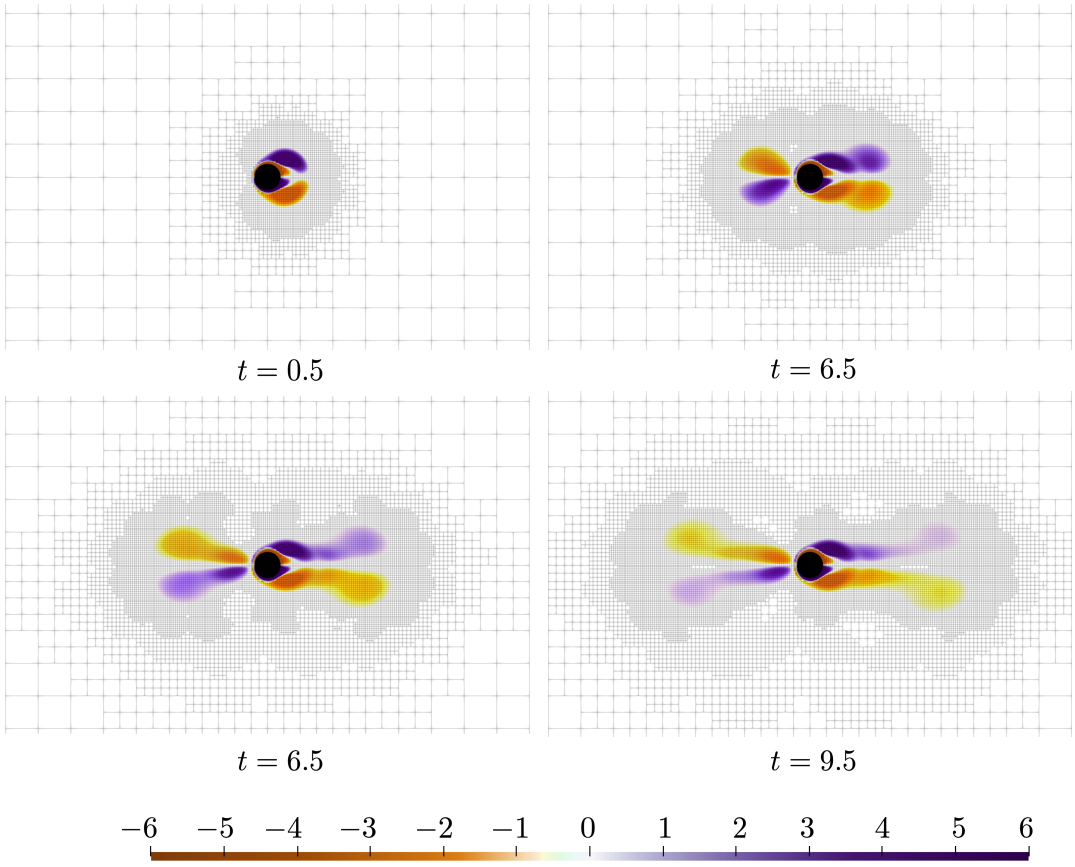


Figure 2.14: Time evolution of the vorticity and adaptive grid refinement for the oscillating cylinder with $Re = 100$ and $KC = 5$.

2.5.3 Von Kármán Vortex Street

Next, we validate our solver by considering the analysis of flow past a cylinder in two and three spatial dimensions.

Two Dimensions

The setup for this problem consists of a cylinder of radius $r = 0.5$ with its center at the location $(8, 0)$ in a rectangular domain $\Omega = [0, 32] \times [-8, 8]$. For the velocity, we impose a Dirichlet boundary condition of $u = 1$ on the left, top, and bottom walls and a homogeneous Neumann boundary condition on the right wall. We impose homogeneous Neumann boundary conditions on the left, right, and bottom walls for the Hodge variable and a Dirichlet boundary condition of $\phi = 0$ on the right wall. For this case, we ran our simulation on adaptive quadtree grids with minimum and maximum refinement levels of 7 and 10, respectively. To solve the advective term of the momentum equation, we use the classical departure point reconstruction method described in Section 2.4.1. We also set a uniform band around the disk to properly capture the boundary layer around the interface.

We solve this problem for several CFL numbers to quantify the impact the time step has on the drag and lift forces on the cylinder and to act as a calibration tool for more complex two- and three-dimensional problems. In Figure 2.15, we see that as we increase the CFL number, both the average drag coefficient values and the amplitude of the coefficient oscillations increase. Qualitatively (Figure 2.16), we observe a more severe destabilization of the wake past the cylinder for the higher CFL numbers, which is consistent with the increased forces. Nonetheless, we see that our solver can produce good qualitative results for large CFL numbers. We also see that the drag and lift coefficients are consistent with previous numerical and experimental results (Table 2.5).

Table 2.5: Drag and lift coefficients for the two-dimensional flow past disk problem.

	Drag coefficient		Lift coefficient	
	$Re = 100$	$Re = 200$	$Re = 100$	$Re = 200$
Ng *et al.* [121]	$1.368 \pm .016$	1.373 ± 0.050	± 0.360	± 0.724
Braza *et al.* [26]	$1.364 \pm .015$	1.400 ± 0.050	± 0.250	± 0.750
Calhoun [28]	$1.330 \pm .014$	1.172 ± 0.058	± 0.298	± 0.668
Engelman *et al.* [47]	$1.411 \pm .010$	-	± 0.350	-
Guittet *et al.* [67]	$1.401 \pm .011$	1.383 ± 0.048	± 0.331	± 0.705
Present (CFL = 2)	$1.387 \pm .019$	1.370 ± 0.060	± 0.346	± 0.762

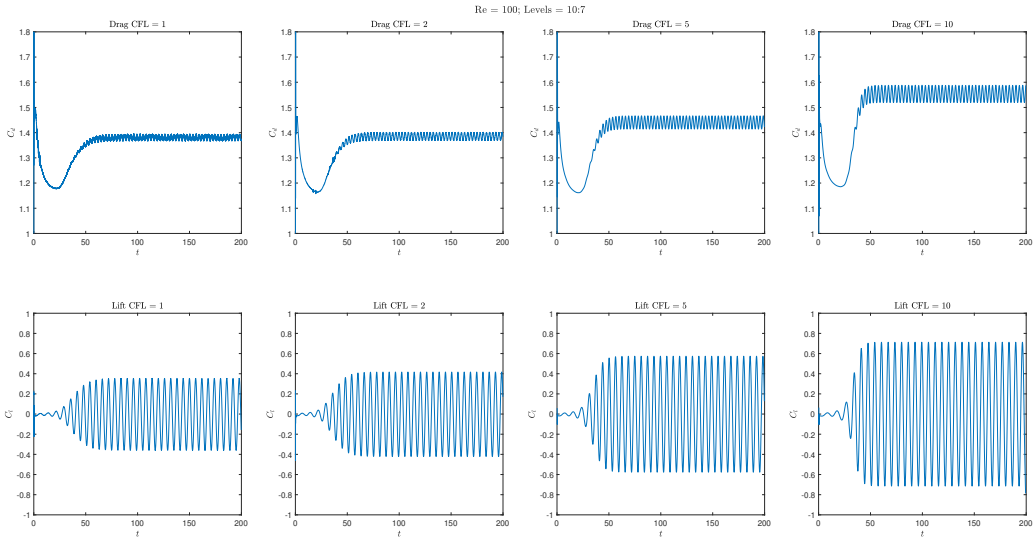


Figure 2.15: Drag and lift computations for the 2-dimensional flow past disk problem with $Re = 100$ for CFL numbers of 1, 2, 5, and 10.

Impact of departure point reconstruction

We implement the two-dimensional flow past disk problem above, with the only difference being that we use our improved departure point method. For $Re = 100$, we see a significant improvement in both the drag and lift coefficients for higher CFL numbers (Figure 2.17) compared to the implementation found in Figure 2.15.

These significant improvements are further realized when looking at the data qualitatively. In Figure 2.18, we compare a snapshot of the vorticity profile with $CFL = 10$ of the two departure point reconstruction methods. The destabilization is not as present in the improved departure point method (as expected from the computed drag and lift computation) and looks similar to the more accurate, lower CFL number simulations.

Three Dimensions

To further validate the solver and demonstrate our solver's capabilities in three dimensions, we solve the three-dimensional flow past sphere problem. The setup is similar to the 2D example above, where the radius $r = 0.5$ and is centered at $(8, 0, 0)$ in the domain $\Omega = [0, 32] \times [-8, 8] \times [-8, 8]$, with minimum and maximum refinement levels of

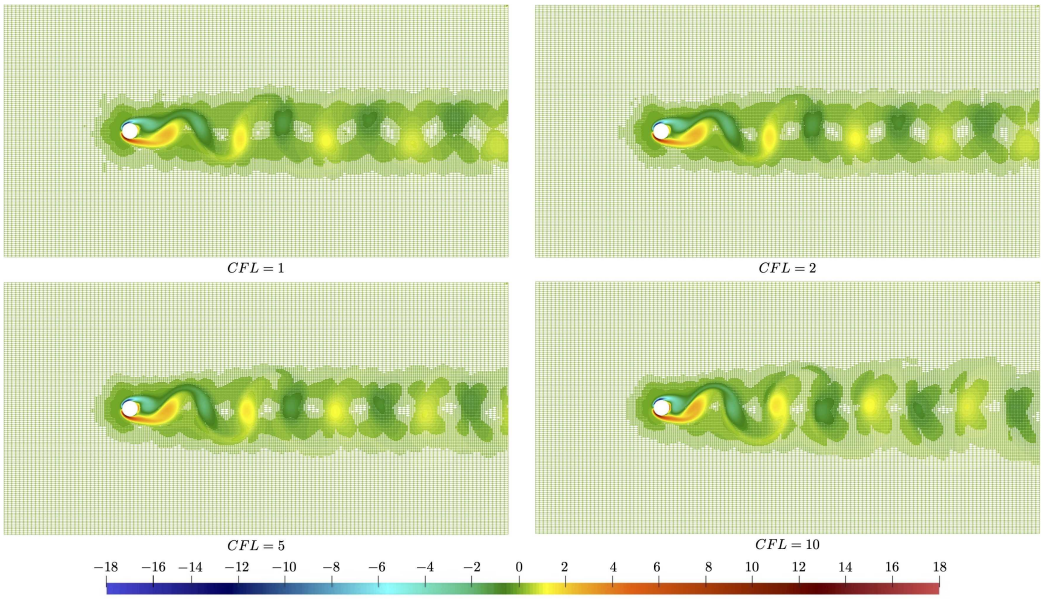


Figure 2.16: Visualization of the 2-dimensional flow past disk problem with Re = 100 and levels = 10:7. The vorticity and grid are shown.

4 and 10, respectively. Like in two dimensions, we quantify the capabilities of our solver by computing the drag coefficients in a similar way to equation (2.93). These results can be seen in Table 2.6, which shows that our data agree with previous numerical and experimental studies. We also visualize the $Re = 350$ simulation in Figure 2.19.

Table 2.6: Drag coefficients for the three-dimensional flow past sphere problem.

	$Re = 100$	$Re = 200$	$Re = 300$	$Re = 350$
Mittal *et al.* [116]	1.08	-	0.67	0.62
Marella *et al.* [108]	1.06	-	0.621	-
Le Clair *et al.* [97]	1.096	0.772	0.632	-
Johnson and Patel [82]	1.1	0.8	0.656	-
Guittet *et al.* [67]	1.112	0.784	0.659	0.627
Egan *et al.* [46]	1.11	-	0.673	0.633
Present (CFL = 2)	1.058	0.768	0.646	0.601

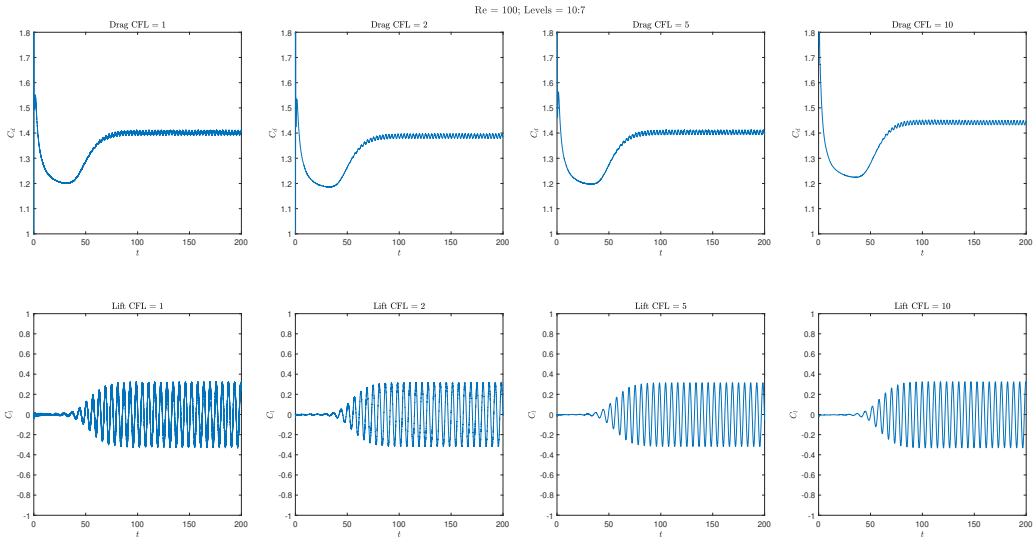


Figure 2.17: Drag and lift computations for the 2-dimensional flow past disk problem with $Re = 100$ for CFL numbers of 1, 2, 5, and 10, with the improved departure point reconstruction.

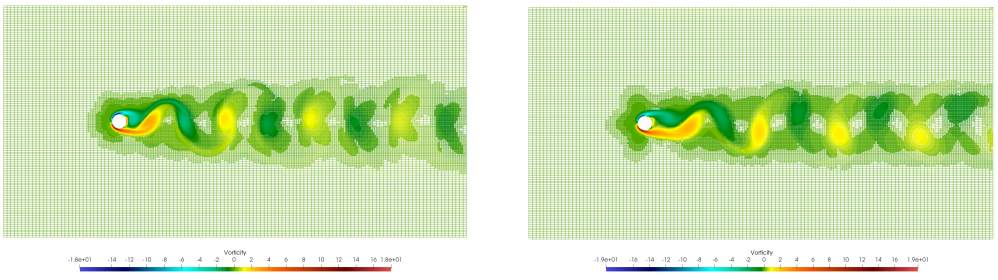


Figure 2.18: Visualization of the vorticity for CFL = 10. The left is the original simulation from Figure 2.16, and the right is the improved departure point reconstruction.

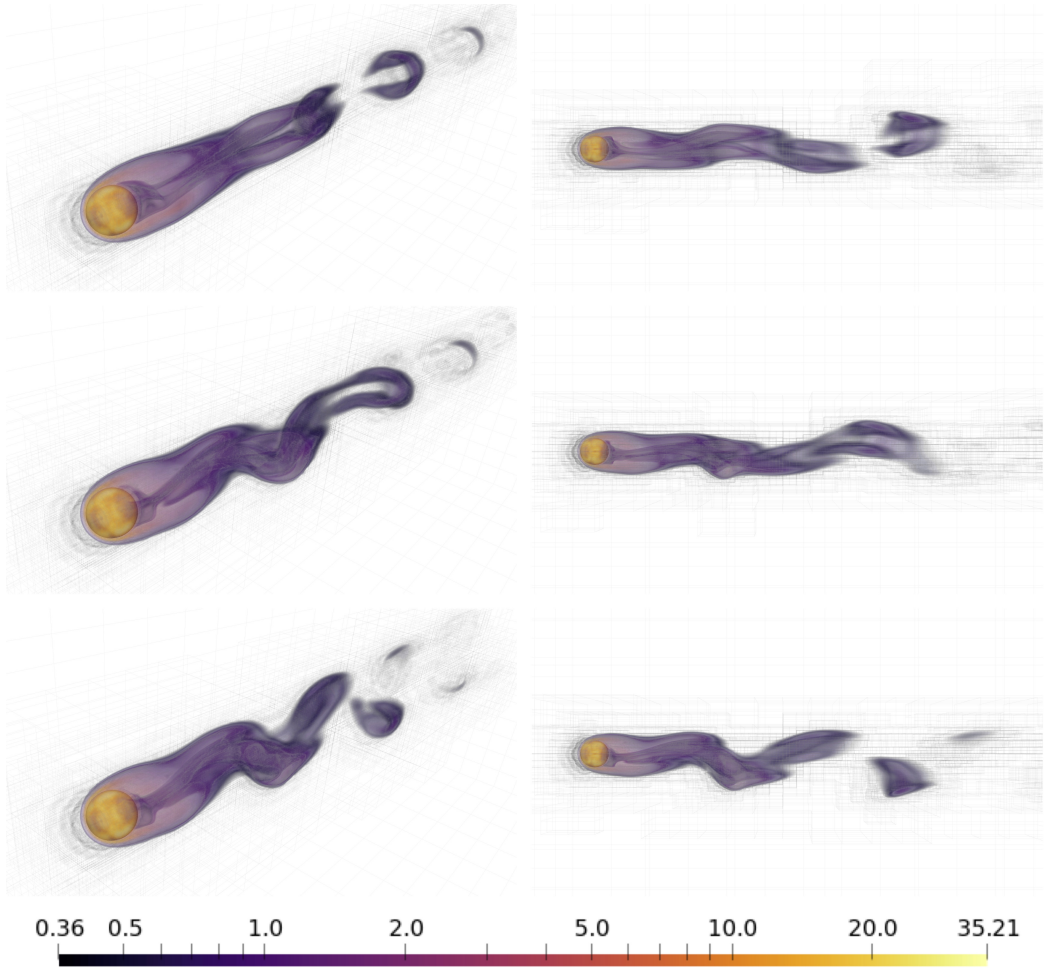


Figure 2.19: Visualization of flow past a sphere for $Re = 350$ at several times. The vorticity and grid are shown.

2.5.4 Flow Past Beginnings

We conclude this section by simulating a high Reynolds number flow past the UC Merced Beginnings sculpture. The model of the sculpture was placed in the domain of $\Omega = [-12.2\,\mathrm{m}, 61.0\,\mathrm{m}] \times [-24.4\,\mathrm{m}, 24.4\,\mathrm{m}] \times [0.0\,\mathrm{m}, 36.8\,\mathrm{m}]$, with minimum and maximum octree refinement levels of 1 and 11, respectively. The Beginnings sculpture is approximately $12.2\,\mathrm{m}$ $(40\,\mathrm{ft})$ in height and we center the sculpture at the point, $(x, y, z) = (0, 0, 0)$. We set the free-stream velocity on the boundary of the domain to $u_0 = 21.5\,\mathrm{m\,s^{-1}}$ $(48\,\mathrm{mph})$, which represents a windy day at UC Merced. The density of the fluid in the simulation is set to $\rho = 1.3\,\mathrm{kg\,m^{-3}}$ and the viscosity to $\mu = 1.8 \times 10^{-5}\,\mathrm{kg\,m^{-1}\,s^{-1}}$, which corresponds to a Reynolds number of $Re = 1.87 \times 10^7$. The velocity at the bottom wall of the domain, z_{min}, is set to 0 and we set the outflow wall of the domain, x_{max}, to a pressure value of 0. A schematic of the problem setup is shown in Figure 2.20.

This example demonstrates the ability of our solver to simulate a high Reynolds number flow around an irregular geometry representing a real-world object, akin to a real world science and engineering problem. Using a 40-core machine, our nodal projection method takes approximately 560 seconds per time iteration with approximately 3 325 000 nodes using the default values for K_{max} and error thresholds. Figure 2.21, taken at $t = 3\,032$ seconds, demonstrates that our approach can resolve a realistic flow field around the Beginnings statue, including the boundary layer while maintaining stability and computational efficiency.

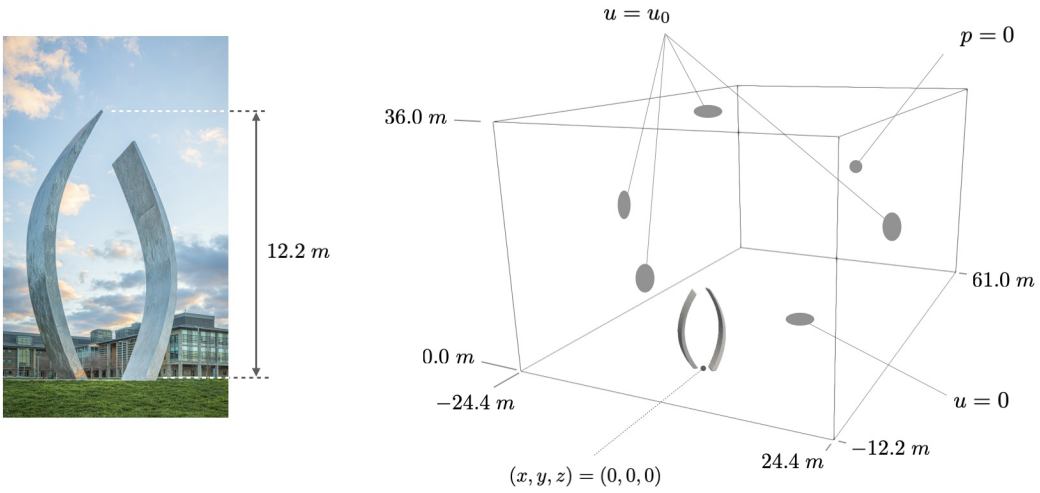


Figure 2.20: Problem schematic for the flow past Beginnings example with a photograph of the statue at UC Merced shown on the left.

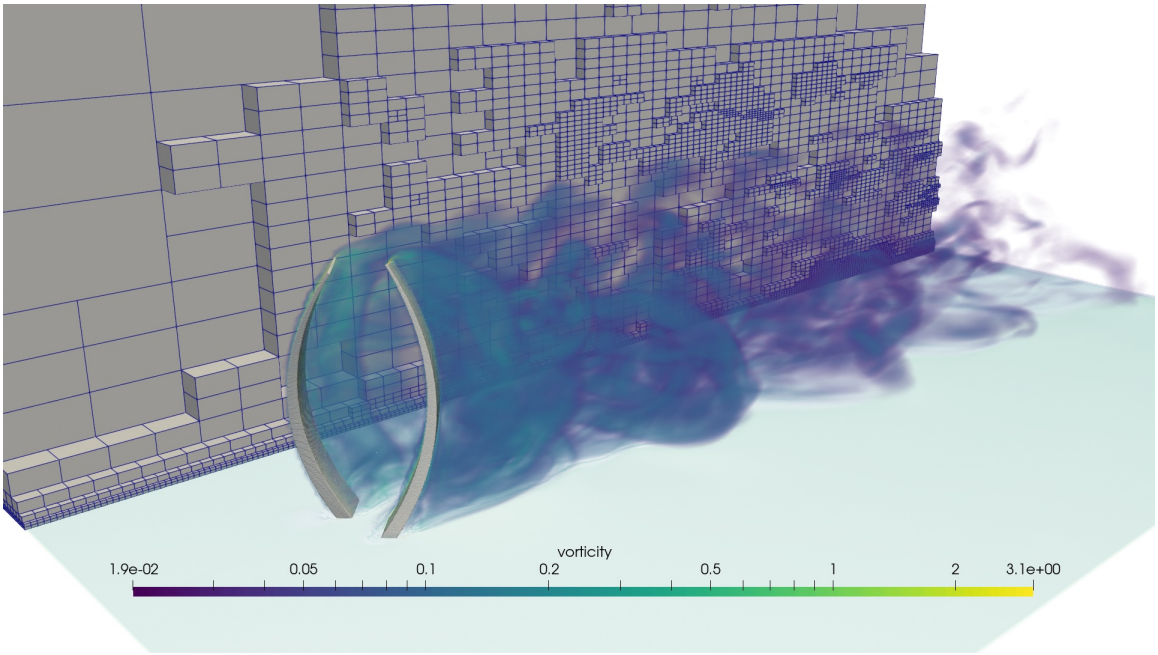


Figure 2.21: Visualization of the flow past the UC Merced Beginnings sculpture for $Re = 1.87 \times 10^7$.

2.6 Conclusions

We presented a novel projection method for the simulation of incompressible flows in arbitrary domains using quad/octrees, where all of the variables are collocated at the grid nodes. By design, our collocated projection operator is an iterative procedure. If it exists, the limit of this iterated procedure is the canonical projection on the incompressible space (Section 2.3.1). Leveraging the connection between the collocated and MAC layouts, we found a technical sufficient condition for this limit to exist on periodic grids. We then verified the stability and convergence of our projection operator numerically in the presence of various boundary and interface conditions. Next, we validated our nodal Navier-Stokes solver using canonical two- and three-dimensional examples and see that our collocated solver achieves high-order convergence (Section 2.4.2), replicates experimental data (Section 2.5.2), and can accurately represent standard computational results with CFL numbers far greater than 1 (Section 2.5.3). Finally, we showed that our solver is well suited to study real-world science and engineering problems by simulating high Reynolds number flows past arbitrary geometries (Section 2.5.4). Our nodal projection method has proven to be a competitive computational fluid dynamics tool for

studying complex fluid flows.

However, we see room for further exploration and improvement at both theoretical and computational levels. Although we derived a technical sufficient condition for the stability of our projection operator, we could not formally prove its stability for our specific ghost node construction and left the boundary effects out of our analysis. Therefore, we relied on an in-depth computational verification process. A more detailed theoretical study could provide information to optimize the construction of collocated operators. Furthermore, our implementation was parallelized using a shared-memory framework. To further extend the capabilities of our nodal solver, we aim to expand it to a distributed memory framework, as this significant enhancement will provide stronger scalability and enable us to investigate a broader range of applications.

To conclude, we emphasize that our nodal projection method uses a fully collocated data layout, drastically reducing implementation costs. The results presented herein demonstrate that our method is a suitable foundation to study a wide range of fluid flows, and we believe that this work can be readily extended to solve more complex phenomena, such as free-surface and multi-phase flows. We foresee our nodal projection method leading to the development of a new class of collocated computational tools capable of accelerating scientific discoveries by lowering accessibility barriers.

Chapter 3

Sharp Collocated Projection Method for Immiscible Two-Phase Flows

3.1 Introduction

Two-phase fluid flows are ubiquitous in nature, and understanding their behavior is critical to solving numerous challenges in both scientific and engineering contexts. These challenges range from phase change heat transfer at the microscale [30, 85, 170] to modeling the transport of oil plumes in oceanic environments [29, 88, 109, 154]. For many such applications, physical experimentation can be prohibitively expensive, making computational methods the preferred tool for investigation [19, 89, 119, 182, 185]. However, the tightly coupled dynamics in two-phase flows make numerical modeling difficult, especially when large interface deformations or adaptive mesh refinement (AMR) are required.

There are two general approaches to modeling fluid interfaces on Eulerian meshes: body-fitted and non-body-fitted methods. Body-fitted methods align the computational mesh to the fluid interface and may use either structured or unstructured grids. These techniques often yield high accuracy but become computationally expensive for complex or moving geometries that require frequent remeshing. Strategies such as arbitrary Lagrangian–Eulerian (ALE) methods [74] help mitigate these costs by allowing the mesh to deform with the fluid and modern ALE variants can handle significant interface motion with high accuracy (see for example [42, 12]). For the work presented herein, we adopt a non-body-fitted approach and focus on those methods for the remainder of this chapter.

Non-body-fitted methods maintain a background mesh, typically Cartesian, and represent the fluid interface either implicitly or with an overlapping grid. The overlapping

grid may be explicit or implicitly defined (*e.g.*, the implicit mesh discontinuous Galerkin framework in [145, 146]). Among the earliest of these is the immersed boundary method (IBM) introduced by Peskin [131, 132], which uses interface-localized forcing terms to impose fluid boundary conditions. This simplicity led to wide adoption [179], but also introduces numerical diffusion near the interface. Higher-order extensions to IBM exist [81, 91, 116, 150], though they often require careful treatment of boundary conditions.

A different class of methods incorporates the interface more directly into the discretization of the governing equations. These include cut-cell finite volume methods and sharp interface methods [4, 37, 174, 173, 172, 186, 59, 58, 108, 121]. Such methods typically rely on explicit interface tracking schemes like front tracking [175], volume-of-fluid (VOF) [73], or level set methods [126]. Cut-cell methods compute intersections between the fluid interface and background grid to create partially filled "cut" cells, while sharp interface methods integrate the interface dynamics directly into the numerical stencil. Cut-cell methods are often used with finite volume or discontinuous Galerkin schemes [14, 64, 38, 68, 145, 146], whereas sharp interface methods are more common with finite difference methods. In this work, we extend the node-based framework of Chapter 2 and [20], which employs a finite difference approach with a sharp interface formulation.

Another consideration for developing grid-based numerical methods is the choice of variable arrangement. A collocated variable arrangement is often preferred from an algorithmic and data structures perspective as this layout can greatly simplify the design of numerical discretizations, especially in the adaptive mesh refinement (AMR) setting (*e.g.*, see the commentary in [9]). Unfortunately, the collocated layout does not naturally lead to a discretization that preserves the analytical properties of the divergence, gradient, and Laplacian operators with standard stencils (*e.g.*, the discrete Laplacian is no longer a composition of the discrete gradient and divergence with central differencing). While there are a number of examples that overcome these challenges (*e.g.*, the work in [112, 37]), using staggered grids, for example the Marker-and-Cell (MAC) layout in [69], naturally preserves the analytic structure of the divergence and gradient operators using standard central differencing. Other staggered layouts, such as the adaptive node-based grid in [66], require more involved discretization stencils, but still preserve the analytic structure of the operators. For these reasons, staggered grids have been commonplace for projection-based Navier-Stokes solvers since the pioneering work of Chorin [35]. For the work herein, we chose to prioritize the algorithmic simplicity of a collocated variable arrangement and extend the node-based framework introduced in Chapter 2 and [20] to the two-phase problem.

We present a novel computational approach to solve the two-phase incompressible Navier–Stokes equations using a fully collocated, adaptive, and sharp-interface

discretization. The key contribution of this work lies in demonstrating how a fully collocated, node-based AMR framework can be leveraged to develop and couple sharp interfacial discretizations in a systematic and extensible manner. By working entirely within a node-based formulation, we are able to naturally construct a hybrid finite difference–finite volume discretization for interfacial problems without the complex interpolations often required in staggered grid layouts. This collocated setting yields simplified data structures and enables the direct design of stable projection-based algorithms on adaptive meshes.

The remainder of this chapter is structured to emphasize the theoretical foundations, the numerical implementation, and the practical performance of the proposed method. Section 3.2 introduces the governing equations, our approach to multiphase interface representation, and the general projection framework. In Section 3.3, we detail the novel spatial and temporal discretization strategies that form the core contribution of this work. Section 3.4 presents verification and validation studies in both two and three dimensions, showcasing the method's robustness in handling rising bubbles, interface deformation, and flow past solid obstacles. Finally, Section 3.5 summarizes our findings and outlines future directions, including extensions to non-Newtonian fluids and multiphysics coupling.

3.2 Background

3.2.1 Governing Equations

We consider a two- or three-dimensional domain, $\Omega = \Omega^+ \cup \Omega^-$, consisting of two incompressible, immiscible, Newtonian fluids each with constant density, $\rho^\pm$, and viscosity, $\mu^\pm$. The two fluids are separated by a closed interface, Γ, with curvature, κ, and surface tension, γ. We denote the fluid and fluid properties within the closed interface using the $+$ superscript (*e.g.*, Ω^+, ρ^+, μ^+) and the fluid and fluid properties outside of the closed interface with the $-$ superscript (*e.g.*, Ω^-, ρ^-, μ^-). The properties defined on the interface (*e.g.*, Γ, κ, γ), do not have a superscript. A schematic of the problem is given in Figure 3.1.

The fluid velocity and pressure are modeled by the incompressible Navier-Stokes equations[1],

$$\left.\begin{aligned} \rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) &= -\nabla p + \nabla\cdot\sigma + \mathbf{f} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \forall \mathbf{x} \in \Omega \setminus \Gamma, \tag{3.1}$$

[1]Note, we have chosen to omit the $\pm$ on the fluid variables for clarity of presentation.

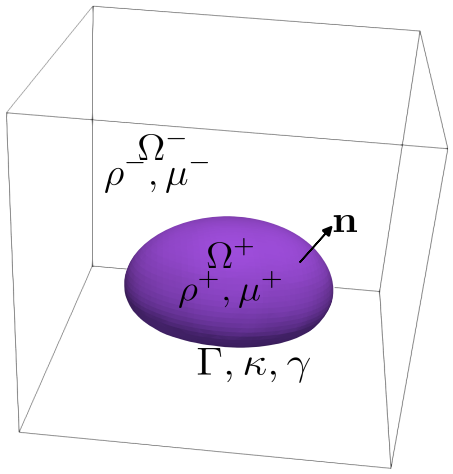


Figure 3.1: Problem schematic in 3D

subject to the interfacial jump conditions,

$$\left.\begin{aligned} [\![\mathbf{u}]\!] &= 0 \\ [\![\sigma \cdot \mathbf{n} - p\mathbf{n}]\!] &= \gamma\kappa\mathbf{n} + \mathbf{q} \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma, \tag{3.2}$$

where $\mathbf{u}$ is the fluid velocity, $\mathbf{n}$ is the interface unit normal (positively oriented from Ω^+ to Ω^-), p is the pressure, and $\sigma = \mu\left(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}}\right)$ is the viscous stress tensor. We use $\mathbf{f}$ to capture any body forces acting on the fluid and, similarly, $\mathbf{q}$ to represent any additional interfacial stress terms. The jump operator, $[\![\cdot]\!]$, is defined as $[\![\chi]\!] = \chi^+ - \chi^-$ and represents the jump of the quantity χ across the interface Γ. Finally, we denote the boundary conditions on the computational domain, $\partial\Omega$, using $\mathbf{B}\left(\mathbf{u}, p\right)$.

3.2.2 Interface Representation

We represent the fluid interface that separates the two immiscible fluids using a level set formulation, where the interface is defined as the zero-contour of a scalar function,

$$\phi(\mathbf{x}) = 0 \qquad \forall \mathbf{x} \in \Gamma. \tag{3.3}$$

Following the traditional formulation [126, 152, 125], the level set function, $\phi(\mathbf{x})$, is constructed as a signed distance function, where $\phi(\mathbf{x}) > 0$ when $\mathbf{x} \in \Omega^+$ and $\phi(\mathbf{x}) < 0$ when $\mathbf{x} \in \Omega^-$ (see Figure 3.1). With this formulation, we can compute the normal, $\mathbf{n}$, as

$$\mathbf{n} = \frac{\nabla\phi}{|\nabla\phi|}. \tag{3.4}$$

Furthermore, we can compute the curvature of the interface, κ, as

$$\kappa = \nabla \cdot \mathbf{n} = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}. \tag{3.5}$$

We further discuss the evolution of the interface using the fluid velocity in Section 3.2.3.

3.2.3 General Two-Phase Projection Method

The general projection approach for solving the two-phase, incompressible Navier-Stokes equations is an extension of Chorin's pioneering work in [35], that incorporates the movement and dynamics of the fluid interface. Our approach, more specifically, follows the work in [167], where a pressure guess is incorporated into the algorithm to improve the accuracy and stability of the numerical method. We summarize the general algorithm in Figure 3.2 and provide details for each of the steps in the following subsections. Additionally, in Section 3.2.4 we provide details for the data structures, interpolations schemes, and adaptive mesh refinement strategy for our quad/octree grids.

Pressure Guess

Pressure correction schemes that either use a guessed pressure or the pressure from a previous timestep are fairly common for the numerical solution of the incompressible Navier-Stokes equations [177, 11, 130]. In the context of multiphase Navier-Stokes equations, such schemes often lead to improved accuracy and stability, and can even relax time step restrictions imposed by capillary forces (see [167] and [147] for a detailed discussion of these benefits). Following [167], we construct our pressure guess by solving an interfacial Poisson problem.

$$\boldsymbol{\Delta} \widetilde{p} = 0 \qquad \forall \mathbf{x} \in \Omega \setminus \Gamma, \tag{3.6}$$

$$\left.\begin{aligned} [\![\widetilde{p}]\!] &= -\gamma\kappa - \mathbf{q} \cdot \mathbf{n} \\ \left[\!\left[\frac{1}{\rho}\mathbf{n} \cdot \nabla \widetilde{p}\right]\!\right] &= 0 \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma, \tag{3.7}$$

subject to the homogeneous Dirichlet boundary conditions,

$$\mathbf{B}(\widetilde{p}) = 0 \qquad \forall \mathbf{x} \in \partial\Omega. \tag{3.8}$$

0. Initialization Initialize the corrective terms $\mathbf{X}_k$ and $\mathbf{c}$ for the intermediate velocity field.

1. Pressure Guess Construct the pressure guess $\widetilde{p}$ as the solution of the Poisson problem

$$\left.\begin{aligned} \boldsymbol{\Delta}\widetilde{p} &= 0 \\ \left[\left[\widetilde{p}\right]\right] &= -\gamma\kappa - \mathbf{q}\cdot\mathbf{n} \\ \left[\left[\frac{1}{\rho}\nabla\widetilde{p}\cdot\mathbf{n}\right]\right] &= 0 \end{aligned}\right\} \quad \begin{aligned} &\forall \mathbf{x}\in\Omega\setminus\Gamma \\ &\forall \mathbf{x}\in\Gamma \end{aligned}$$

2. Repeat until convergence

a) Viscosity step Compute the intermediate velocity field $\mathbf{u}_k^*$ as the solution of

$$\left.\begin{aligned} \rho\frac{D\mathbf{u}_k^*}{Dt} &= -\nabla\widetilde{p} + \nabla\cdot\sigma_k^* + \mathbf{f} \\ \mathbf{B}(\mathbf{u}_k^*) &= \mathbf{B}(\mathbf{u}^{n+1}) + \mathbf{c} \end{aligned}\right\} \quad \forall \mathbf{x}\in\Omega\setminus\Gamma$$

$$\left.\begin{aligned} \left[\left[\mathbf{u}_k^*\right]\right] &= \mathbf{X}_k \\ \left[\left[\sigma_k^*\cdot\mathbf{n}\right]\right] &= \mathbf{q} - \mathbf{n}(\mathbf{q}\cdot\mathbf{n}) \end{aligned}\right\} \quad \forall \mathbf{x}\in\Gamma$$

b) Projection step Compute the Hodge variable Φ as the solution of

$$\boldsymbol{\Delta}\Phi = \nabla\cdot\mathbf{u}_k^* \qquad \forall \mathbf{x}\in\Omega\setminus\Gamma$$

$$\left.\begin{aligned} [\![\rho\Phi]\!] &= 0 \\ [\![\nabla\Phi\cdot\mathbf{n}]\!] &= 0 \end{aligned}\right\} \quad \forall \mathbf{x}\in\Gamma$$

and use it to project the intermediate velocity on the divergence-free space:

$$\mathbf{u}_{k+1}^* = \mathbf{u}_k^* - \nabla\Phi.$$

Note: The projection step may be iterated. See Section 3.2.3 for details.

Check the convergence criteria
If $\left\|\mathbf{u}^{n+1} - \mathbf{u}_{k+1}^*\right\|_\infty < \epsilon,\ \forall \mathbf{x}\in\partial\Omega$, then $\mathbf{u}^{n+1} = \mathbf{u}_{k+1}^*$. Otherwise, update the corrective terms $\mathbf{X}_{k+1}$ and $\mathbf{c}$ as required.

3. Interface evolution Compute the updated interface, ϕ^{n+1}, by solving the advection problem

$$\frac{\partial\phi}{\partial t} + \mathbf{u}_\Gamma\cdot\nabla\phi = 0 \qquad \forall \mathbf{x}\in\Omega.$$

4. Update mesh Adapt the mesh to ϕ^{n+1} and $\mathbf{u}^{n+1}$ and update all the variables accordingly.

Figure 3.2: Overview of the general projection method to evolve the fluid velocity from time t^n to t^{n+1} for two-phase, incompressible Navier-Stokes equations based on [167].

Viscosity Step

In the viscosity step, we solve the momentum equation of the Navier-Stokes system for the intermediate velocity field, $\mathbf{u}_k^*$, where our pressure guess, $\widetilde{p}$, is used in place of the true pressure,

$$\left.\begin{aligned} \rho\frac{D\mathbf{u}_k^*}{Dt} &= -\nabla\widetilde{p} + \nabla\cdot\sigma_k^* + \mathbf{f} \\ \mathbf{B}(\mathbf{u}_k^*) &= \mathbf{B}(\mathbf{u}^{n+1}) + \mathbf{c} \end{aligned}\right\} \quad \forall \mathbf{x} \in \Omega \setminus \Gamma, \tag{3.9}$$

subject to the interfacial jump conditions,

$$\left.\begin{aligned} [\![\mathbf{u}_k^*]\!] &= \mathbf{X}_k \\ [\![\sigma_k^*\cdot\mathbf{n}]\!] &= \mathbf{q} - \mathbf{n}(\mathbf{q}\cdot\mathbf{n}) \end{aligned}\right\} \quad \forall \mathbf{x} \in \Gamma. \tag{3.10}$$

Here, $\mathbf{B}(\mathbf{u}^{n+1})$ represents the appropriate boundary conditions for the incompressible velocity field at time t^{n+1} and the corrective terms, $\mathbf{X}_k$ and $\mathbf{c}$, follow the work of [167] and [20] to strictly enforce the interfacial jump conditions and the wall boundary conditions, respectively. Briefly, $\mathbf{X}_k$, is designed to satisfy the continuity of velocity across the interface in $\mathbf{u}^{n+1}$ and is iterated such that,

$$\lim_{k\to\infty} \left[\!\left[\mathbf{u}_{k+1}^*\right]\!\right] = 0. \tag{3.11}$$

Note that $\mathbf{u}_{k+1}^*$ represents the intermediate velocity field after the projection step. Similarly, the corrective term, $\mathbf{c}$, is designed to compensate for splitting errors at the boundaries of the computational domain, $\partial\Omega$, and is iterated until,

$$\left\|\mathbf{u}^{n+1} - \mathbf{u}_k^*\right\|_\infty < \epsilon \quad \forall \mathbf{x} \in \partial\Omega. \tag{3.12}$$

We compute the successive corrective terms by

$$\mathbf{X}_{k+1} = \mathbf{X}_k - \omega\left[\!\left[\mathbf{u}_{k+1}^*\right]\!\right] \quad \forall \mathbf{x} \in \Gamma, \tag{3.13}$$

and

$$\mathbf{c} = \mathbf{c} - \omega\left(\mathbf{u}^{n+1} - \mathbf{u}_k^*\right) \quad \forall \mathbf{x} \in \partial\Omega, \tag{3.14}$$

where ω is a small, strictly positive constant. For additional details on why these iteration schemes converge, we refer the reader to [167] (jump condition) and [67] (wall boundary condition).

Projection Step

In this step, we project the intermediate velocity field, $\mathbf{u}_k^*$, onto the space of divergence-free vector fields. Traditionally, this is done by solving for the Hodge variable, Φ, which is the solution to the following interfacial Poisson system,

$$\boldsymbol{\Delta}\Phi = \nabla \cdot \mathbf{u}_k^* \qquad \forall \mathbf{x} \in \Omega \setminus \Gamma \tag{3.15}$$

$$\left.\begin{aligned} [\![\rho\Phi]\!] &= 0 \\ [\![\nabla\Phi \cdot \mathbf{n}]\!] &= 0 \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma \tag{3.16}$$

$$\mathbf{B}(\Phi) = \mathbf{B}(p) \qquad \forall \mathbf{x} \in \partial\Omega. \tag{3.17}$$

Note that the homogeneous jump conditions on the interface are a result of our choice of pressure guess, $\widetilde{p}$ (again, see [167] for details). We can further simplify the computation of Φ by rescaling the system to remove the density, ρ, from the jump condition. We define,

$$\Phi = \frac{1}{\rho}\varphi, \tag{3.18}$$

and the rescaled problem becomes,

$$\frac{1}{\rho}\boldsymbol{\Delta}\varphi = \nabla \cdot \mathbf{u}_k^* \qquad \forall \mathbf{x} \in \Omega \setminus \Gamma \tag{3.19}$$

$$\left.\begin{aligned} [\![\varphi]\!] &= 0 \\ \left[\!\left[\frac{1}{\rho}\nabla\Phi \cdot \mathbf{n}\right]\!\right] &= 0 \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma. \tag{3.20}$$

After solving for φ, we can undo the rescaling and use Φ to project the intermediate velocity field, $\mathbf{u}_k^*$, onto the divergence free space,

$$\mathbf{u}_{k+1}^* = \mathbf{u}_k^* - \nabla\Phi. \tag{3.21}$$

If the convergence criteria noted in the previous section are met, then $\mathbf{u}^{n+1} = \mathbf{u}_{k+1}^*$ and we see the traditional formulation of the Helmholtz-Hodge decomposition for a projection method (*e.g.*, $\mathbf{u}^{n+1} = \mathbf{u}^* - \nabla\Phi$).

As established in our single phase work [20], our projection method is approximate, meaning the resulting velocity field is only divergence free to second-order accuracy. Typically only a single projection is required, but we can iteratively apply the projection operator to the intermediate velocity field to remove additional compressible modes and improve the results in our two-phase solver. When needed, we iteratively project the intermediate velocity field until

$$\left\|\mathbf{u}^{n+1} - \mathcal{P}_N\mathbf{u}^{n+1}\right\|_\infty < \epsilon_i \left\|\mathbf{u}^{n+1}\right\|_\infty, \tag{3.22}$$

or a predefined maximum number of iterations, K_{max}, has been reached. Typically, we set ϵ_i to 10^{-3} and $K_{max} = 3$. A more detailed overview of our projection method can be found in Chapter 2 and [20].

Interface Evolution

After computing the incompressible velocity field, $\mathbf{u}^{n+1}$, we evolve the fluid interface, Γ, by solving the level set equation

$$\frac{\partial \phi}{\partial t} + \mathbf{u}_\Gamma \cdot \nabla \phi = 0 \qquad \forall \mathbf{x} \in \Omega, \tag{3.23}$$

where $\mathbf{u}_\Gamma$ is the velocity of the interface itself. Following [167], we construct the interfacial velocity by first extending the velocity fields in Ω^+ and Ω^- across the interface using the third-order method proposed in [8]. We define the interfacial velocity, $\mathbf{u}_\Gamma$, as the average of the two extensions at the physical location of the interface. Finally, we extend the interfacial velocity to the entire domain using a constant extrapolation. With $\mathbf{u}_\Gamma$ computed, we solve the level set function using the Volume-Preserving Reference Map (VPRM) methodology [164], which is based on the Coupled Level Set Reference Map (CLSRM) technique presented in [13]. We briefly review both methods below and refer the interested reader to [13] and [164] for the details of the CLSRM and VPRM, respectively.

Coupled Level Set Reference Map

The CLSRM method [13] was inspired by the work of [83] and evolves a level set function by considering the deformation of the computational domain due to an advecting velocity, $\mathbf{u}$. Following Theillard [164][2], we define the original domain at time $t = 0$ as $\mathcal{B}_0$ and the deformed domain at time t as $\mathcal{B}(t)$. The motion map, $\chi(t, \mathbf{x}_0)$, is a mapping that takes any point $\mathbf{x}_0 \in \mathcal{B}_0$ to the corresponding point $\mathbf{x}(t) \in \mathcal{B}(t)$,

$$\mathbf{x}(t) = \chi(t, \mathbf{x}_0), \ t \geq 0, \ \ \mathbf{x}_0 \in \mathcal{B}_0. \tag{3.24}$$

The reference map, denoted as $\xi(t, \mathbf{x})$, is the inverse of the motion map, and maps any point $\mathbf{x}(t) \in \mathcal{B}(t)$ to its corresponding point $\mathbf{x}_0 \in \mathcal{B}_0$,

$$\mathbf{x}_0 = \xi(t, \mathbf{x}(t)), \ t \geq 0, \ \ \mathbf{x} \in \mathcal{B}(t). \tag{3.25}$$

[2]Note, see Figure 4.3 for a visual representation of the reference map.

In the CLSRM framework, we evolve the level set function, ϕ, using the reference map, $\xi(t, \mathbf{x})$, by solving the auxiliary advection equation to compute the map from time t to time $t = 0$,

$$\frac{\partial \xi}{\partial t} + \mathbf{u} \cdot \nabla \xi = 0, \ \forall t \geq 0, \ \forall \mathbf{x} \in \mathcal{B}(t), \tag{3.26}$$

where

$$\xi(t = 0, \mathbf{x}) = \mathbf{x}, \ \mathbf{x} \in \mathcal{B}_0. \tag{3.27}$$

We can then reconstruct $\phi(t, \mathbf{x})$ by evaluating the composition of the initial level set function, ϕ_0, and the reference map, $\xi(t, \mathbf{x})$,

$$\phi(t, \mathbf{x}) = \phi_0(\xi(t, \mathbf{x})), \ t \geq 0, \ \ \mathbf{x} \in \mathcal{B}(t). \tag{3.28}$$

This formulation has two primary benefits over advecting the level set function directly. First, the regularity of the advected object is greater using the CLSRM. Provided the velocity field is smooth (*e.g.*, incompressible velocities), the reference map will be smooth whereas a reinitialized level set function will have discontinuities in its derivatives. The second benefit of the CLSRM methodology is that we can avoid reinitializing the level set function. In practice, the level set function only needs to be reinitialized when the reference map is restarted (*i.e.* when the bijectivity of the map is lost numerically, see [13]). This results in a much more accurate representation of the advected interface and a significant reduction in mass loss compared to the traditional level set method.

Volume Preserving Reference Map

The VPRM methodology builds on the CLSRM by incorporating a volume-preserving correction to further reduce mass loss. Specifically, the VPRM method uses the reference map to compute local deviations from the volume-preserving space and then uses these local measurements to correct the map by projecting it onto the space of volume-preserving diffeomorphisms. This results in a volume preserving map that conserves mass with second-order accuracy, even in the presence of inaccuracy in the velocity field or numerical errors associated with the advection scheme.

In practice, the VPRM adds two steps to the CLSRM procedure. First, we compute the correction to the reference map by solving a Poisson problem in the shell, $\mathcal{S}$, prescribed around the interface,

$$-\Delta \lambda = 1 - \det\left(\nabla \xi^*\right) \quad \forall \mathbf{x} \in \mathcal{S} \tag{3.29}$$

$$\lambda(\mathbf{x}) = 0 \qquad\qquad \forall \mathbf{x} \in \partial \mathcal{S}, \tag{3.30}$$

for λ. Here, we use ξ^* to denote the intermediate reference map, which may not be volume preserving. Next, we compute the correction, $\gamma^{-1}(\mathbf{x})$, as

$$\gamma^{-1}(\mathbf{x}) = \mathbf{x} - \nabla\lambda \tag{3.31}$$

and use $\gamma^{-1}(\mathbf{x})$ to construct the volume-preserving map as,

$$\xi(\mathbf{x}) = \xi^* \left(\gamma^{-1}(\mathbf{x})\right). \tag{3.32}$$

A few remarks are in order. First, we adopt the notation of [164] for the presentation of the VPRM for consistency, which allows the interested reader to quickly access specific details of the algorithm in the original work. Second, we solve the Poisson problem for λ using a shell prescribed around the interface. Doing so reduces the computational expense of solving the Poisson problem, but some care is needed in defining the shell (see [164]). Finally, we note that the construction of the volume-preserving map is based on the assumption that the deviations from the volume-preserving space are small. For the examples we consider herein, this assumption remains satisfied.

Pressure Reconstruction

In our formulation of the projection method, pressure is never explicitly computed. If desired, the pressure p in the numerical method can be reconstructed using the pressure guess $\widetilde{p}$, Hodge variable Φ, and $\mathbf{u}^*$, and follows from the choice of temporal discretization. Following [167], which has an identical procedure on a staggered Marker and Cell (MAC) grid layout, the pressure reconstruction formula is

$$p = \widetilde{p} + \frac{\alpha\rho}{\Delta t_n}\Phi - \mu\nabla\cdot\mathbf{u}^*, \tag{3.33}$$

where the coefficient α depends on the adaptive time step and is later defined in Section 3.3.1.

3.2.4 Quad/Octree Grids

Data Structures

As in Chapter 2, our numerical method is implemented on non-graded quadtree (2D) and octree (3D) grids. The computational grid is initialized as a single root cell that represents the entire computational domain and is split into four (2D) or eight (3D) children cells. These children cells are then split in the same manner based on user-specified splitting criteria, with the splitting process continuing recursively on new cells until the splitting criteria are fulfilled.

Nodal Interpolations & Discretizations

The stencils for the discrete nodal divergence, gradient, and Laplacian operators, $\mathcal{D}_N$, $\mathcal{G}_N$, and $\mathcal{L}_N$, respectively, are constructed using the four direct neighbors in 2D (six direct neighbors in 3D) of any given node. In the case where a direct neighbor to the center node does not exist in a particular direction, we follow [114] and define ghost values of desired nodal quantities at these T-junctions to circumvent the lack of direct neighbor. These ghost values are constructed using a third-order accurate interpolation scheme. An example of this scenario in 2D is shown in Figure 2.5. Here, node n_0 does not have a direct neighbor to the right, so we introduce a ghost node n_r on the face delimited by nodes n_{r_t} and n_{r_b}. For any nodal quantity ϕ, sampled at the existing nodes, we calculate a third-order accurate ghost value ϕ_r using the information at n_0, its direct neighbors (*i.e.* n_t, n_b), and at the neighboring nodes n_{r_t} and n_{r_b} as (previously defined by Equation (2.41))

$$\phi_r = \frac{r_b\phi_{r_t} + r_t\phi_{t_b}}{r_t + r_b} - \frac{r_t r_b}{t+b}\left(\frac{\phi_t - \phi_0}{t} - \frac{\phi_0 - \phi_b}{b}\right). \tag{3.34}$$

From this approach, we arrive at the discrete operators (previously defined by Equations (2.43)-(2.45))

$$\mathcal{L}_N\phi\Big|_0 = \frac{2}{r+\ell}\left(\frac{\phi_r - \phi_0}{r} - \frac{\phi_0 - \phi_\ell}{\ell}\right) + \frac{2}{t+b}\left(\frac{\phi_t - \phi_0}{t} - \frac{\phi_0 - \phi_b}{b}\right), \tag{3.35}$$

$$\mathcal{D}_N(u,v)\Big|_0 = \frac{u_r - u_\ell}{r+\ell} + \frac{v_t - v_b}{t+b}, \tag{3.36}$$

$$\mathcal{G}_N\phi\Big|_0 = \left(\frac{\ell}{r+\ell}\frac{\phi_r - \phi_0}{r} + \frac{r}{r+\ell}\frac{\phi_0 - \phi_\ell}{\ell}, \quad \frac{b}{t+b}\frac{\phi_t - \phi_0}{t} + \frac{t}{t+b}\frac{\phi_0 - \phi_b}{b}\right). \tag{3.37}$$

Adaptive Mesh Refinement

As was done in previous studies [67, 167, 13, 164, 20] and in Chapter 2, the quad/octree mesh is dynamically refined near interfaces (*e.g.* level sets or fluid-fluid interfaces) or in areas of interesting dynamics (high velocity or vorticity gradients). Typically, at the completion of each time step, we recursively apply the chosen splitting criteria at each cell. Near interfaces, we split each cell $\mathcal{C}$ if the following criterion is met

$$\min_{n\in\text{nodes}(\mathcal{C})} |\phi(n)| \leq B \cdot \text{Lip}(\phi) \cdot \text{diag}(\mathcal{C}) \quad \text{and} \quad \text{level}(\mathcal{C}) \leq l_{\text{max}}, \tag{3.38}$$

where $\text{Lip}(\phi)$ is an upper estimate of the minimal Lipschitz constant of the level-set function ϕ, $\text{diag}(\mathcal{C})$ is the length of the diagonal of cell $\mathcal{C}$, B is the user-specified width of

the uniform band around the interface, where finer resolution is desired. We use l_{max} as the user-specified maximum grid level. Since the level set used to create the mesh will be reinitialized and $|\nabla\phi| = 1$, we use $\text{Lip}(\phi) = 1.2$.

The refinement based criterion for areas of high velocity gradients is the following

$$\min_{n\in\text{nodes}(\mathcal{C})} \text{diag}(\mathcal{C}) \cdot \frac{\|\nabla \mathbf{u}\|}{\|\mathbf{u}\|_\infty} \geq T_V \quad \text{and} \quad \text{level}(\mathcal{C}) \leq \text{V}_{max}, \tag{3.39}$$

where T_V is the user-specified threshold on the velocity gradient and V_{max} is the maximum grid level allowed for velocity gradient-based refinement. We typically choose the velocity gradient-based refinement maximum level V_{max} to be one level lower than the maximum grid level l_{max} due to the most significant dynamics occurring near the interface.

Similarly, the vorticity refinement is

$$\min_{n\in\text{nodes}(\mathcal{C})} \text{diag}(\mathcal{C}) \cdot \frac{\|\nabla \times \mathbf{u}\|}{\|\mathbf{u}\|_\infty} \geq T_W \quad \text{and} \quad \text{level}(\mathcal{C}) \leq \text{W}_{max}, \tag{3.40}$$

where T_W is the user-specified threshold on the vorticity and W_{max} is the maximum grid level allowed for vorticity-based refinement. As with the velocity refinement, we typically chose W_{max} to be one level lower than the maximum grid level.

Finally, we ensure that a minimum resolution of l_{min} is maintained through the following refinement criterion,

$$\text{level}(\mathcal{C}) \geq l_{\text{min}}. \tag{3.41}$$

If none of these criteria are met, we merge $\mathcal{C}$ by removing all its descendants.

3.3 Numerical Discretizations

In this section, we focus on the development of new temporal and spatial discretizations for the two-phase, incompressible Navier-Stokes equations. Our approach builds on the collocated variable framework introduced in Chapter 2 and [20], leveraging that nodal layout to construct a hybrid finite difference–finite volume scheme that achieves high accuracy with a simplified implementation.

We begin by describing our temporal discretization, which is based on the semi-Lagrangian backward differentiation formula (SLBDF) scheme, and introduce a local temporal limiter designed to stabilize time integration in the two-phase setting. We then present our spatial discretization for the momentum equation, incorporating interfacial jump conditions as defined in (3.10). The collocated layout enables a natural finite volume treatment at the fluid interface, allowing us to discretize the full stress tensor

by directly coupling velocity components. In the remainder of this section, we present convergence results for our coupled jump solver, and in the next section, we evaluate the impact of the temporal limiter within the full Navier-Stokes solver.

3.3.1 Temporal Discretizations

We discretize the momentum equation of the Navier-Stokes system using the semi-Lagrangian, backward differentiation formula (SLBDF) scheme as in [102, 67, 167, 20] and in Chapter 2. Briefly, this scheme discretizes the momentum equation along the characteristics of the velocity field following the semi-Lagrangian (SL) method, treating the non-linear advection term explicitly, and then uses a backward differentiation formula to handle the diffusive terms implicitly. The SLBDF scheme leads to the following discretization of Equation (3.9),

$$\rho\left(\alpha\frac{\mathbf{u}^{*}-\mathbf{u}_{d}^{n}}{\Delta t^{n}}+\beta\frac{\mathbf{u}_{d}^{n}-\mathbf{u}_{d}^{n-1}}{\Delta t^{n-1}}\right)=-\nabla\widetilde{p}+\nabla\cdot\left(\mu\left(\nabla\mathbf{u}^{*}+(\nabla\mathbf{u}^{*})^{\mathrm{T}}\right)\right)+\mathbf{f}, \tag{3.42}$$

where $\Delta t^{n} = t^{n+1} - t^{n}$ and $\Delta t^{n-1} = t^{n} - t^{n-1}$ are the adaptive time steps and the coefficients α and β are defined as

$$\alpha=\frac{2\Delta t^{n}+\Delta t^{n-1}}{\Delta t^{n}+\Delta t^{n-1}},\ \beta=-\frac{\Delta t^{n}}{\Delta t^{n}+\Delta t^{n-1}}. \tag{3.43}$$

Departure Point Computation with Phase Accounting

The terms $\mathbf{u}_{d}^{n}$ and $\mathbf{u}_{d}^{n-1}$ are the velocities $\mathbf{u}^{n}$ and $\mathbf{u}^{n-1}$ evaluated at the departure points $\mathbf{x}_{d}^{n}$ and $\mathbf{x}_{d}^{n-1}$, respectively. The departure points $\mathbf{x}_{d}^{n}$ and $\mathbf{x}_{d}^{n-1}$ correspond to the root of the characteristic curve starting at position $\mathbf{x}$ at time t^{n+1} and ending at time t^{n} and t^{n-1}, respectively. We trace the characteristic curve backward in time using a backward RK2 scheme,

$$\hat{\mathbf{x}}=\mathbf{x}^{n+1}-\frac{\Delta t^{n}}{2}\mathbf{u}^{n+1}(\mathbf{x}^{n+1}), \tag{3.44}$$

$$\hat{\mathbf{u}}=\left(1+\frac{\Delta t^{n}}{2\Delta t^{n-1}}\right)\mathbf{u}^{n}(\hat{\mathbf{x}})-\frac{\Delta t^{n}}{2\Delta t^{n-1}}\mathbf{u}^{n-1}(\hat{\mathbf{x}}), \tag{3.45}$$

$$\mathbf{x}_{d}^{n}=\mathbf{x}^{n+1}-\Delta t^{n}\hat{\mathbf{u}}, \tag{3.46}$$

and

$$\hat{\mathbf{x}}=\mathbf{x}^{n+1}-\Delta t^{n}\mathbf{u}^{n}(\mathbf{x}^{n+1}), \tag{3.47}$$

$$\hat{\mathbf{u}}=\mathbf{u}^{n+1}(\hat{\mathbf{x}}), \tag{3.48}$$

$$\mathbf{x}_{d}^{n-1}=\mathbf{x}^{n+1}-(\Delta t^{n}+\Delta t^{n-1})\hat{\mathbf{u}}. \tag{3.49}$$

Here, the velocity $\mathbf{u}^n$ is evaluated at the intermediate points $\hat{\mathbf{x}}$ using quadratic interpolation. The interpolations of the velocity fields at the interpolation points $\mathbf{x}^{n+1}$ are done using a stable weighted essentially non-oscillatory (WENO) scheme, maintaining a second-order accurate discretization.

While it is noted in [24, 86] that using $\mathbf{u}^n$ to represent the unknown $\mathbf{u}^{n+1}$ is generally an acceptable strategy for SL methods, we used the improved reconstruction scheme from Chapter 2 and [20] to construct the velocity at $\mathbf{u}^{n+1}$. We compute $\mathbf{u}^{n+1}$ using the expansion (previously defined by Equation (2.83)),

$$\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) = \mathbf{u}^n(\mathbf{x}^{n+1}) + \frac{\Delta t^n}{\Delta t^{n-1}}\left(\mathbf{u}^n(\mathbf{x}^{n+1}) - \mathbf{u}^{n-1}(\mathbf{x}^{n+1})\right) + \mathcal{O}(\Delta t^2). \tag{3.50}$$

In Chapter 2 and [20], it was found that this trajectory reconstruction scheme offers improved accuracy and stability at larger CFL number (up to CFL $= 20$). For the tests herein, the improvements are present, but less pronounced than in the single phase study (likely due to the time step restrictions for two-phase systems, see below).

If the departure point originating from one fluid domain crosses into the other fluid domain, we use a simple phase accounting procedure to ensure that the current velocity field is used for interpolation (at the departure point). Using the level set representation of our interface, we check that the phase at the departure point matches the phase from which the trajectory originated. If the phase of the origin point is different than the departure point, the velocity field corresponding to phase of the origin point is used for interpolation of the departure point velocity.

Local Temporal Limiter

While the standard SLBDF scheme is unconditionally stable for the scalar advection-diffusion equation with linear interpolation, this is not true in general (*e.g.*, the analysis of [23, 22] and the references therein). For our work, we use WENO-inspired quadratic interpolation in the construction of our finite difference and finite volume stencils, which can introduce spurious oscillations in regions of high curvature, leading to potential instabilities. To address this, we introduce a local temporal limiter that, when triggered, uses the first-order semi-Lagrangian backward Euler scheme,

$$\rho\left(\frac{\mathbf{u}^{n+1} - \mathbf{u}_d^n}{\Delta t^n}\right) = -\nabla\widetilde{p} + \nabla\cdot\left(\mu\left(\nabla\mathbf{u}^{n+1} + (\nabla\mathbf{u}^{n+1})^T\right)\right) + \mathbf{f}, \tag{3.51}$$

which is unconditionally stable [24, 183, 86]. The threshold for triggering the limiter is computed locally, node by node, considering the quantity derived from (3.42),

$$\mathrm{SL}_{\mathbf{u}} = \left(1 - \frac{\beta}{\alpha}\frac{\Delta t^n}{\Delta t^{n-1}}\right)\mathbf{u}_d^n + \frac{\beta}{\alpha}\frac{\Delta t^n}{\Delta t^{n-1}}\mathbf{u}_d^{n-1}. \tag{3.52}$$

After computing Eq. (3.52) at each node, we compare this value to the maximum and minimum of the velocity at the departure points and check the following criteria

$$\text{if } \text{SL}_{\mathbf{u}} > \text{MAX}(\mathbf{u}_d^n, \mathbf{u}_d^{n-1}) \text{ or if } \text{SL}_{\mathbf{u}} < \text{MIN}(\mathbf{u}_d^n, \mathbf{u}_d^{n-1}). \tag{3.53}$$

If this criteria is met for any velocity component at that particular node, then we use the first-order scheme Eq. (3.51) to ensure stability.

The threshold criterion for the local temporal limiter is designed such that the weighted combination of the departure point velocities does not exceed the maximum or minimum value of either departure point velocity. Using this criterion for the local temporal limiter, we essentially bound the magnitude of the velocity along the characteristic path. In practice, this temporal limiter is rarely activated, but remains a crucial element in ensuring the robustness of our solver. We discuss the practical impacts of the temporal limiter in Section 3.4.1 and note the tests when the limiting criteria is activated (*e.g.*, the rising bubble example in Section 3.4.2).

Time Step Restriction

We chose the time step for numerical integration following the work of [167]. Specifically, we chose the time step as,

$$\Delta t \leq \Delta t_{\text{max}} = \text{MIN}\left(\Delta t_{\text{CFL}}, c_{GV}\Delta t_{\text{GV}}\right). \tag{3.54}$$

The first term in Eq. (3.54) is from the CFL condition,

$$\Delta t_{\text{CFL}} = \frac{c_0 \Delta x}{\max \|\mathbf{u}^n\|_\infty}, \tag{3.55}$$

where c_0 is the pre-set CFL number and Δx is the smallest cell length in the entire quad/octree grid. Note, $\max \|\mathbf{u}^n\|_\infty$ is taken over the entire domain. Given that our method uses a semi-Lagrangian method for handling advection, the CFL number c_0 in general can be chosen larger than 1.

The second term in Eq. (3.54) is a generalization of the time step restriction of Galusinski and Vigneaux [56] presented in [167] for two-phase problems with different density and viscosity in each phase. We compute Δt_{GV} as

$$\Delta t_{\text{GV}} = \frac{c_1 \mu \Delta x}{\gamma}\left(1 + \sqrt{1 + \frac{c_2 \rho \Delta x \gamma}{2\pi c_1^2 \mu^2}}\right), \tag{3.56}$$

where c_1 and c_2 are constant pre-set coefficients that typically are set such that $0 < c_1, c_2 < 1$. This can be seen as an extension of the traditional Brackbill stability

condition in [25] that scales linearly with Δx for viscous flows and significantly relaxes the time step restriction in AMR settings.

We further control the maximum allowable time step by multiplying Δt_{GV} by a coefficient, c_{GV}, in Eq. (3.54). In [56], the authors note that their restriction is only stable for small enough Reynolds numbers where the flow can be considered laminar. We find that controlling this restriction via an additional coefficient improves the range of flow dynamics we can consider. In practice, we determined that choosing $c_{GV} \in (0, 1]$ is a good starting point and ensures numerical stability for all of the examples we tested. Choosing $c_{GV} > 1$ can be done and we do show examples in Section 3.4.2 where this choice remains stable. However, these instances are likely exceptions to the rule and we refer the interested reader to the analysis in [56] and the commentary in [167] for more details.

3.3.2 Spatial Discretizations

After computing the departure point velocities in Eq. (3.42), we are left to treat the diffusive terms in the momentum equation implicitly. In [167], the authors discretized the implicit, coupled system using a MAC grid and only considered the jump in the normal derivative of $\mathbf{u}^*$. This choice of discretization allowed each of the velocity components to be solved independently, but requires an iterative correction to properly address the tangential derivative. Furthermore, this splitting methodology introduces spurious oscillations near the fluid interface, which can exacerbate any parasitic currents (see [54, 156, 134, 167]).

In contrast to the previous work in [167], our solver uses an entirely collocated variable arrangement, which allows us to fully discretize the stress tensor intuitively. While this does result in a coupled system for the velocity, we no longer require an iterative correction and the resulting solver avoids introducing spurious oscillations from splitting error. We present this novel discretization in the following subsections by first generalizing this problem as an interfacial, Poisson type equation. Next, we detail the discretization of the generalized equation at the interface, including the treatment of the jump conditions. Finally, we provide verification of our coupled jump solver by presenting convergence results for two- and three-dimensional examples, with and without a continuous viscosity field.

Generalized Interfacial Poisson Jump Problem

We can generalize the implicit treatment of the diffusive terms, after the SLBDF discretization, as the following interfacial Poisson system,

$$\eta \mathbf{v} - \nabla \cdot \sigma = \mathbf{r} \qquad \forall \mathbf{x} \in \Omega \setminus \Gamma \tag{3.57}$$

$$\left.\begin{aligned} [\![\mathbf{v}]\!] &= \mathbf{k} \\ [\![\sigma \cdot \mathbf{n}]\!] &= \mathbf{h} \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma \tag{3.58}$$

where

$$\sigma = \mu \left(\nabla \mathbf{v} + (\nabla \mathbf{v})^{\mathrm{T}} \right). \tag{3.59}$$

Here, we introduced the notation change from $\mathbf{u}^*$ to $\mathbf{v}$ for both readability and to generalize our solver for both the vector (implicit viscosity solve) and scalar case (pressure guess, projection, etc.). The coefficient η, the right-hand side, $\mathbf{r}$, and the jump conditions, $\mathbf{k}$ and $\mathbf{h}$, include the appropriate contributions in either case.

While the flux jump, $\mathbf{h}$, is naturally treated with a finite volume discretization (see below), the velocity field jump condition, $\mathbf{k}$, needs special care. We address this by extending the jump throughout the entire domain, thereby removing this interfacial boundary condition. Specifically, we introduce the extension $\widetilde{\mathbf{k}}$ of $\mathbf{k}$ such that $\widetilde{\mathbf{k}}$ satisfies

$$\eta \widetilde{\mathbf{k}} - \mu^- \nabla \cdot (\nabla \widetilde{\mathbf{k}}) = 0 \qquad \forall \mathbf{x} \in \Omega^-, \tag{3.60}$$

$$\widetilde{\mathbf{k}} = 0 \qquad \forall \mathbf{x} \in \Omega^+, \tag{3.61}$$

$$\widetilde{\mathbf{k}} = \mathbf{k} \qquad \forall \mathbf{x} \in \Gamma. \tag{3.62}$$

We then introduce the function $\psi = \mathbf{v} - \widetilde{\mathbf{k}}$ which, by Eqs. (3.57) and (3.58), satisfies

$$\eta \psi - \nabla \cdot \widetilde{\sigma} = \widetilde{\mathbf{r}} \qquad \forall \mathbf{x} \in \Omega \setminus \Gamma \tag{3.63}$$

$$\left.\begin{aligned} [\![\psi]\!] &= 0 \\ [\![\widetilde{\sigma} \cdot \mathbf{n}]\!] &= \mathbf{h} - \left[\!\left[\sigma_{\widetilde{\mathbf{k}}} \cdot \mathbf{n}\right]\!\right] \end{aligned}\right\} \qquad \forall \mathbf{x} \in \Gamma \tag{3.64}$$

where

$$\widetilde{\mathbf{r}} = \mathbf{r} - \mu^- \nabla \cdot ((\nabla \widetilde{\mathbf{k}})^{\mathrm{T}}), \tag{3.65}$$

$$\widetilde{\sigma} = \mu \left(\nabla \psi + (\nabla \psi)^{\mathrm{T}} \right), \tag{3.66}$$

and

$$\sigma_{\widetilde{\mathbf{k}}} = \mu \left(\nabla \widetilde{\mathbf{k}} + (\nabla \widetilde{\mathbf{k}})^{\mathrm{T}} \right). \tag{3.67}$$

Notice that, while this system has the same form as Eq. (3.57) and (3.58), the jump in the velocity field is now homogeneous. This simplification, however, comes at the cost of solving an additional problem, the jump extension problem. The jump extension problem, Eq. (3.60), is a simple (non-interfacial) Poisson problem with a Dirichlet boundary condition prescribed on the interface, which we solve using the supra-convergent[3] Poisson solver of [114].

In addition to solving the jump extension problem, the right-hand side of Eq. (3.63) includes two derivatives of $\widetilde{\mathbf{k}}$. Hence, the solution to ψ will, in general, only be first-order accurate. In practice, small jumps in the velocity do not significantly impact the overall order of convergence for the coupled jump solver. As our intermediate velocity field, $\mathbf{u}^*$, should indeed only have small jumps, we do not see a significant degradation in the accuracy of the full Navier-Stokes solver either.

Coupled Jump Solver

We solve the generalized Poisson system, Eq. (3.63) and (3.64), using a finite volume discretization strategy. Away from the fluid interface, we discretize the Poisson system using a standard scheme with the collocated variable framework presented in Chapter 2 and [20]. At the fluid interface, we use a novel method that discretizes the full stress tensor and naturally treats the interfacial boundary conditions. We present the two-dimensional case in full here and note that the extension to three-dimensions is straightforward.

Consider the control volume C_{ij} centered at the node (i,j) as shown in Figure 3.3. We integrate Eq. (3.63) over the control volume and apply the divergence theorem to obtain

$$\int_{C_{ij}} \eta\psi \, \mathrm{d}V - \int_{\partial C_{ij}} \widetilde{\sigma} \cdot \mathbf{n} \, \mathrm{d}S = \int_{C_{ij}} \tilde{\mathbf{r}} \, \mathrm{d}V, \tag{3.68}$$

where ∂C_{ij} represents the boundary of cell C_{ij}. We further expand this equation by splitting each integral into parts corresponding to the two fluid phases,

$$\int_{C_{ij}^+} \eta\psi \mathrm{d}V + \int_{C_{ij}^-} \eta\psi \mathrm{d}V - \int_{\partial C_{ij}^+} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\partial C_{ij}^-} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S = \int_{C_{ij}^+} \tilde{\mathbf{r}} \mathrm{d}V + \int_{C_{ij}^-} \tilde{\mathbf{r}} \mathrm{d}V, \tag{3.69}$$

where C_{ij}^+ and C_{ij}^- represent the portions of the cell in Ω^+ or Ω^-, respectively. In this form, we can directly incorporate the flux jump condition by observing that

$$-\int_{\partial C_{ij}^+} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\partial C_{ij}^-} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S = -\int_{\partial C_{ij}^+ \backslash \Gamma} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\partial C_{ij}^- \backslash \Gamma} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\Gamma} \widetilde{\mathbf{h}} \mathrm{d}S, \tag{3.70}$$

[3] By supra-convergent we mean that the solution and its gradient are second-order accurate.

where, $\widetilde{\mathbf{h}} = \mathbf{h} - \left[\!\left[\sigma_{\widetilde{\mathbf{k}}} \cdot \mathbf{n}\right]\!\right]$. The resulting equation, and the starting point for our discretization, is then,

$$\int_{C_{ij}^{+}} \eta\psi \mathrm{d}V + \int_{C_{ij}^{-}} \eta\psi \mathrm{d}V - \int_{\partial C_{ij}^{+}\backslash\Gamma} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\partial C_{ij}^{-}\backslash\Gamma} \widetilde{\sigma} \cdot \mathbf{n} \mathrm{d}S - \int_{\Gamma} \widetilde{\mathbf{h}} \mathrm{d}S = \int_{C_{ij}^{+}} \widetilde{\mathbf{r}} \mathrm{d}V + \int_{C_{ij}^{-}} \widetilde{\mathbf{r}} \mathrm{d}V. \tag{3.71}$$

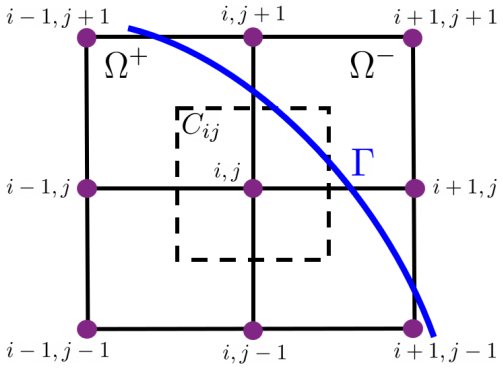


Figure 3.3: Control volume C_{ij} near the interface Γ in the two-dimensional case.

We approximate the volumetric integrals in the standard manner (*e.g.*, [99, 72]). The approximation of the boundary integrals, however, requires some additional work. In order to illustrate this, consider the component-wise expansion of the integrand,

$$\widetilde{\sigma} \cdot \mathbf{n} = \left(\mu\left(\nabla\psi + (\nabla\psi)^{\mathrm{T}}\right)\right) \cdot \mathbf{n} = \mu \begin{pmatrix} 2\left(\psi_1\right)_x n_1 + \left(\left(\psi_1\right)_y + \left(\psi_2\right)_x\right) n_2 \\ \left(\left(\psi_1\right)_y + \left(\psi_2\right)_x\right) n_1 + 2\left(\psi_2\right)_y n_2 \end{pmatrix}, \tag{3.72}$$

where, $\psi = (\psi_1, \psi_2)$ and $\mathbf{n} = (n_1, n_2)$. Additionally, we use the x and y subscripts to denote the appropriate derivatives. We can clearly see the cross terms (*e.g.*, $\left(\psi_1\right)_y n_1$) that result from the discretization of the full stress tensor.

These cross terms lead to two primary challenges. The first is that the discretized momentum equation (3.42) becomes a coupled system and can no longer be solved for each component independently. Furthermore, the coupled linear system is no longer strictly diagonally dominant due to the additional off-diagonal terms. In practice, this difficulty is manageable and we found that the system can be solved using a preconditioned biconjugate gradient algorithm [176].

The second challenge with the cross terms is that these terms are more difficult to discretize because they represent derivatives along the face of the control volume

(orthogonal to the normal) for which they are defined. We address this by taking an average of the derivatives surrounding the appropriate face of the control volume. For example, consider computing the derivative, $(\psi_2)_x$ on the top face of the control volume centered at the node (i, j) (see Figure 3.3). We compute this mixed component derivative as,

$$(\psi_2)_x|_T = \frac{W_L}{2}\left(\frac{(\psi_2)_{i-1,j+1} - (\psi_2)_{i,j+1}}{\Delta x_L} + \frac{(\psi_2)_{i-1,j} - (\psi_2)_{i,j}}{\Delta x_L}\right) + \frac{W_R}{2}\left(\frac{(\psi_2)_{i,j+1} - (\psi_2)_{i+1,j+1}}{\Delta x_R} + \frac{(\psi_2)_{i,j} - (\psi_2)_{i+1,j}}{\Delta x_R}\right) + \mathcal{O}\left(\Delta x^2\right), \tag{3.73}$$

where $\Delta x_L = x_{i,j} - x_{i-1,j}$ and $\Delta x_R = x_{i+1,j} - x_{i,j}$. The weights W_L and W_R are defined as

$$W_L = \frac{\text{Area}(C_R)}{\text{Area}(C_R) + \text{Area}(C_L)}, W_R = \frac{\text{Area}(C_L)}{\text{Area}(C_R) + \text{Area}(C_L)}, \tag{3.74}$$

where C_L and C_R are upper left and upper right cells, respectively. Similarly, we can compute the discretization of the mixed component, $(\psi_1)_y$, on the right wall of the control volume as

$$(\psi_1)_y|_R = \frac{W_T}{2}\left(\frac{(\psi_1)_{i,j+1} - (\psi_1)_{i,j}}{\Delta y_T} + \frac{(\psi_1)_{i+1,j+1} - (\psi_1)_{i+1,j}}{\Delta y_T}\right) + \frac{W_B}{2}\left(\frac{(\psi_1)_{i,j} - (\psi_1)_{i,j-1}}{\Delta y_B} + \frac{(\psi_1)_{i+1,j} - (\psi_1)_{i+1,j-1}}{\Delta y_B}\right) + \mathcal{O}\left(\Delta y^2\right), \tag{3.75}$$

where Δy_T, Δy_B, W_T, and W_B are defined analogously.

We treat the remaining terms (derivatives normal to the face of the control volume) using the second-order difference scheme shown in [20].

Coupled Jump Solver Convergence

We verify the convergence of our two-dimensional coupled jump solver by defining the interfacial jump data, boundary conditions, and right hand side source terms such that the exact solution is

$$\mathbf{v}^-(x, y) = \begin{cases} \sin(xy) \\ xy \end{cases} \tag{3.76}$$

and

$$\mathbf{v}^+(x,y) = \begin{cases} \sin(xy)\cos(xy) \\ y\sin(x) \end{cases} \tag{3.77}$$

in Ω^- and Ω^+, respectively. We define the entire test domain as $\Omega = [-1,1]^2$ and represent the interface between the two subdomains with the level set function,

$$\phi(x,y) = \sqrt{x^2+y^2} - 0.52. \tag{3.78}$$

Given this exact solution, we perform the convergence study with a continuous viscosity, where $\mu^+ = \mu^- = 1.0$ and a discontinuous viscosity, where $\mu^- = 0.5$ and $\mu^+ = 1.0$. The results of this convergence study are shown in Table 3.1.

Table 3.1: Convergence of the coupled jump solver in 2D for continuous and discontinuous viscosity. We use $\mathbf{v}$ for the numerical solution and $\mathbf{v}_e$ to represent the exact solution. Levels refers to the minimum and maximum level of refinement based on the criteria in Section 3.2.4.

Level		$\mu^+ = 1.0,\ \mu^- = 1.0$			
Min	Max	$\|\mathbf{v}-\mathbf{v}_e\|_1$	Order	$\|\mathbf{v}-\mathbf{v}_e\|_\infty$	Order
3	7	2.23e-03	–	3.24e-03	–
4	8	1.04e-03	1.10	1.58e-03	1.04
5	9	3.59e-04	1.53	5.18e-04	1.61
6	10	1.25e-04	1.52	1.77e-04	1.55
7	11	5.10e-05	1.30	7.53e-05	1.23

Level		$\mu^+ = 1.0,\ \mu^- = 0.5$			
Min	Max	$\|\mathbf{v}-\mathbf{v}_e\|_1$	Order	$\|\mathbf{v}-\mathbf{v}_e\|_\infty$	Order
3	7	4.90e-03	–	3.88e-03	–
4	8	2.28e-03	1.11	1.70e-03	1.19
5	9	7.55e-04	1.59	5.77e-04	1.56
6	10	2.69e-04	1.49	1.93e-04	1.58
7	11	1.11e-04	1.27	7.61e-05	1.34

We also perform a convergence study for the coupled jump solver in three-dimensions by considering the analytic solution,

$$\mathbf{v}^- = \begin{cases} \cos(x)\sin(z) \\ \sin(y)\cos(x) \\ \sin(xy) \end{cases} \tag{3.79}$$

Table 3.2: Convergence of the coupled jump solver in 3D for continuous and discontinuous viscosity. We use $\mathbf{v}$ for the numerical solution and $\mathbf{v}_e$ to represent the exact solution. Levels refers to the minimum and maximum level of refinement based on the criteria in Section 3.2.4.

Level		$\mu^+ = 1.0,\ \mu^- = 1.0$			
Min	Max	$\|\mathbf{v} - \mathbf{v}_e\|_1$	Order	$\|\mathbf{v} - \mathbf{v}_e\|_\infty$	Order
1	4	1.64e-03	–	9.29e-03	–
2	5	3.74e-04	2.13	3.18e-03	1.55
3	6	9.71e-05	1.95	1.02e-03	1.64
4	7	2.67e-05	1.86	3.68e-04	1.47

Level		$\mu^+ = 1.0,\ \mu^- = 0.5$			
Min	Max	$\|\mathbf{v} - \mathbf{v}_e\|_1$	Order	$\|\mathbf{v} - \mathbf{v}_e\|_\infty$	Order
1	4	2.23e-03	–	1.26e-02	–
2	5	5.45e-04	2.03	3.25e-03	1.95
3	6	1.67e-04	1.71	1.31e-03	1.31
4	7	6.12e-05	1.45	6.16e-04	1.09

and

$$\mathbf{v}^+ = \begin{cases} \cos(x)\sin(z) + z^2 \\ \sin(y)\cos(x) + xy \\ \sin(xy) + y\sin(x) \end{cases} \tag{3.80}$$

in Ω^- and Ω^+, respectively. We define the whole domain as $\Omega = [-1, 1]^3$ and use a level set representation for the interface with the level set function defined as,

$$\phi(x, y, z) = \sqrt{x^2 + y^2 + z^2} - 0.52. \tag{3.81}$$

As with the two-dimensional example, we use a continuous and discontinuous viscosity with $\mu^+ = \mu^- = 1.0$ and where $\mu^- = 0.5$ and $\mu^+ = 1.0$. The results of the three-dimensional convergence study are shown in Table 3.2.

In both the two- and three-dimensional problems, we see a convergence between first- and second-order accuracy, which is expected based on our treatment of the jump conditions. In the L^1 norm for the three-dimensional case, we see convergence closer to second-order accuracy, however, this is likely an artifact of the adaptive mesh refinement strategy (see Section 3.2.4). It is also important to note that the order of accuracy remains similar for both continuous and discontinuous viscosity coefficients, with a slight

improvement in accuracy for the continuous case. These results show that the novel coupled jump solver will be a robust solution methodology for the full, incompressible two-phase Navier-Stokes solver.

3.4 Results

In this section, we demonstrate the capabilities of the full, incompressible two-phase Navier-Stokes solver. We begin by verifying that our extension of the nodal projection operator in Chapter 2 and [20] remains stable in the presence of interfacial jump conditions. Next, we examine the performance of our novel solver using the standard analytic vortex and parasitic currents examples for two-phase flows (*e.g.*, see [167]). These two examples provide a quantitative measure of the order of accuracy for the two-phase solver and the magnitude of spurious currents driven by the numerical treatment of the fluid interface. We then validate the two-phase solver by simulating oscillating and rising bubbles in two- and three-dimensions. These examples are compared with both analytic and experimental results. Finally, we highlight the capabilities of our novel solver by simulating multiple rising bubbles and interactions between the rising bubbles and solid obstructions.

3.4.1 Verification Examples

Stability of the Projection Step

As noted in Chapter 2 and [20], the numerical stability of a collocated projection operator is not guaranteed for all boundary and interfacial conditions (see the work in [59] and [37]). In Chapter 2 and [20], we demonstrated both analytically and numerically that our collocated projection operator is stable in the presence of an extensive variety of boundary conditions. Here, we extend those verification results and demonstrate the stability of our collocated projection operator in the presence of interfacial jump conditions.

We start by considering the initially divergence-free velocity field,

$$\mathbf{u}^{\pm}(x,y) = \begin{cases} \sin(x)\cos(y) + \mathbf{c} \\ -\cos(x)\sin(y) + \mathbf{c}, \end{cases} \tag{3.82}$$

with the interface, Γ, defined by the level set function,

$$\phi(x,y) = 0.1 - \sin(x)\sin(y). \tag{3.83}$$

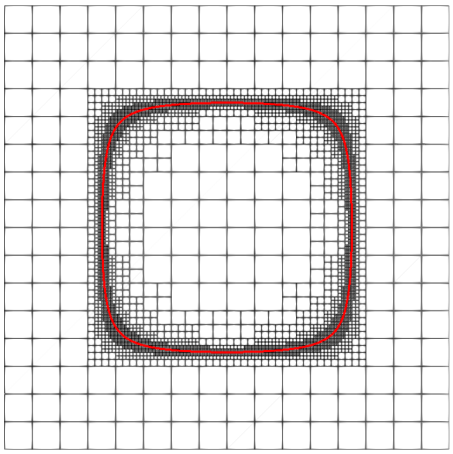


Figure 3.4: Adaptive quadtree grid with a minimum refinement level of 4 and a maximum refinement level of 8. For this example the grid is refined around the interface (in red), with a uniform band set to a minimum of 1 cell.

Here, we use the constant, $\mathbf{c}$, to add a small discontinuity in the velocity field. For the test shown below, we set $\mathbf{c} = 10^{-3}$ in Ω^- and $\mathbf{c} = 0$ in Ω^+. This value is chosen based on a heuristic and matches the threshold we set for the boundary condition iteration described in Section 3.2.3. Additionally, we impose a discontinuity in the density field by setting $\rho^- = 1000$ and $\rho^+ = 1$ in Ω^- and Ω^+, respectively. Finally, we use the computational domain $\Omega = \left[-\frac{\pi}{3}, \frac{4\pi}{3}\right]^2$.

Using an adaptive quadtree grid with a minimum refinement level of 4 and a maximum refinement level of 8 (see Figure 3.4), we repeatedly apply the projection operator to the incompressible field and monitor the variation in the norm of the velocity field at each iteration. Additionally, we perform this test with homogeneous Dirichlet, Neumann, and mixed (*e.g.*, Dirichlet and Neumann) boundary conditions on $\partial\Omega$ applied to Eq. (3.19) and (3.20). As with the single phase operator, a stable projection operator will show the variation in the norm of the velocity field tend to zero. The results of this study are shown in Figure 3.5.

In Figure 3.5, we see that the norm of the velocity field does indeed tend to zero for each of the applied boundary conditions, demonstrating numerical stability. A tolerance of 10^{-12} was used for the linear solvers for this test, and we see that these results match what was presented in our single-phase work (see Chapter 2 and [20]). Note that in practice only a small number of iterations, typically between 1 and 3, of the projection operator are used

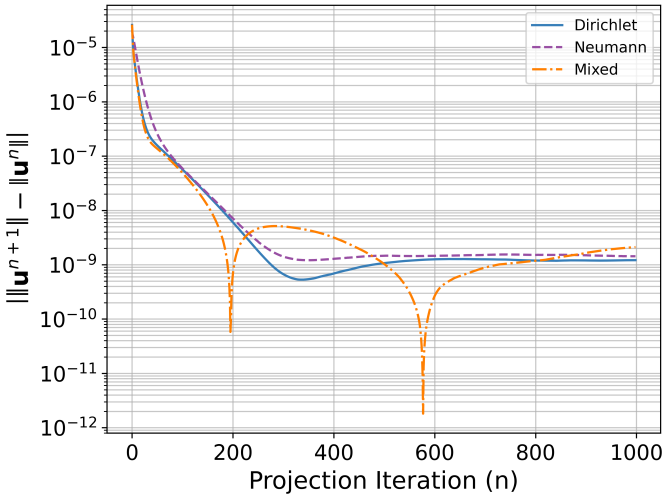


Figure 3.5: Difference between successive projections of an incompressible field, for various boundary conditions on the Hodge variable.

and we show in the next section that we see between first- and second-order accuracy of the full two-phase solver with this approach.

Analytic Vortex

We verify the convergence for the full two-phase solver using the standard, analytic vortex problem (see [67, 167, 20]). We solve the complete incompressible, two-phase Navier-Stokes equations, Eq. (4.38) - (4.39), given the exact analytic solution,

$$\mathbf{u}^{\pm}(x,y,t) = \begin{cases} \sin(x)\cos(y)\cos(t) \\ -\cos(x)\sin(y)\cos(t) \end{cases} \tag{3.84}$$

$$p^{\pm}(x,y,t) = 0, \tag{3.85}$$

with the interface, Γ, defined by the level set function,

$$\phi(x,y) = 0.1 - \sin(x)\sin(y). \tag{3.86}$$

We then set the forcing terms, $\mathbf{f}^{\pm}(x,y,t)$ and $\mathbf{q}(x,y,t)$, as

$$\mathbf{f}^{\pm}(x,y,t) = \rho^{\pm}\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) - \mu^{\pm}\boldsymbol{\Delta}\mathbf{u} \tag{3.87}$$

$$\mathbf{q}(x,y,t) = -\gamma\nabla\cdot\frac{\nabla\phi}{|\nabla\phi|} + \left[\!\left[\mu\nabla\mathbf{u}\cdot\frac{\nabla\phi}{|\nabla\phi|}\right]\!\right]. \tag{3.88}$$

Table 3.3: Convergence of the analytic vortex example with and without the local temporal limiter.

Level		No Limiter			
Min	Max	$\|\mathbf{u}-\mathbf{u}_e\|_\infty$	Order	$\|\phi-\phi_e\|_\infty$	Order
3	7	4.86e-02	–	5.85e-02	–
4	8	1.19e-02	2.03	2.94e-02	0.99
5	9	2.70e-03	2.14	1.56e-02	0.91
6	10	1.51e-03	0.83	8.31e-03	0.91
7	11	1.35e-03	0.56	4.79e-03	0.87

Level		Temporal Limiter			
Min	Max	$\|\mathbf{u}-\mathbf{u}_e\|_\infty$	Order	$\|\phi-\phi_e\|_\infty$	Order
3	7	5.31e-02	–	7.43e-02	–
4	8	1.84e-02	1.53	2.88e-02	1.37
5	9	6.24e-03	1.56	1.39e-02	1.05
6	10	2.76e-03	1.18	7.26e-03	0.94
7	11	1.33e-03	1.05	3.96e-03	0.88

Finally, we set the remaining parameters as,

$$\mu^+, \rho^+ = 1, \qquad \mu^-, \rho^- = 10, \qquad \gamma = 0.1, \qquad \Omega = \left[-\frac{\pi}{3}, \frac{4\pi}{3}\right]^2, \qquad t_{\text{final}} = \pi. \tag{3.89}$$

We prescribe no-slip boundary conditions on the walls of the computational domain and use an adaptive quadtree mesh with a span of 4 between the minimum and maximum refinement levels. For this particular example, we refine only near the interface using the criteria specified in Section 3.2.4 (*i.e.* no vorticity-based refinement is used). The result for increasing levels of refinement with and without the temporal limiter are shown in Table 3.3.

We see in Table 3.3 that our solver yields between first- and second-order accuracy for velocity, with and without the temporal limiter. We also see first-order accuracy in the interface position for both cases, as expected. In both cases, only a single iteration of the projection operator was used at each time step. When the local temporal limiter is applied, we see a slight reduction in the accuracy of the solver on coarse grids. With higher levels of refinement, we no longer see an impact of the local temporal limiter in the accuracy of the velocity. We also note that we see an improvement in the accuracy of the interface position using the temporal limiter. This result should be expected as the first-order time

integration scheme is triggered locally and acts a limiter in the growth of the velocity along the characteristic curve. In both cases, with and without the temporal limiter, our results improve upon those reported in the previous study [167].

We note that the inclusion of the local temporal limiter is to ensure stability in the two-phase solver. In practice, the threshold for the limiter is rarely triggered and typically only when we see large topological deformations in the interface. A more detailed discussion of the examples where the limiter is triggered can be found in Section 3.4.2.

Parasitic Currents

In this example, we examine the performance of our solver using the canonical parasitic currents example [54, 156, 134]. Parasitic currents (also referred to as spurious currents) are non-physical, artificial flows that are generated by the numerical treatment of the interface boundary conditions, especially when surface tension forces are dominant. For this study, we follow [31] and consider an initially circular drop with radius, $R = 0.003$, centered in the domain, $\Omega = [-0.005, 0.005]^2$, with the following fluid parameters,

$$\rho^+ = 1261,\ \rho^- = 1 \quad \mu^+ = 1.4746,\ \mu^- = 1, \quad \gamma = 0.05, \quad \text{and} \quad \Omega = [-0.005, 0.005]^2. \tag{3.90}$$

Additionally, we enforce homogeneous Dirichlet boundary conditions for the velocity on the x-normal walls (*i.e.* $x = -0.005,\ 0.005$) and homogeneous Neumann boundary condition for the velocity on the y-normal walls (*i.e.* $y = -0.005,\ 0.005$). The boundary conditions for pressure are defined as the opposite of the velocity boundary conditions (*e.g.*, Neumann when velocity is Dirichlet and Dirichlet when velocity is Neumann).

For this example, we follow the approach in [167] and start with a uniform grid with a minimum and maximum refinement level of 4. Then, we increment only the maximum level of refinement to create a highly non-graded adaptive mesh. We run the simulation to a time of $t = 1$ and measure the L^∞ error in both the velocity, $\mathbf{u}$, and interface position, ϕ. We select the final time, $t = 1$, based on the work in [167] and note that this is approximately where the decay of the parasitic currents stabilize. The result of this study are shown in Table 3.4 and in Figure 3.6.

Here we see similar results to those reported in [167], with less than first-order accuracy in the velocity error and second-order accuracy in the interface error. The accuracy of the interface error is related to the accuracy of our advection and reinitialization schemes, which are known to be second-order accurate (see [112]). The velocity error for this example corresponds to the magnitude of the parasitic currents (*i.e.* the velocity field should be identically zero). On uniform grids, we would expect the

Table 3.4: Convergence of the parasitic currents example using adaptive grids.

Max Level	$\|u - u_e\|_\infty$	Order	$\|\phi - \phi_e\|_\infty$	Order
4	6.53e-07	–	3.44e-05	–
5	5.08e-07	0.36	7.00e-06	2.30
6	6.48e-08	2.97	1.11e-06	2.66
7	5.86e-08	0.14	2.18e-07	2.35
8	4.73e-08	0.31	5.59e-08	1.96

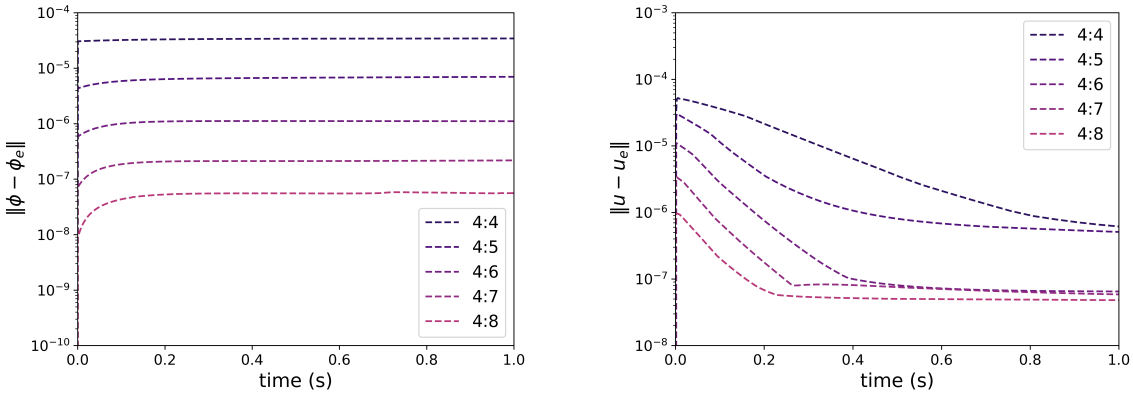


Figure 3.6: Parasitic currents example: interface error (left) and velocity error (right) for increasing maximum grid resolution.

magnitude of the parasitic currents to scale linearly as the dominant contribution to the parasitic currents is the error from approximating the curvature of the interface. In our example, we are increasing the grade of the mesh (the span between the minimum and maximum refinement levels) without increasing the size of the uniform band around the interface (the size of the band is maintained at 1 cell). This decreases the accuracy in the approximation of κ and as a result we expect the order of accuracy for the velocity error to scale sublinearly. We note that the magnitude of our parasitic currents are significantly below the viscous and inertial scaling thresholds presented in [71] (see U_V and U_T in Section 3.2.4) and, for the work presented herein, have no significant impact on the results of our solver.

3.4.2 Validation Examples

Oscillating Bubble

In this next example, we validate our solver using the oscillating bubble problem originally studied by Lamb [95]. We consider an initially circular bubble with radius $R = 1$ and perturb the interface to induce a non-uniform curvature, which generates capillary forces producing oscillations in the velocity field. These oscillations are small and are damped by the viscous effects over time.

We follow [167] and define the perturbed interface for the two-dimensional problem using the level set function,

$$\phi(r, \theta) = -r + \left(R + \frac{\epsilon}{2}(3\cos^2\theta - 1) \right), \tag{3.91}$$

where r is the radial coordinate, θ is the polar angle, R is the initial bubble radius, and $\epsilon = 0.01$ is the perturbation to the radius. We then defined the fluid parameters as,

$$\mu^+ = 0.02, \quad \rho^+ = 1, \quad \frac{\mu^+}{\mu^-} = \frac{\rho^+}{\rho^-} = 10^3, \quad \text{and} \quad \gamma = 0.5. \tag{3.92}$$

Furthermore, we define the computational domain $\Omega = [-1.5, 1.5]^2$ and enforce homogeneous Neumann boundary conditions for the velocity on the x-normal walls (*i.e.* $x = -1.5,\ 1.5$) and homogeneous Dirichlet boundary conditions for the velocity on the y-normal walls (*i.e.* $y = -1.5,\ 1.5$). Boundary conditions for the pressure are defined as the opposite of the velocity boundary conditions (*e.g.*, Neumann when velocity is Dirichlet and Dirichlet when velocity is Neumann).

As with the previous examples, we use a constant minimum refinement level of 4 and incrementally increase the maximum level of refinement from a level of 6 to a level of 9. In Figure 3.7, we plot the radius of the bubble along the x-axis and we compute the period of the oscillations in Table 3.5. We see with this example that the period of oscillation is converging to the theoretical prediction for the three-dimensional problem ($T = 3.629$) and that we have a similar exponential decay to that predicted by Lamb [95] (dotted green line in Figure 3.7). Additionally, when compared to the MAC-based two-phase solver in [167] we see that our coarsest grid (level 4/6) is able to accurately capture the dynamics of the oscillating bubble (see Figure 10 in [167]). In Figure 3.8, we show the velocity profile of this example at the point of maximum expansion and contraction using a minimum and maximum level refinement of 4 and 9, respectively.

Table 3.5: Convergence of period of oscillation for the oscillating bubble example. The 3D theoretical prediction is 3.629.

Max level	Period of oscillation
6	3.834
7	3.774
8	3.706
9	3.693

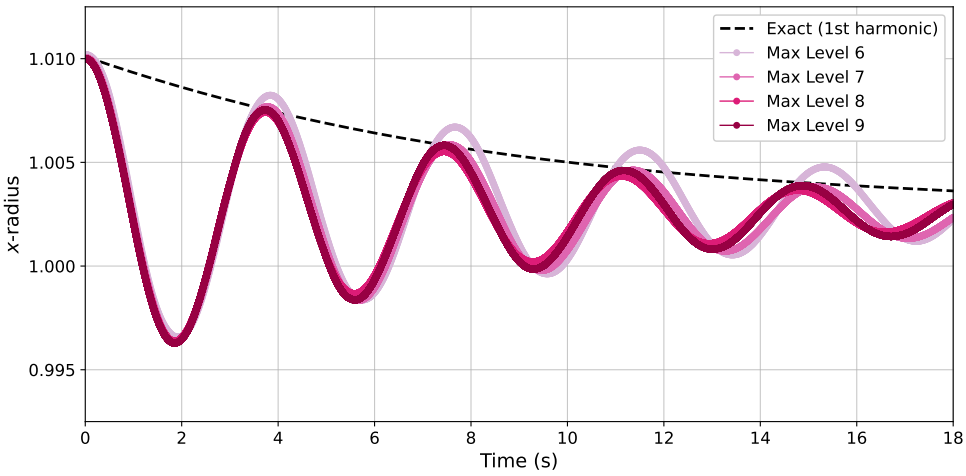


Figure 3.7: Time evolution of the oscillating bubble radius in the x-direction for increasing maximum quadtree refinement levels. The green line indicates the predicted exponential decay of the bubble in 3D.

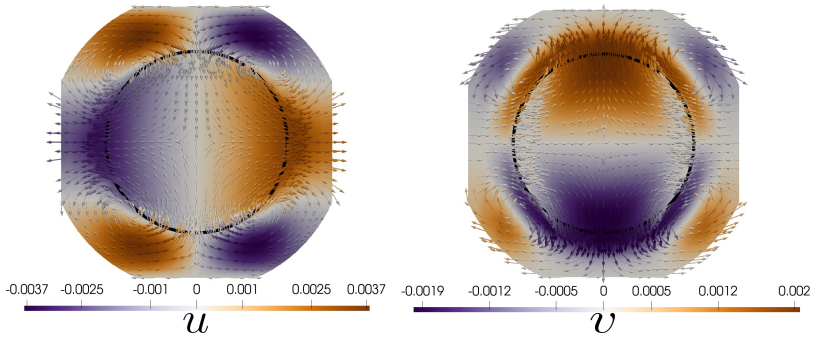


Figure 3.8: Velocity profile of the oscillating bubble at times of maximum expansion and contraction. The colormap shows the horizontal velocity u at time of expansion (left) and vertical velocity v at time of contraction (right). The bubble interface is represented in black.

Dynamics and Deformation of Rising Bubbles

For the remaining examples, we consider the dynamics and deformation of rising bubbles in three-dimensions, subject to strong density and surface tension driven deformation. To start, we consider the Bhaga-Weber experiment of a single rising bubble and validate our solver using cases a through h from [16]. These dynamics are described by the non-dimensional Morton (Mo), Eötvös (Eo), and Reynolds (Re) numbers, defined as

$$Mo = \frac{g(\mu^-)^4}{\rho^-\gamma^3}, \quad Eo = \frac{gd^2\rho^-}{\gamma}, \quad \text{and} \quad Re = \frac{\rho^- U d}{\mu^-}, \tag{3.93}$$

where U is the asymptotic rising velocity measured at the tip of the bubble, g is the acceleration due to gravity, and d is the initial diameter of the undeformed bubble. Cases a through h in [16] are defined using these dimensionless numbers and we summarize these parameters in Table 3.6.

In order to simplify the presentation of our results, we set the rising velocity, U, and

Table 3.6: Definition of the dimensionless parameters, Mo, Eo, and Re, for the Bhaga Weber cases a through h [16].

Case	a	b	c	d	e	f	g	h
Mo	711	711	8.20e-4	266	4.63e-3	8.20e-4	43.1	43.1
Eo	8.67	17.7	32.2	243	115	237	339	641
Re	7.80e-2	2.32e-1	55.3	7.77	94	259	18.3	30.3

the initial bubble diameter, d, to unity (*e.g.*, $U = d = 1$) and then define the remaining simulation parameters as,

$$\rho^- = 1, \quad \frac{\rho^-}{\rho^+} = 10^3, \quad \mu^- = \frac{\rho^-}{Re}, \quad \frac{\mu^-}{\mu^+} = 10^2, \quad \gamma = \frac{(\mu^-)^2}{\rho^-}\sqrt{\frac{Eo}{Mo}}, \tag{3.94}$$

and

$$g = \frac{(Mo)\rho^-\gamma^3}{(\mu^-)^4}. \tag{3.95}$$

We use the computational domain $\Omega = [-16, 16]^3$ and initialize the bubble at $\mathbf{x} = (0, -8, 0)$. We chose the distance from the walls to minimize boundary effects and define no-slip boundary conditions at all but the top wall. On the top wall (*e.g.*, $\mathbf{x} = (x, 16, z)$), we impose a no-flux boundary condition. For each case, we run the simulation until the bubble reaches its asymptotic shape and rising velocity. Similar to the previous two-dimensional studies, we use a minimum refinement level of 4 and refine the mesh using a combination of interface, Eq. (3.38), and vorticity, Eq. (3.40), based refinement. For the vorticity based refinement, we use the following thresholds,

$$T_V = 0.01 \quad \text{and} \quad \max_V = \max_{\text{level}} - 1. \tag{3.96}$$

Figure 3.9 shows the final interface shape of the bubbles along with streamlines of the magnitude of the apparent velocity, $\|\mathbf{u}_a\|_2 = \|\mathbf{u} - \mathbf{u}_{ref}\|_2$, for each experimental case. Here, we use the rising velocity (*e.g.*, the velocity at the tip of the bubble) as the reference velocity, $\mathbf{u}_{ref}$. We observe qualitative agreement with the experiments over the full range of parameters, though we note that the simulated bubbles feature some roughness along their edges as a result of the limited spatial resolution. While we are unable to fully resolve the thin film skirts present in cases (e), (g), and (h) in the asymptotic regime, we still observe the formation of these thin regions as the bubbles rise before the thin regions disappear due to limitations in resolution. Figures 3.10 and 3.11 demonstrate the time evolution for cases (e) and (h), respectively, showing our ability to resolve the more widely varying topological configurations these bubbles take on over time. These results are qualitatively consistent with those presented in other numerical studies [167, 31]. In Table 3.7, we summarize the results of our study.

In each of the cases, we observe the rising velocity is approximately equal to the asymptotic rising velocity, as expected. Additionally, we show that our solver produces a small relative volume loss in comparison to the two-phase, MAC based solver in [167]. We measure the volume loss at the final time as

$$\text{relative volume loss} = \frac{\frac{4}{3}\pi r_0^3 - \int_{\Omega^+} \mathrm{d}V}{\frac{4}{3}\pi r_0^3}, \tag{3.97}$$

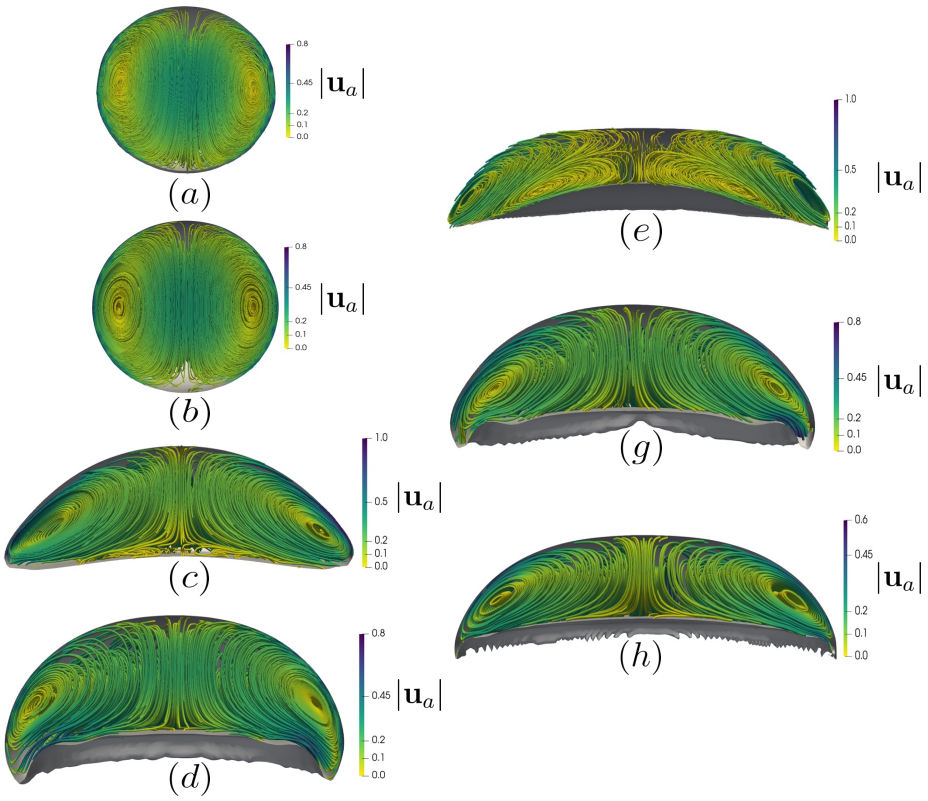


Figure 3.9: Final bubble shapes and interior flow fields corresponding to cases a through h in [16], with exception of case f. The streamlines are colored according to the magnitude of the velocity of the fluid in the reference frame of the rising bubble, $|\mathbf{u}_a|$. For clarity, each bubble is displayed sliced in half.

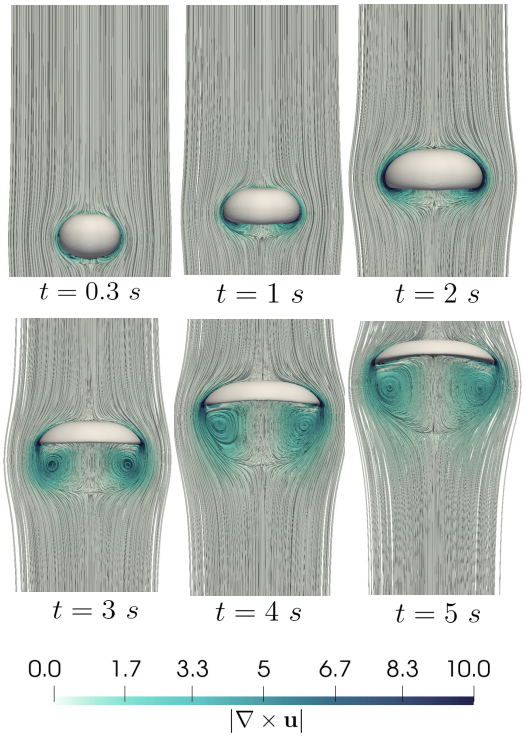


Figure 3.10: Evolution of the rising bubble and apparent exterior flow for case e of [16]. The streamlines in the suspending fluid are colored by the vorticity magnitude, $|\nabla \times \mathbf{u}|$.

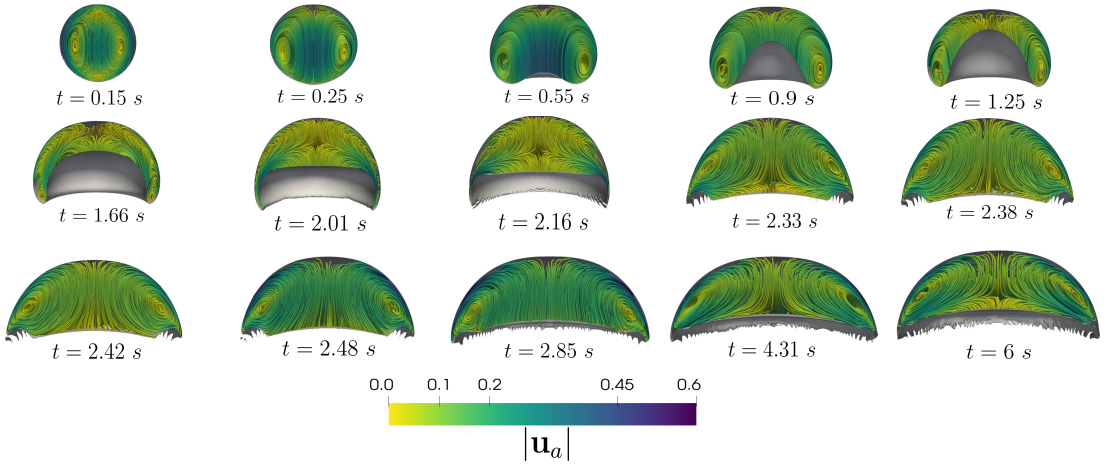


Figure 3.11: Evolution of the rising bubble for case h of [16]. The streamlines are colored by the magnitude of the apparent velocity $|\mathbf{u}_a|$.

where r_0 is the initial radius of the bubble. This result is expected as our solver incorporates the volume-preserving reference map of [164] for the advection of our interface (see Section 3.2.3). Finally, we note that our solver is able to use very large time steps (CFL numbers and GV coefficients for each case are presented in Table 3.7). For cases a through d, we are able to chose quite large time steps without impacting the numerical stability of the two-phase solver. For cases e, g, and h, we chose to use a more restrictive time step based on the discussion in [31] and to ensure that we maintain numerical stability in these more topologically complex cases. We also note that the use of the local temporal limiter (Section 3.3.1) was only triggered in cases e, f, and h . Case f, in particular, exhibited considerably different behavior compared to the other examples, and we provide a detailed analysis for this case below.

Table 3.7: Parameters and results summary for the Bhaga-Weber cases a through h, with exception of case f. The rising velocity is measured at the final time step at the leading edge of the bubble.

Case:	a	b	c	d	e	g	h
Final time (s)	3	3	6	5	6	4	6
Rising velocity	1.16	1.11	1.04	1.08	0.83	0.99	0.9
Relative volume loss	0.2%	0.2%	0.2%	0.19%	0.05%	0.19%	0.05%
Max level	10	10	10	10	11	10	11
CFL number (c_0)	1	1	1	1	0.25	0.45	0.45
GV coefficient (c_{GV})	2	2	2	2	0.3	1	0.3

Analysis of Case f

As noted above, case f exhibited different behavior than the other cases simulated herein and in previous studies, such as [167]. Figure 3.12 shows the early time evolution of the bubble shape and we observe that the middle section of the bubble becomes thin as the bubble rises. At simulation time $t = 0.861s$, the thinning of the tip of the bubble reaches the limit of resolution on our grid and, at this point, the bubble begins to break apart. This is similar to how our solver resolves the thin skirts of cases e and h. Figure 3.13 shows the start of the bubble rupture and Figure 3.14 shows a two-dimensional slice of the level set function and the adaptive mesh. We see in this instance that only a single grid node is resolving the interface at the tip of the bubble. Figures 3.15 and 3.16 show the significant topological changes as the rupture in the bubble continues to expand and leads to breakup.

There are several conclusions to draw from this case. The first is that our solver is capable of simulating significant topological changes and bubble breakup while

maintaining numerical stability. This demonstrates our ability to resolve complex hydrodynamics involving considerable interfacial deformations. On the other hand, this case also highlights several limitations in our solver and computational approach. Case f as defined in [16] has the highest Reynolds number of Re $= 259$, and thus features the most momentum of all of the cases. Given our crude initialization of the bubble shape (perfect sphere) and initial velocity profile (uniformly zero) as well as our neglect of higher order interfacial forces (*e.g.*, Marangoni forces), our simulation is likely under resolved and, hence, does not resemble the experiment from [16] or past numerical studies such as [167]. While we can increase the resolution used for this example to improve the accuracy, we feel that it is instructive to see the limitations of our solver and the types of numerical artifacts it produces when under resolved. Our future goal is to investigate different initial configurations that may lead to a more realistic terminal bubble shape given this resolution.

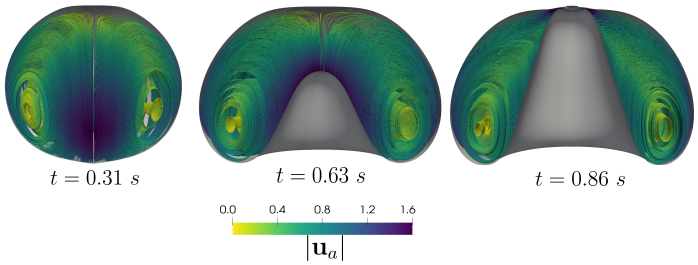


Figure 3.12: Evolution of the instantaneous shape and apparent velocity for case f in [16]. The streamlines are colored by the magnitude of the apparent velocity, $|\mathbf{u}_a|$. The max refinement level of this simulation is 12. For clarity, each bubble is displayed sliced in half.

Multiple Rising Bubbles

To further illustrate the capabilities of our numerical method, we simulate the evolution of 20 rising bubbles using the parameters of case d in [16], initialized in the domain using random positions and initial sizes. We perform this simulation using the minimum and maximum levels of refinement set to 4 and 10, respectively, and define the interface and vorticity refinement thresholds using the same parameters as in the previous section. We show the final configuration of this example in Figure 3.17 and visualize the flow field through streamlines of velocity centered on each bubble, colored by the velocity magnitude. Here, in the zoomed region (a), we see an illustration of some asymmetries in the vertical direction of the flow due to the influence of nearby bubbles. We show that our solver is able to resolve complex multi-bubble interactions in regions where bubbles are close to

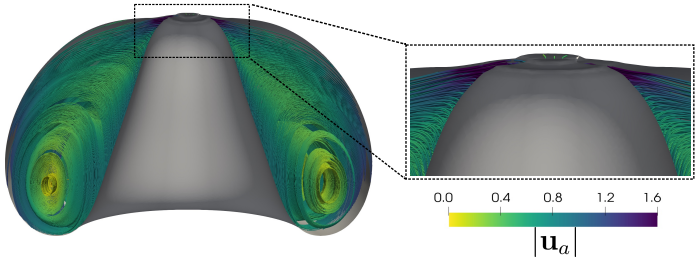


Figure 3.13: Instantaneous shape and apparent velocity for case f in [16] at time $t = 0.861s$, the first time step at which the tip of the bubble begins to break apart. The full bubble is shown (left) along with a zoomed in view of the tip beginning to break apart (right). The streamlines are colored by the magnitude of the apparent velocity, $|\mathbf{u}_a|$. The max level of refinement for this simulation is 12. For clarity, the full bubble is displayed sliced in half.

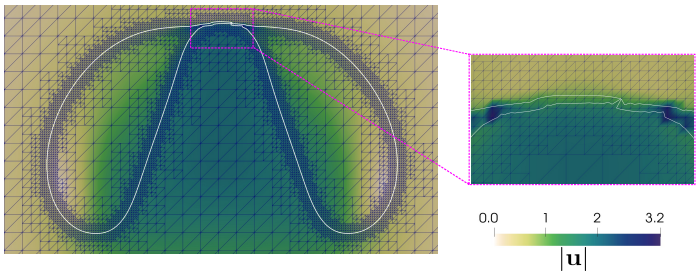


Figure 3.14: Vertical slice in 2D for case f in [16] at time $t = 0.861s$, the first time step at which the tip of the bubble begins to break apart. The full slice is shown (left) along with a zoomed in view of the tip beginning to break apart (right). The adaptive grid is shown and the slice is colored by the velocity magnitude, $|\mathbf{u}|$. The max level of refinement for this simulation is 12.

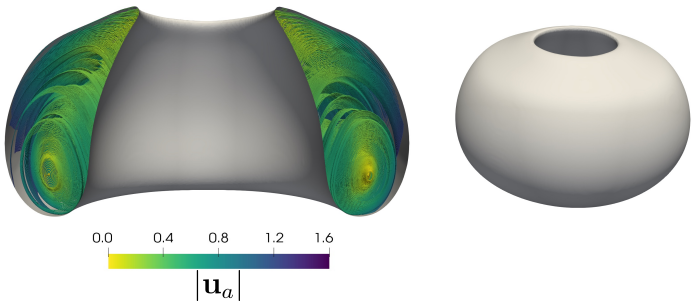


Figure 3.15: Instantaneous shape and apparent velocity for case f in [16] at the time $t = 1.3s$, after the tip has fully broken apart (left). For clarity, the bubble is displayed sliced in half. The streamlines are colored by the magnitude of the apparent velocity, $|\mathbf{u}_a|$. The instantaneous shape of the full bubble is also shown (right), depicting the full breakup of the bubble's tip. The max level of refinement for this simulation is 12.

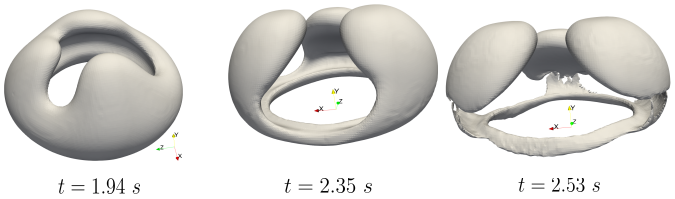


Figure 3.16: Evolution of the bubble shape for case f in [16] for later times after the initial breakup of the tip.

one another, demonstrating the capability of our solver to effectively handle multi-scale hydrodynamics.

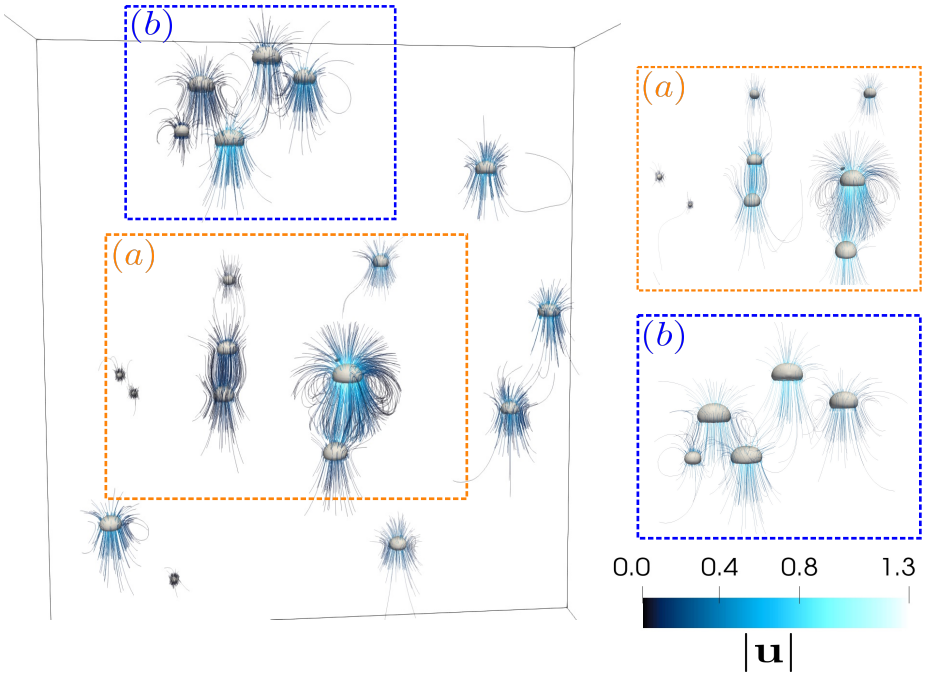


Figure 3.17: Final configuration of 20 rising bubbles with random initial positions and sizes. The flow field is visualized using streamlines colored by velocity magnitude, $|\mathbf{u}|$. Panels (a) and (b) show zoomed-in regions that highlight the complex flow structures where multiple bubbles are in close proximity.

Rising Bubbles in an Inverted Funnel

We conclude this results section by simulating a single rising bubble in a domain that features solid, non-moving obstructions. As in the previous example, we consider a bubble with the parameters of case d from [16], and define solid obstructions using a level set formulation. To treat these objects numerically, we set the velocity field $\mathbf{u} = 0$ in the solid obstruction region.

In Figure 3.18, we show the time evolution of a single bubble rising through an inverted funnel, where the converging conical section leads to a narrow port half the size of the bubble's initial diameter. This results in a large topological deformation of the bubble and provides a means for us to measure the volume preservation of our two-phase solver. We show five snapshots of this example in Figure 3.19, demonstrating how the mesh adapts

around the bubble as it changes its rising velocity and shape. Additionally, we show the flow field around the bubble in Figure 3.20. We measure the volume preservation for this example in Figure 3.21 using Equation (3.97) and observe good conservation throughout the entire evolution.

This example is a simple demonstration of the ability of our solver to simulate a two-phase flow impinging on an arbitrary irregular geometry, while achieving satisfactory volume conservation and maintaining numerical stability. We do not model the dynamics of the contact angle between the wall and the bubble and we are not treating the geometry of the obstruction in a sharp manner. These features, however, can be added to our solver to create a more refined simulation to address a variety of physically motivated problems, such as those mentioned in the introduction.

3.5 Conclusions

We present a novel sharp collocated projection method for solving the immiscible, two-phase Navier-Stokes equations in two- and three-dimensions. Our method is built using non-graded, adaptive quadtree and octree grids, where all of the fluid variables are defined on the nodes of the mesh. This nodal framework allows us to design novel spatial and temporal discretizations that accurately capture the interfacial dynamics present in the two-phase problem, without the need for specialized data structures or an overly complex algorithm.

Our work herein is an extension of the single-phase projection method presented in Chapter 2 and [20] and enhances that framework in a number of ways. First, the projection operator itself has been modified to account for the interfacial jump conditions present in the two-phase problem. Similar to the analysis in the single-phase study, we demonstrate that our two-phase projection operator is stable for a variety of boundary and interfacial jump conditions. Next, we designed a new temporal discretization strategy that leverages phase accounting and a temporal limiter to enhance stability and robustness for the overall algorithm. This strategy ensures that velocities at the semi-Lagrangian departure points are interpolated using the correct field and limits oscillatory growth of the velocity along the characteristic paths. Finally, the key contribution of our work is the hybrid finite volume-finite difference scheme we developed to solve the coupled momentum equation, which treats the interfacial jump conditions in an entirely sharp manner. With this methodology, we see between first- and second-order accuracy for the complete Navier-Stokes solver and, when compared with a similar MAC (staggered) layout, a lower magnitude of total error. We show that our two-phase algorithm is capable of solving complex fluid flows through a number of two- and three-dimensional examples that exhibit large topological deformation

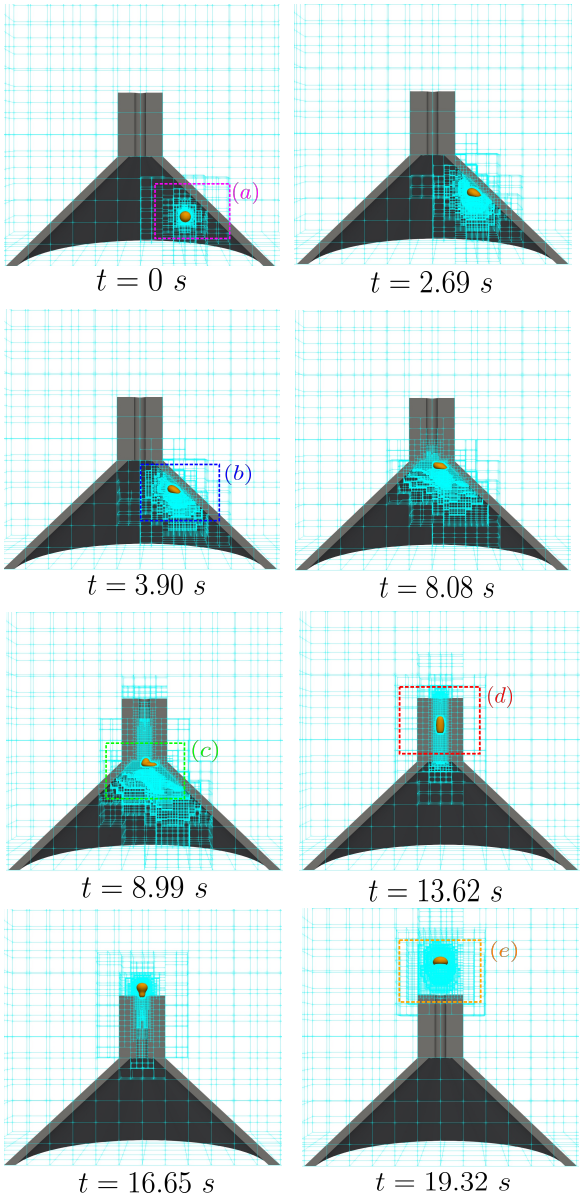


Figure 3.18: Evolution of the shape and adaptive octree grid for a bubble rising in an inverted funnel. For clarity, the flow obstruction is displayed sliced in half. The droplet is shown to maintain its volume as it rises through the complex flow geometry. Five time steps labeled $(a) - (e)$ are identified with boxes and highlighted in Figure 3.19.

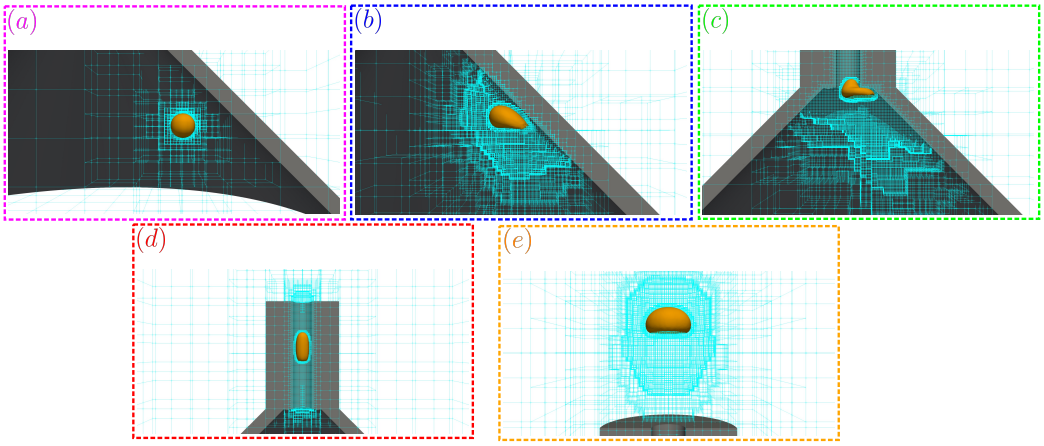


Figure 3.19: Zoomed in snapshots from Figure 3.18.

and interaction with solid obstructions.

We emphasize that the collocated framework employed by our solver dramatically simplifies the design of algorithms and data structures required to implement these numerical methods on non-graded adaptive grids. While this simplification results in the loss of some desirable properties (*i.e.* an orthogonal projection), our method does result in a lower total error than an equivalent MAC-based solver (*e.g.*, see [167]). Furthermore, we show that our framework is naturally extended to more complex physics, making our method well suited, for example, to simulate non-Newtonian multi-phase flows with spatially varying stress. We believe that our solver serves as an ideal tool for scientists and engineers wishing to develop simulations of multi-phase flow applications, due to the combination of its accuracy and its ease of implementation.

We would like to note that there are a number of directions for future work. First, a natural extension of the solver would include contact angle dynamics between bubbles and walls and treat solid geometries in a sharp manner. Next, we wish to expand the capabilities of our solver to handle variable density flows and non-uniform interfacial tension (due to for instance the Marangoni effect), enabling the study of atmospheric and oceanic applications featuring density stratified flows, such as rising oil droplets [29, 88, 154, 109, 107], fine particle pollution in the atmosphere [171], and settling marine snow [135]. Finally, we wish to further enhance the scalability and parallelism of our solver by moving to a distributed memory paradigm (MPI) and leveraging GPU hardware.

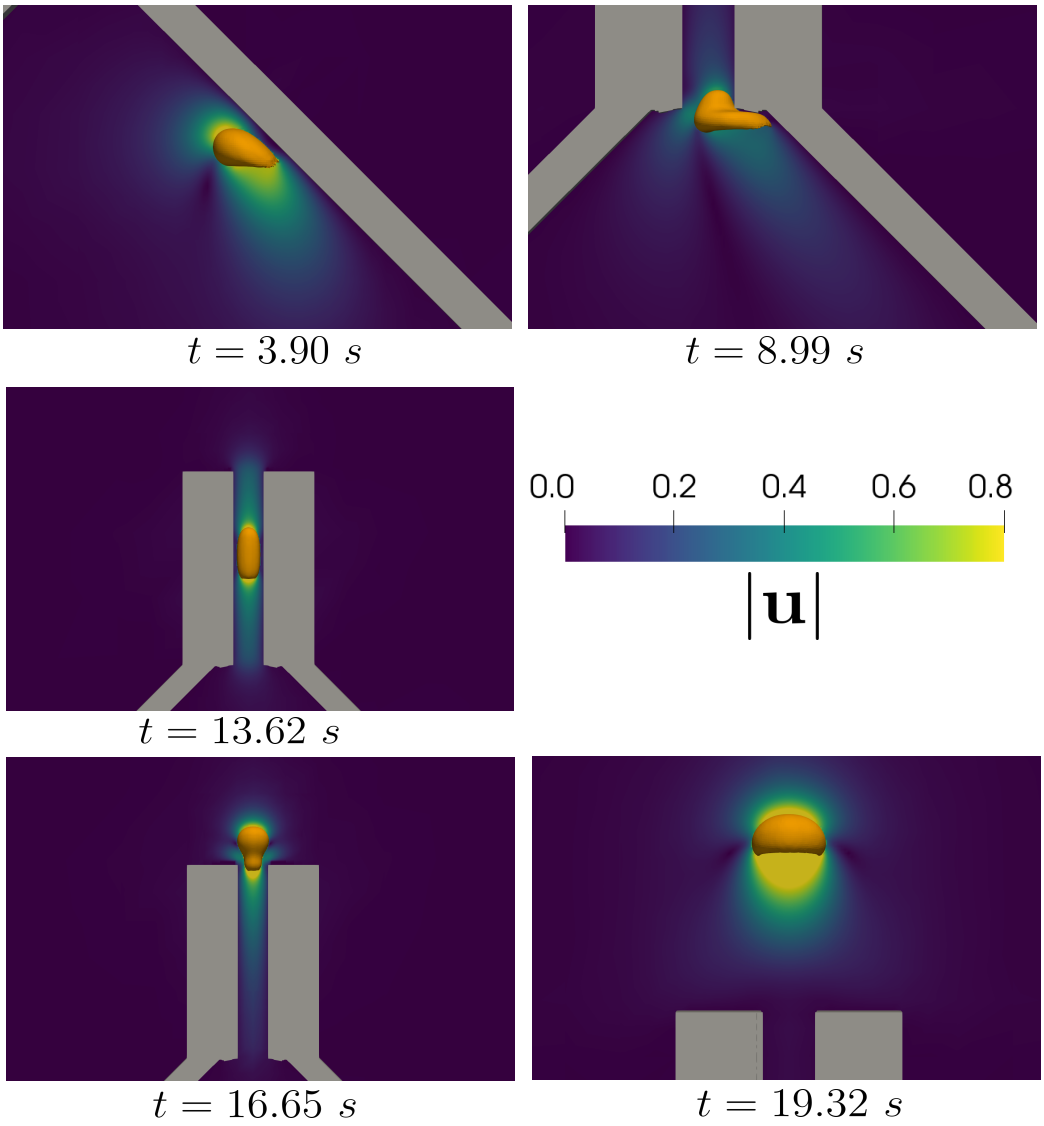


Figure 3.20: Evolution of the velocity magnitude, $|\mathbf{u}|$, for the bubble rising in an inverted funnel. Four of the five times shown correspond to steps (b) through (e) in Figure 3.19.

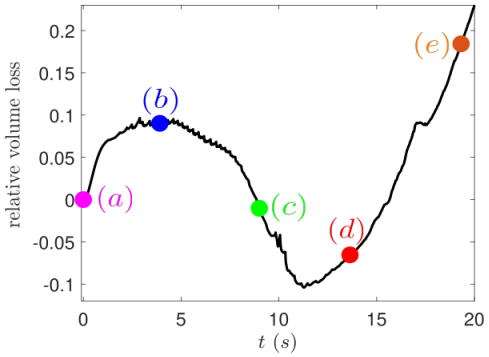


Figure 3.21: The relative volume loss of the bubble rising in an inverted funnel. Time steps (a) through (e) correspond to the time steps shown in Figure 3.19.

Chapter 4

Characteristic Bending for Incompressible Flows

4.1 Introduction

Advection under incompressible velocity fields arises in a wide range of applications, from oceanic transport to multiphase and interfacial flows. A defining property of incompressibility is that volumes are preserved exactly as they are transported. Enforcing this property numerically, however, is challenging. Errors from time integration, interpolation, or from velocity fields that are only approximately divergence-free introduce spurious compressibility, which accumulates over the course of a simulation. This can result in mass loss, distortion of interfaces, and reduced fidelity in simulations intended to resolve fine-scale structure. Although a variety of numerical strategies have been developed to mitigate these issues, these methods typically assume that the advecting velocity is incompressible and thus may not eliminate all sources of error.

In this work, we address these challenges by developing the Characteristic Bending (CB) method, a volume-preserving advection scheme designed specifically for incompressible flows. Our approach builds on standard numerical advection schemes, introducing a correction mechanism that removes spurious compressibility arising from discretization error or how incompressibility is enforced in a larger numerical scheme. The resulting method is broadly applicable for incompressible transport problems and can even be used to augment existing advection methods. To place our approach in context, we first review several common numerical strategies for advection, beginning with the semi-Lagrangian (SL) method.

The SL method is a standard numerical approach for advection problems that

circumvents traditional timestep restrictions imposed by the Courant–Friedrichs–Lewy (CFL) condition [137, 155]. The method advances the solution by tracing characteristics of the velocity field backward in time and interpolating the advected quantity at the departure point from the previous timestep. For scalar advection with linear interpolation, the SL method is unconditionally stable [51], which makes it particularly effective for simulations that require long-time advection [155] or for adaptive mesh refinement (AMR) settings, where small cut-cells can otherwise impose severe timestep constraints [139]. The primary drawback is that SL methods are not inherently conservative, and their accuracy is dominated by interpolation and integration errors, which accumulate systematically over repeated steps.

A wide range of conservative extensions of the semi-Lagrangian method have been developed to address the lack of inherent conservation. Flux-form conservative formulations, such as SLICE and CSLAM, recast the method in a finite-volume framework to enforce mass conservation through flux reconstruction [190, 96, 136]. Related approaches include the cubic interpolated propagation method with conservative semi-Lagrangian corrections (CIP–CSL) [184, 181], trajectory correction schemes based on the Monge-Ampère equation [39], and finite-volume variations adapted to cut-cell grids [139]. Lagrange–Galerkin methods are an alternate approach that solves the conservative form of the advection equation using a variational finite-element framework, with conservation enforced through a mapping of upstream volumes [117]. These schemes achieve both conservation and high accuracy, and when paired with spectral elements (see, e.g. [62, 61]), can resolve fine-scale features beyond the reach of flux-form formulations. Their primary drawback is the high computational expense associated with conservative mapping, which introduces globally coupled operations comparable in cost to the viscous or pressure solve in incompressible Navier–Stokes.

While conservative semi-Lagrangian methods for incompressible advection improve accuracy and mass conservation, they typically rely on the assumption that the velocity field is exactly divergence-free (*e.g.*, in [96]). When the velocity field is only approximately divergence-free, such as in approximate projection methods [6, 4, 5], semi-Lagrangian schemes will still follow characteristics of the inexact field. In the context of Navier–Stokes solvers, this rarely causes significant issues when the deviations from the divergence-free space are small, for example, in [20]. In other settings, a projection based on the Helmholtz–Hodge decomposition can be used to clean the velocity field, as in the classical MAC projection [3].

An additional limitation of semi-Lagrangian schemes is that, after every timestep, the advected quantity is interpolated and the reference to the original configuration is lost. Even if this process is performed in a conservative manner, repeated interpolation gradually degrades fine-scale features unless higher-order methods are employed. This

compounding error motivates an alternative perspective. Rather than advecting and interpolating the field directly, we can advect a map of the characteristics and compute the solution using a reference configuration.

The reference map technique (RMT) introduced in [83] makes this idea more precise. Originally developed to bridge Eulerian and Lagrangian formulations in fluid–structure interaction (FSI), RMT advances a reference configuration map that encodes material deformation. The construction parallels classical solid mechanics, where deformation gradients are the fundamental kinematic variable, while simultaneously providing a consistent link to Eulerian transport. While the formulation in [83] focused on constitutive modeling for solids, the underlying mechanism naturally extends to more general advection problems.

The Characteristic Mapping Method (CMM) [110, 188] and the Coupled Level Set Reference Map (CLSRM) [13] are two general approaches that extend the reference map perspective specifically to advection problems. In CMM, the characteristic map (*i.e.* the reference map) is evolved under the advecting velocity, and the advected quantity is reconstructed by composition with this map. Accuracy is maintained by decomposing the global deformation into near-identity submaps, evolved on a coarse grid and periodically remapped by interpolation onto a finer grid. In contrast, CLSRM advances a single reference map and avoids interpolation-based remapping, instead introducing a restarting procedure that reinitializes the map as needed to prevent a loss of bijectivity. Despite this difference in strategy, both methods can achieve very high accuracy when characteristics are well resolved numerically.

An important feature of the reference map formulation is that the map gradient encodes deformation metrics of the flow. This parallels the role of deformation gradients in solid mechanics, but here it provides a measure of how numerical advection distorts volume. Building on this observation, Theillard [164] introduced a volume-preserving projection for reference maps. The central idea is to measure the deviation of the map from the volume-preserving space and then apply a projection that enforces this constraint. For analytic velocity fields, where characteristics are well resolved and the velocity is exactly divergence-free, the projection provides little benefit and can even slightly degrade interface accuracy [164]. However, for more practical applications, such as rising droplets in density-stratified media, the projection substantially improves performance, reducing mass loss by two orders of magnitude. While the projection has little impact when the advecting velocity is exactly divergence-free, it provides a significant benefit in the presence of spurious compressible components introduced by numerical error. This observation motivates the framework developed in the present work.

An alternative strategy for enforcing volume preservation within the Characteristic Mapping framework was proposed in [75], where a symplectic time integration scheme

was employed to guarantee exact volume conservation during advection. While this approach is robust and structure-preserving, it assumes that the advecting velocity field is already divergence-free and the symplectic integrator itself does not correct residual compressibility that may arise, for example, from approximate projection methods. Similarly, [98] introduced a volume-conserving extension of the level set method for free-surface flows, achieving strict mass conservation through a geometric reconstruction procedure. Although effective, both approaches do not remove spurious compressible modes from the underlying velocity field.

In this work, we present the CB method. This method builds on foundational ideas of the CMM, CLSRM, and the VPRM to create a generalized method for advecting flows with incompressible velocity fields. We begin by recasting the classic SL scheme in the reference map setting and using this formulation to produce near-identity maps of the characteristics at each time step. Next, we apply the volume-preserving projection of [164] to reduce numerical errors from time integration and interpolation and filter any compressible modes present in the advecting velocity. The central idea is to view the projection as bending the characteristics toward the divergence-free space. Although the advecting velocity may contain small spurious compressible components, the characteristic bending framework applies the volume-preserving projection only to near-identity, single–time-step maps. This design ensures that corrections remain minimal and well posed, while preventing the accumulation of compressible errors over long integrations. When used as the primary advecting scheme, CB can further augment the reference map framework by creating long-time maps through the composition of CB-corrected steps. This yields a natural extension of the CLSRM method in which a single reference map is advected using CB to filter spurious compressibility through a volume-preserving transformation. The result is a robust methodology that can be viewed either as a drop-in replacement for SL advection or as a volume-preserving extension of reference map approaches.

The remainder of this chapter is organized as follows. Section 4.2 introduces the governing equations, the classical semi-Lagrangian scheme, general reference map advection, and the construction of the volume preserving projection. In Section 4.3, we provide the full details of our CB algorithm and discuss implementation details for adaptive quadtree and octree grids. Section 4.4 presents verification and validation studies, highlighting the key performance differences compared to other schemes. Additionally, in Section 4.4.2 we show how the CB can be incorporated into a two-phase, incompressible Navier-Stokes solver and highlight the associated improvements. Finally, Section 4.5 summarizes our findings and outlines future directions.

4.2 Preliminaries

In this section, we present the scalar advection equation and discuss its solutions when using an incompressible advecting velocity. Next, we review the traditional semi-Lagrangian method as our primary tool for solving the advection equation and discuss how the SL scheme is a natural way of constructing the reference map. The key takeaway from this section is that the SL method reconstructs individual characteristics, while the reference map represents the collection of characteristics. With this framework, we can leverage the reference map to design a volume-preserving correction for advecting quantities under incompressible fields.

4.2.1 Governing Equations

The Characteristic Bending method is fundamentally a numerical scheme for solving the advection equation that builds on ideas from the semi-Lagrangian (SL) method [155] and the reference map technique (RMT) [83, 140]. The scalar advection equation, using an Eulerian frame of reference, describes the transport of a scalar quantity (*e.g.*temperature, pressure, level set functions, etc.) by a velocity field and has the form,

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{u}\phi) = 0, \tag{4.1}$$

subject to the initial conditions;

$$\phi(\mathbf{x}, 0) = \phi_0(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega. \tag{4.2}$$

Here $\phi = \phi(\mathbf{x}, t)$ is the scalar quantity, $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ is the advecting velocity, $\mathbf{x}$ is the spatial coordinate in the domain Ω, and time $t \geq 0$. We are primarily concerned with incompressible velocity fields (*i.e.* $\nabla \cdot \mathbf{u} = 0$) for the work presented herein and can rewrite the advection equation as

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0. \tag{4.3}$$

If we consider a Lagrangian frame of reference, we can write Equation (4.3) in Lagrangian form as,

$$\frac{D\phi}{Dt} = 0, \tag{4.4}$$

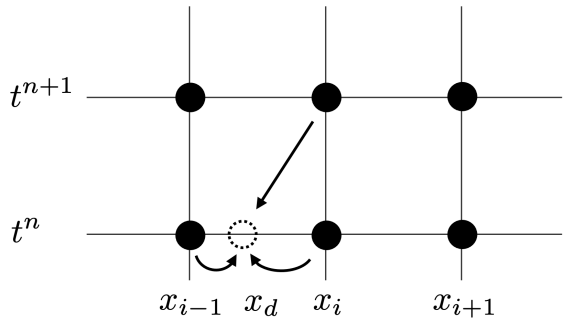


Figure 4.1: Schematic of the semi-Lagrangian method where the characteristic passing through grid point x_i at time t^{n+1} is traced backward to the departure point x_d. Additionally, the value of the advected quantity at the departure point is computed through interpolation using the neighboring points, x_{i-1} and x_i at time t^n.

where D/Dt is the material derivative. This implies that the advected quantity, ϕ, is constant along the characteristic curves, $\mathbf{X}(t)$,

$$\frac{d}{dt}\mathbf{X}(t) = \mathbf{u}\left(\mathbf{X}(t), t\right). \tag{4.5}$$

This invariance property of the advection equation naturally enforces conservation of the transported scalar.

4.2.2 Semi-Lagrangian Method

The semi-Lagrangian method, coined by Sawyer [143], is a numerical scheme for solving the advection equation by tracing characteristics backward in time using an Eulerian grid. The basic algorithm can be viewed as a two-step procedure. First, the backward trajectory of a grid point is computed to locate its departure point at the previous time. Since this departure point rarely coincides with a grid node, the second step involves evaluating the advected quantity at that location using an interpolation procedure. A schematic of this process is shown in Figure 4.1. The interested reader is referred to [53] (or the classic text [24]) for details of the historical development of semi-Lagrangian methods.

The semi-Lagrangian methodology offers two notable advantages. First, unlike forward-in-time integration schemes, it is not restricted by a traditional

Courant–Friedrichs–Lewy (CFL) condition[1] [137]. Second, when linear interpolation is used at departure points, the scheme is unconditionally stable for linear advection [51]. These methods are essential in numerical weather prediction, which often uses very large time steps [155, 24]. They are also valuable in situations where time-step restrictions are severe. For example, semi-Lagrangian approaches have been applied to turbulence simulations, where they relax stability limits while retaining high-order spatial accuracy [183], and in cut-cell finite volume methods, where they alleviate the restrictive constraints imposed by small cells [139].

By construction, the semi-Lagrangian method does not enforce conservation, and low-order time integration and interpolation schemes can introduce excessive dissipation. In fact, with linear interpolation and a CFL number less than one, the scheme is equivalent to the first-order upwind method, which is well known to be diffusive [137]. To address these shortcomings, a number of conservative formulations have been proposed [184, 136, 181], including Lagrange–Galerkin variations [117], though these generally increase computational cost as noted in [63]. Similarly, higher-order time integration and interpolation schemes have been developed to improve accuracy, but these are no longer unconditionally stable [51].

While considerable developments have been made to improve semi-Lagrangian schemes, their accuracy ultimately remains limited by the choice of time integration and interpolation. At its core, the semi-Lagrangian method is a single time-level scheme, where each step traces characteristics over one interval and updates the advected field through interpolation. This repeated tracing and interpolation compound errors at every step. An alternative approach is to consider the advection of the characteristics themselves, building a smooth mapping that evolves from the initial configuration to the current time.

4.2.3 Reference Map Advection

The Reference Map Technique (RMT), originally proposed by Kamrin [84] and independently by Cottet *et al.* [40, 105], solves advection problems by advecting the characteristics themselves. It does so by introducing a static Eulerian grid together with a reference map that encodes the mapping from the current configuration back to the original one. The notion of advecting characteristics through an auxiliary mapping is not new, however. Early formulations appeared in the Arbitrary Lagrangian–Eulerian (ALE) framework [74, 78, 43], where the computational mesh is deformed to follow the flow or

[1] In semi-Lagrangian schemes, the solution is interpolated at characteristic departure points rather than evaluated on a fixed stencil, ensuring that the update lies within the PDE's domain of dependence even when the CFL number exceeds unity.

large displacements, but requires periodic remeshing to maintain mesh quality. Although modern ALE formulations have significantly advanced remeshing and mesh optimization strategies (e.g., see [42, 178]), the RMT avoids these issues altogether by maintaining a fixed Eulerian mesh and reconstructing fields through interpolation at the roots of characteristic curves.

The Characteristic Mapping Method (CMM) [110] is a related technique developed specifically for advection problems. It builds on the mapping approach of [92] and employs the Gradient Augmented Level Set (GALS) method of [120] to advect the reference map on a fixed Eulerian grid. A similar idea, introduced for level set advection, is the Coupled Level Set–Reference Map (CLSRM) of Bellotti and Theillard [13], in which the standard level set method is augmented by advecting the reference map and reconstructing the level set function at each time step via pullback. This formulation reduces the frequency of level set reinitialization, thereby reducing mass loss and interface error. The CLSRM framework provides a natural and flexible formulation for general advection problems, and we adopt this viewpoint for the presentation herein.

The central idea of the reference map approach is to replace the direct advection of a scalar field ϕ with the advection of an auxiliary vector field, the reference map $\boldsymbol{\xi}$. At any location $\mathbf{x}$ and time t, the value $\boldsymbol{\xi}(\mathbf{x}, t)$ gives the initial position $\mathbf{x}_0$ along the characteristic curve that passes through $\mathbf{x}$ at time t. The reference map evolves according to

$$\frac{\partial \boldsymbol{\xi}}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\xi} = 0, \tag{4.6}$$

with the initial condition,

$$\boldsymbol{\xi}(\mathbf{x}, 0) = \mathbf{x}, \qquad \forall \mathbf{x} \in \Omega. \tag{4.7}$$

Once $\boldsymbol{\xi}$ is known, we can then construct the solution of Equation (4.3) through the composition of the reference map and ϕ_0 as,

$$\phi(\mathbf{x}, t) = \phi_0(\boldsymbol{\xi}(\mathbf{x}, t)), \quad \forall t \geq 0, \quad \forall \mathbf{x} \in \Omega. \tag{4.8}$$

This formulation makes it clear that the reference map is essentially a collection of the characteristics of the advection problem. In this sense, the semi-Lagrangian method can be interpreted as a special case of reference map advection over a single time step, where the "initial condition" is redefined at each update. The advantage of evolving $\boldsymbol{\xi}$ directly is that the initial configuration of the scalar field is preserved, avoiding error accumulation from repeated interpolation. Moreover, the reference map is typically a smoother object than the scalar field, making it easier to interpolate accurately. A schematic comparison of the two approaches is shown in Figure 4.2.

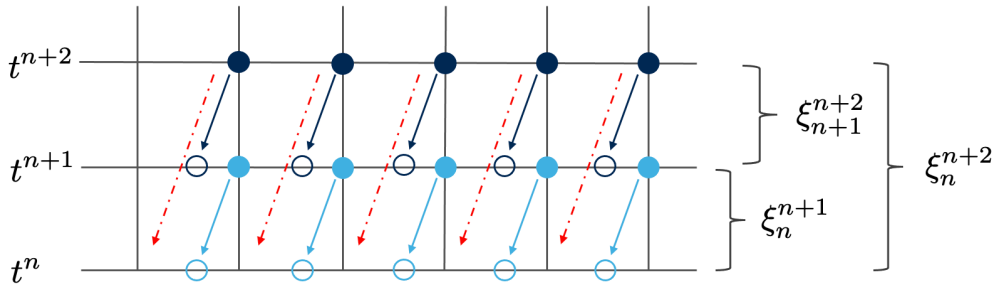


Figure 4.2: A schematic comparison of the semi-Lagrangian (light blue, dark blue) and Reference Map Advection (red) methods using the reference map, $\boldsymbol{\xi}$. The semi-Lagrangian method can be seen as a single time step variation of the reference map method (*e.g.* using $\boldsymbol{\xi}_n^{n+1}$, $\boldsymbol{\xi}_{n+1}^{n+2}$), whereas the reference map method traces the characteristics over multiple time steps (*e.g.* using $\boldsymbol{\xi}_n^{n+2}$).

4.2.4 Volume Preserving Correction

Thus far, we have introduced the reference map as a collection of characteristics used to reconstruct advected fields. A more formal viewpoint, however, reveals additional structure that can be used to quantify errors and guide corrections to the reference map formulation. In what follows, we adopt the treatment of [164] to develop this perspective.

We consider the deformed domain, $\mathcal{B}(t)$, and scalar field, $\phi(\mathbf{x}, t)$, defined for $t \geq 0$. We denote the initial domain as $\mathcal{B}_0$ and the initial scalar field as $\phi_0(\mathbf{x}, 0)$. We assume that this domain is deformed under the action of the velocity field, $\mathbf{u}$, and we define the motion map $\boldsymbol{\chi}(\cdot, t)$ as the morphism that transforms the initial domain $\mathcal{B}_0$ into the deformed one $\mathcal{B}(t)$. Specifically, this maps every material point $\mathbf{x}_0 \in \mathcal{B}_0$ to its corresponding image on $\mathcal{B}(t)$,

$$\mathbf{x}(t) = \boldsymbol{\chi}(\mathbf{x}_0, t) \quad \forall t \geq 0, \quad \forall \mathbf{x}_0 \in \mathcal{B}_0. \tag{4.9}$$

Assuming that $\boldsymbol{\chi}(\cdot, t)$ is a smooth invertible mapping (which holds provided the velocity field is well-behaved, *i.e.* it does not generate shocks, rarefactions, or intersecting trajectories), the inverse mapping $\boldsymbol{\xi}$ is defined as

$$\mathbf{x}_0 = \boldsymbol{\xi}(\mathbf{x}, t) \quad \forall t \geq 0, \quad \forall \mathbf{x} \in \mathcal{B}(t). \tag{4.10}$$

Here, $\boldsymbol{\xi}$ is the reference map, which maps each point $\mathbf{x}$ to the corresponding point $\mathbf{x}_0$ in the initial domain, $\mathcal{B}_0$. With this view, the deformation of the domain, $\mathcal{B}(t)$, is captured by the

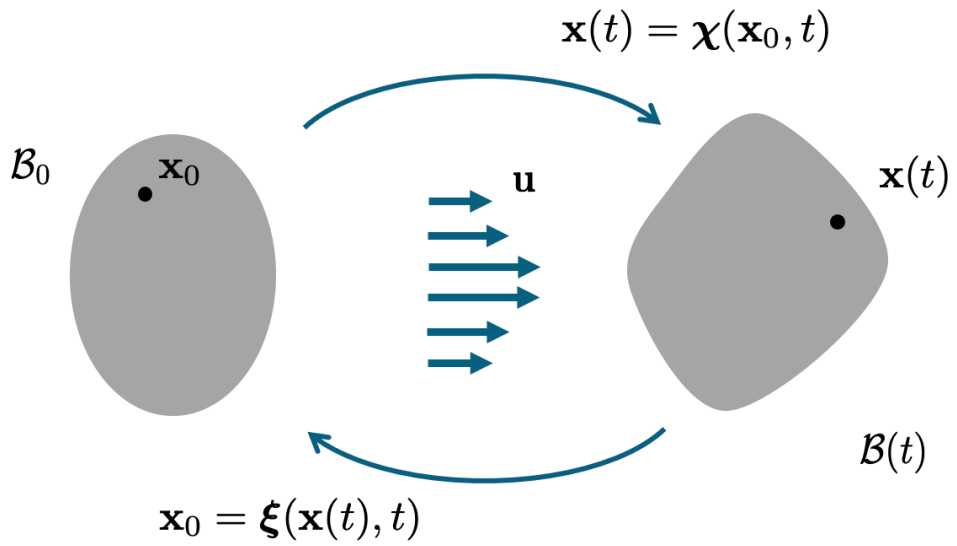


Figure 4.3: The basic schematic of the reference map method. We consider the initial domain, $\mathcal{B}_0$ deformed by the velocity field $\mathbf{u}$. The motion map, $\boldsymbol{\chi}(\cdot, t)$, transforms any point $\mathbf{x}_0$ in the domain, $\mathcal{B}_0$ to its associated image, $\mathbf{x}(t)$, in the deformed domain, $\mathcal{B}(t)$. The reference map, $\boldsymbol{\xi}(\cdot, t)$, is the inverse of the motion map and transforms any point $\mathbf{x}(t)$ to its initial position, $\mathbf{x}_0$, in the domain $\mathcal{B}_0$.

advection of the reference map

$$\frac{\partial \boldsymbol{\xi}}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\xi} = 0 \quad \forall t \geq 0, \quad \forall \mathbf{x} \in \mathcal{B}(t) \tag{4.11}$$

$$\boldsymbol{\xi}(\mathbf{x}, 0) = \mathbf{x} \qquad \forall \mathbf{x} \in \mathcal{B}_0. \tag{4.12}$$

The solution to the original advection problem can then be reconstructed from the composition of the reference map and the initial scalar field as

$$\phi(\mathbf{x}, t) = \phi_0(\boldsymbol{\xi}(\mathbf{x}, t)) \quad \forall \mathbf{x} \in \mathcal{B}(t). \tag{4.13}$$

A schematic of the relation between the motion map, $\boldsymbol{\chi}$, and the reference map, $\boldsymbol{\xi}$, is shown in Figure 4.3.

With this formalism, we can now use the reference map, $\boldsymbol{\xi}$, to probe properties of the transformation itself. In the Eulerian frame, incompressibility corresponds to a divergence-free velocity field (assuming constant material properties). In the Lagrangian frame, the equivalent statement is that the transformation preserves volume (see Figure 4.4). Using

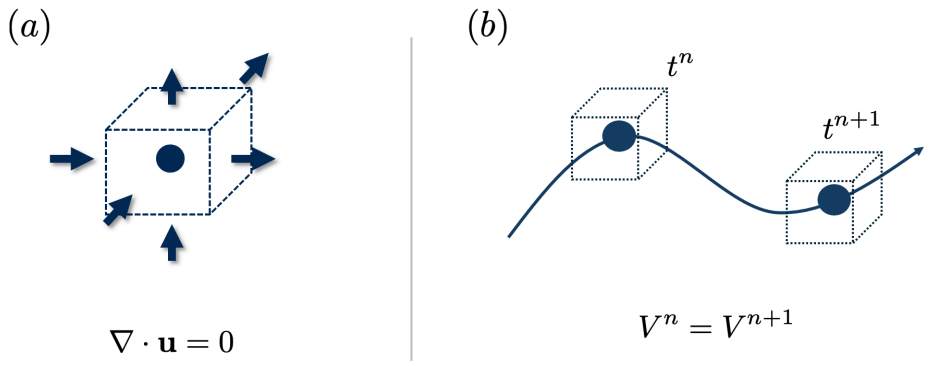


Figure 4.4: The Eulerian vs. Lagrangian perspective of incompressibility. In the Eulerian frame (a), incompressibility corresponds to a divergence-free velocity field, $\nabla \cdot \mathbf{u} = 0$. In the Lagrangian frame (b), the equivalent statement is that a material control volume preserves its volume during motion, $V^n = V^{n+1}$.

the reference map, this condition is expressed as

$$\det\left(\nabla \boldsymbol{\xi}(\mathbf{x}, t)\right) = 1 \quad \forall \mathbf{x} \in \mathcal{B}(t). \tag{4.14}$$

While this condition holds in theory, the numerical construction of the reference map (time integration and interpolation) introduces errors that cause the map to drift away from the volume-preserving space. Furthermore, for velocity fields that are only approximately divergence-free, such as those obtained with approximate projection methods, the characteristics themselves can produce deviations from the volume preserving space.

In [164], these errors are addressed by introducing a volume-preserving projection for the reference map. The key idea is that the numerical evolution of the map may drift from the volume-preserving space, even when advected with an incompressible velocity field. By quantifying these deviations, we can design a projection that corrects the map by projecting it back into the admissible space. To formalize this, we denote the intermediate map as $\boldsymbol{\xi}^*(\mathbf{x}, t)$, which is obtained form the numerical approximation of Equation (4.11). Since $\boldsymbol{\xi}^*$ is not guaranteed to satisfy (4.14), we seek a correction operator $\boldsymbol{\gamma}^{-1}$ such that the composition satisfies

$$\det\left(\nabla \boldsymbol{\gamma}^{-1}\left(\boldsymbol{\xi}^*(\mathbf{x}, t)\right)\right) = 1. \tag{4.15}$$

Here, the correction $\boldsymbol{\gamma}^{-1}$ is assumed to be a near identity (see [164]) transformation as the deviations from the volume-preserving space are typically on the order of the numerical accuracy of the scheme used to solve (4.14) (*e.g.* $\mathcal{O}\left(\Delta t^\alpha + \Delta x^\beta\right)$ for the semi-Lagrangian

method, where α and β represent the order of the backward time integration and spatial interpolation, respectively). Informally, $\boldsymbol{\gamma}^{-1}$ can be interpreted as re-mapping the reference map to restore volume preservation.

Although many volume-preserving corrections could be constructed, [164] adopts a particular choice based on a variational principle. The correction $\boldsymbol{\gamma}^{-1}$ is defined as the unique minimizer of a constrained L^2 optimization problem that seeks the smallest modification of $\boldsymbol{\xi}^*$ subject to the volume-preserving constraint. This optimality condition reduces to a Poisson equation for the adjoint λ,

$$-\Delta\lambda = 1 - \det\left(\nabla\boldsymbol{\xi}^*\right) \quad \forall \mathbf{x} \in \Omega, \tag{4.16}$$

$$\lambda = 0 \qquad\qquad\quad \forall \mathbf{x} \in \partial\Omega, \tag{4.17}$$

for which the correction is, to leading order[2]

$$\boldsymbol{\gamma}^{-1}(\mathbf{x}) = \mathbf{x} - \nabla\lambda. \tag{4.18}$$

Conceptually, this projection is a direct analog of the projection step used in incompressible Navier-Stokes solvers, carried out in the Lagrangian reference frame.

The Volume-Preserving Reference Map (VPRM) method of Theillard [164] incorporates a volume-preserving projection directly into the numerical evolution of the reference map by applying this correction after every time step. Designed explicitly for the evolution of level set–defined interfaces, this approach has been shown to substantially reduce mass loss in multiphase flow simulations. However, for general advection problems in which the advecting velocity is analytically divergence-free, the volume-preserving projection may introduce small numerical perturbations to the reference map arising from errors in the construction of the characteristics. These perturbations can, in turn, lead to distortion of the level set interface. As a result, the benefits of the VPRM are limited for idealized advection problems, and a modest degradation in interface accuracy may be observed. These observations motivate a more careful examination of how and when the volume-preserving projection should be applied, which forms the basis for the present work.

It is important to highlight several key assumptions underlying the derivation of the volume-preserving correction in [164]. In particular, the correction is assumed to remain close to the identity, under the expectation that deviations from the volume-preserving space are of the same order as the discretization error of the underlying advection scheme.

[2]Here, leading order refers to a first-order expansion about the identity map, under the assumption that the correction remains close to the identity. In this regime, the deviation of the intermediate map from volume preservation satisfies $\det(\nabla\boldsymbol{\xi}^*) - 1 = \mathcal{O}(\varepsilon)$ with $\varepsilon \ll 1$. Equation (4.18) follows from linearizing the determinant constraint and neglecting higher-order terms of $\mathcal{O}(\varepsilon^2)$. See Theillard [164] for details.

While this assumption is well justified for a single time step, its implications are less clear when the reference map is evolved over many steps, as in the VPRM algorithm. This issue is closely related to the linearization of the determinant constraint, since small deviations may be repeatedly encoded into the long-time reference map. In practice, these effects are partially mitigated by the combination of projection and restarting strategies, but they nonetheless highlight important considerations for the long-time performance of volume-preserving schemes.

4.3 Characteristic Bending

In this section, we present the characteristic bending method, a generalized framework for reference map advection. We begin with a high-level description of the algorithmic structure and the motivation for reformulating existing approaches. Next, we outline the specific numerical discretizations used to construct the reference map with the semi-Lagrangian method and to perform the volume-preserving projection. Finally, we briefly describe where to find the additional implementation details required to incorporate the method on adaptive quadtree and octree grids.

The key takeaway from this section is to interpret the volume-preserving projection as a means of bending characteristics toward the divergence-free space. This operation simultaneously reduces numerical errors from characteristic tracing and mitigates compressible effects introduced by numerical approximation of the advecting velocity. In this work, the projection is applied to single-step reference maps that are close to the identity by construction, ensuring that the correction operates in the regime where the volume-preserving constraint can be treated linearly. This yields a method that attenuates compressible modes while avoiding repeated perturbations of the reference map.

4.3.1 Overall Method

The characteristic bending method is a general framework for advection with incompressible velocity fields. This method can be applied as a standalone advection scheme or it can be used to augment other numerical methods, such as in reference map advection. The basic CB algorithm is presented in Figure 4.5, and its extension to reference map advection is shown in Figure 4.6.

The basic characteristic bending algorithm proceeds in three stages. In the advection step, an intermediate reference map $\boldsymbol{\xi}^*$ is advanced from t^n to t^{n+1} by solving an advection equation with velocity $\mathbf{u}$. The map is initialized as the identity, $\boldsymbol{\xi}^*(\mathbf{x}, t) = \mathbf{x}$, so that after advection it represents the characteristic curves traced backward from t^{n+1} to t^n. In the

1. Advection Step

Advect an intermediate reference map, $\boldsymbol{\xi}^*$, from time t^n to t^{n+1} by solving the advection equation

$$\left.\begin{aligned} \frac{\partial \boldsymbol{\xi}^*}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\xi}^* = 0 \\ \boldsymbol{\xi}^*\left(\mathbf{x}, t^n\right) = \mathbf{x} \end{aligned}\right\} \forall \mathbf{x} \in \Omega.$$

2. Correction Step

Compute the correction to the map by solving the Poisson system for λ

$$\begin{aligned} -\Delta\lambda = 1 - \det\nabla\boldsymbol{\xi}^* \quad \forall \mathbf{x} \in \Omega, \\ \lambda = 0 \qquad\qquad\quad \forall \mathbf{x} \in \partial\Omega, \end{aligned}$$

and the correction as

$$\gamma^{-1}(\mathbf{x}) = \mathbf{x} - \nabla\lambda.$$

The volume-preserving map, $\boldsymbol{\xi}$, is then constructed as

$$\boldsymbol{\xi}(\mathbf{x}, t^{n+1}) = \boldsymbol{\xi}^*(\gamma^{-1}(\mathbf{x}), t^{n+1}).$$

3. Reconstruction

Reconstruct the solution of the scalar field, ϕ, at time t^{n+1} as

$$\phi(\mathbf{x}, t^{n+1}) = \phi\left(\boldsymbol{\xi}\left(\mathbf{x}, t^{n+1}\right), t^n\right).$$

Figure 4.5: The basic characteristic bending algorithm. At each time step, an intermediate reference map is advected, corrected through a volume-preserving projection, and used to reconstruct the scalar field.

correction step, deviations of $\boldsymbol{\xi}^*$ from the volume-preserving space form the right hand side of a Poisson equation for the adjoint λ. The gradient of λ is used to determine the correction γ^{-1} and the corrected, volume-preserving map is obtained by composing this correction with $\boldsymbol{\xi}^*$. Importantly, because the intermediate map is initialized as the identity at each step, numerical errors are confined to the characteristic tracing and interpolation within a single time step. Finally, in the reconstruction step, the scalar field ϕ at time t^{n+1} is obtained by composing $\phi(\cdot, t^n)$ with the corrected, volume-preserving map.

The characteristic bending method differs from existing approaches in how it applies the volume-preserving projection. The semi-Lagrangian scheme, which CB uses to construct the intermediate reference map, is efficient and stable, but it does not inherently conserve volume or eliminate compressible modes introduced by numerical error. This behavior is illustrated in Figure 4.7, which contrasts standard characteristic tracing of the semi-Lagrangian scheme with the corrected trajectories produced by the characteristic

1. Advection Step

Advect an intermediate reference map, $\boldsymbol{\psi}^*$, from time t^n to t^{n+1} by solving the advection equation

$$\left.\begin{aligned} \frac{\partial \boldsymbol{\psi}^*}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\psi}^* = 0 \\ \boldsymbol{\psi}^*\left(\mathbf{x}, t^n\right) = \mathbf{x} \end{aligned}\right\} \forall \mathbf{x} \in \Omega.$$

2. Correction Step

Compute the correction to the map by solving the Poisson system for λ

$$\begin{aligned} -\Delta\lambda = 1 - \det\nabla\boldsymbol{\psi}^* \quad \forall \mathbf{x} \in \Omega, \\ \lambda = 0 \qquad\qquad\quad \forall \mathbf{x} \in \partial\Omega, \end{aligned}$$

and the correction as

$$\gamma^{-1}(\mathbf{x}) = \mathbf{x} - \nabla\lambda.$$

The volume-preserving map, $\boldsymbol{\psi}$, is then constructed as

$$\boldsymbol{\psi}(\mathbf{x}, t^{n+1}) = \boldsymbol{\psi}^*(\gamma^{-1}(\mathbf{x}), t^{n+1}).$$

3. Reference Map Update

Update the reference map, $\boldsymbol{\xi}$, by composition as

$$\boldsymbol{\xi}(\mathbf{x}, t^{n+1}) = \boldsymbol{\xi}(\boldsymbol{\psi}(\mathbf{x}, t^{n+1}), t^n).$$

4. Reconstruction

Reconstruct the solution of the scalar field, ϕ, at time t^{n+1} as

$$\phi(\mathbf{x}, t^{n+1}) = \phi\left(\boldsymbol{\xi}\left(\mathbf{x}, t^{n+1}\right), t^n\right).$$

Figure 4.6: The characteristic bending method applied to reference map advection. Here, the basic CB algorithm is embedded within the reference map framework. Instead of directly reconstructing the scalar field from the corrected single-step map, the long-time reference map is updated by composition with the corrected near-identity map.

bending step. Similarly, the CLSRM framework [13] advects the characteristics of the velocity field but lacks any internal mechanism to enforce incompressibility. For analytically divergence-free velocities, CLSRM is highly accurate, but in practice it is prone to errors in the velocity field unless frequent restarts are employed. The characteristic bending method is closely related to the VPRM and, in fact, is equivalent when the map is restarted every iteration. The distinction between the two approaches becomes clear when considering multi–time-step reference maps. The VPRM applies its projection directly to the evolving long-time map, which can correct small numerical

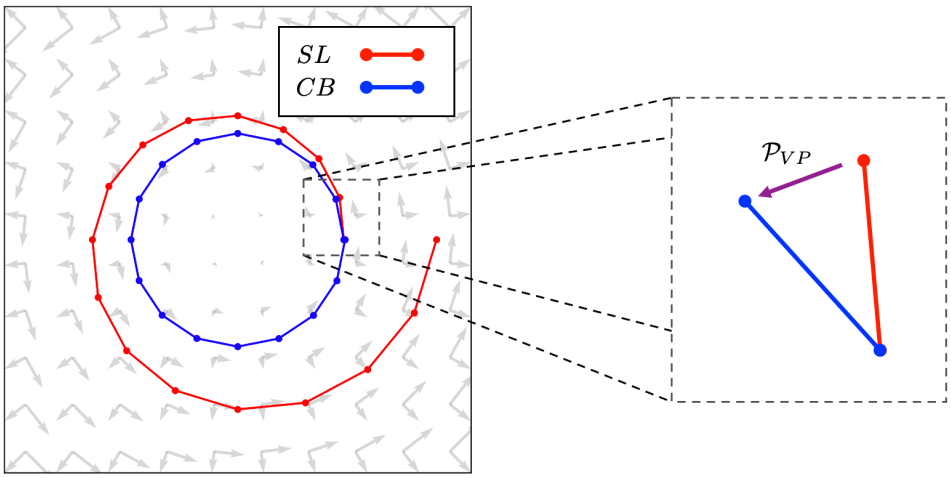


Figure 4.7: Schematic illustration of the SL (red) and the CB (blue) methods tracing the characteristics for solid-body rotation with artificial expansion. The SL method spirals outward following the modified velocity field, while the CB method remains concentric. The projection triangle, shows the correction and the difference between the two methods. Here, $\mathcal{P}_{VP}$ is a volume preserving projection applied to a single-time level characteristic.

errors but may accumulate distortion and fail to fully filter compressible modes introduced by the velocity field. By contrast, characteristic bending applies the projection only to near-identity single-step maps, ensuring the correction is minimal and well-posed, and then composes these corrected maps into the long-time reference map. A comparison of the three reference map-based methods is shown in Figure 4.8.

The characteristic bending method provides a general framework for advection with incompressible velocity fields that builds on the semi-Lagrangian method, reference map advection, and a volume-preserving projection. The central novelty is to apply the projection only to near-identity maps, ensuring that corrections remain minimal, consistent, and free from the long-time drift associated with projecting the full reference map. This viewpoint clarifies CB as a principled recombination of existing building blocks that substantially improves robustness and accuracy. In the following subsections, we provide the discretization details for the advection step, the projection step, and reconstruction of the solution for implementation on adaptive quadtree and octree grids.

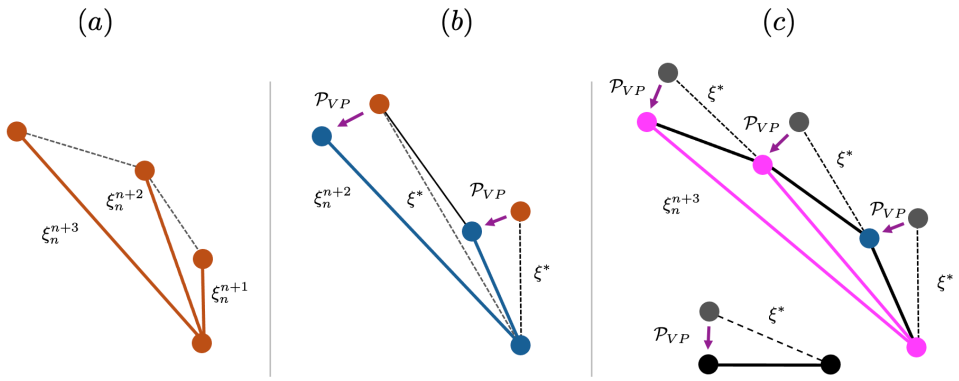


Figure 4.8: (a) Construction of the reference map (CLSRM) for solid-body rotation with artificial expansion. The CLSRM maps trajectories back to the starting point. (b) The VPRM projects the entire reference map. (c) RMCB projects only a temporary, single-timestep reference map and then composes this with the full reference map.

4.3.2 Advection Step

The first step of the characteristic bending method advances the intermediate reference map ξ^* from t^n to t^{n+1} using the semi-Lagrangian method. Since ξ^* is initialized as the identity at each time step, this advection step amounts to tracing characteristics backward from grid points at t^{n+1} to their departure locations at t^n.

The departure point $\mathbf{x}_d^n$ is computed using a second-order backward Runge–Kutta (RK2) scheme (previously defined by Equations (3.44)-(3.46)),

$$\hat{\mathbf{x}} = \mathbf{x}^{n+1} - \frac{\Delta t^n}{2}\mathbf{u}^{n+1}(\mathbf{x}^{n+1}), \tag{4.19}$$

$$\hat{\mathbf{u}}^{n+1/2}(\hat{\mathbf{x}}) = \left(1 + \frac{\Delta t^n}{2\Delta t^{n-1}}\right)\mathbf{u}^n(\hat{\mathbf{x}}) - \frac{\Delta t^n}{2\Delta t^{n-1}}\mathbf{u}^{n-1}(\hat{\mathbf{x}}), \tag{4.20}$$

$$\mathbf{x}_d^n = \mathbf{x}^{n+1} - \Delta t^n \hat{\mathbf{u}}^{n+1/2}, \tag{4.21}$$

where $\mathbf{u}^n$ is the velocity field at time t^n. To evaluate velocities at off-grid points, we use quadratic interpolation with WENO-limited second derivatives. Specifically, if $\mathbf{x} = (x, y)$ lies in a cell $\mathcal{C}$, here taken for illustration as $[0, 1]^2$, we define our generic interpolant φ as

$$\begin{aligned}\varphi(x,y) = \varphi^{00}(1-x)(1-y) + \varphi^{01}(1-x)y + \varphi^{10}x(1-y) + \varphi^{11}xy \\ -\varphi_{xx}\frac{x(1-x)}{2} - \varphi_{yy}\frac{y(1-y)}{2},\end{aligned} \tag{4.22}$$

where φ^{ij} denotes the nodal values at the corners of $\mathcal{C}$, and the second derivatives are computed in a WENO fashion [101], e.g.

$$\varphi_{xx} = \frac{1}{W_x} \sum_{n \in \text{nodes}(\mathcal{C})} \varphi^n_{xx} w^n_x, \qquad w^n_x = \frac{1}{|\varphi^n_{xx}|^2}, \qquad W_x = \sum_{n \in \text{nodes}(\mathcal{C})} w^n_x, \tag{4.23}$$

where φ^n_{xx} are the discrete second-order derivatives at node n, computed using the finite-differences method presented in [111].

With the departure point defined, we next address the approximation of the velocity $\mathbf{u}^{n+1}$. Although it is common in SL methods to approximate $\mathbf{u}^{n+1}$ by $\mathbf{u}^n$ (*e.g.*, [24, 86]), we instead adopt the trajectory reconstruction proposed in [20] (previously defined in (2.83)),

$$\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) = \mathbf{u}^n(\mathbf{x}^{n+1}) + \frac{\Delta t^n}{\Delta t^{n-1}} \Big(\mathbf{u}^n(\mathbf{x}^{n+1}) - \mathbf{u}^{n-1}(\mathbf{x}^{n+1}) \Big) + \mathcal{O}(\Delta t^2), \tag{4.24}$$

which has been shown to improve stability and accuracy, particularly at large CFL numbers.

We note that using a forward-time extrapolated velocity field for $\mathbf{u}^{n+1}$ may lead to instabilities in the standard two–time-level semi-Lagrangian method. In [76], the authors identify this extrapolation as the source of these instabilities and propose the SETTLS scheme to correct the issue while maintaining second-order accuracy in time. In the examples presented here, $\mathbf{u}^{n+1}$ is either known a priori or, in the case of incompressible multiphase flows, the simulation is diffusion-dominated, which appears to suppress these instabilities. In strongly advection-dominated flows, a predictor–corrector formulation is often used, with $\mathbf{u}^{n+1} = \mathbf{u}^n$ in the predictor step. For the incompressible Euler and incompressible multiphase Navier–Stokes examples considered here, we did not observe any difference between the predictor–corrector and time-extrapolation approaches and therefore used the method presented in [20].

Once the departure points $\mathbf{x}^n_d$ are computed, the intermediate reference map ξ^* is updated by interpolating its values from t^n at those locations. The combination of the second-order RK scheme with WENO interpolation yields a second-order accurate method in both space and time. For further analysis of the accuracy of SL methods, we refer the reader to [51].

4.3.3 Correction Step

In the projection step, we solve the Poisson problem (4.16)–(4.17) and compute the correction (4.18). The system is solved using the supra-convergent[3] solver of Min *et al.* [114], designed for quadtree and octree grids. This solver achieves second-order accuracy

[3]Here supra-convergent means that both the solution and its gradients are both second-order accurate.

for both the solution and its gradient, ensuring that the correction is computed with second-order accuracy.

The projection step is the most computationally expensive component of the characteristic bending algorithm, as it requires the solution of a global elliptic problem. As noted in [164], this cost can exceed that of alternative correction strategies such as the Coupled Level Set/Volume-of-Fluid method [159, 157] or conservative level set formulations [122, 123, 41, 128]. These approaches are primarily designed to enforce global mass or volume conservation and often impose stricter stability constraints. Moreover, they do not directly address compressible effects introduced by numerical errors in the advecting velocity field.

In contrast, the projection in the characteristic bending method is driven by local, pointwise deviations of the reference map from the volume-preserving space and uses this information to compute a global correction through an elliptic solve. In this sense, the method performs a local-to-global correction, where local violations of the volume-preserving constraint are identified and attenuated through a globally consistent modification of the map. To leading order, this procedure suppresses compressible distortions in the reference map, without relying solely on global volume conservation. This local enforcement mechanism, applied here to single-step reference maps, is central to the robustness of the characteristic bending framework.

4.3.4 Numerical Implementation

Details of the numerical implementation, including the data structures used for the quadtree/octree grids and the adaptive mesh refinement (AMR) methodology, are presented in Section 3.2.4. These aspects follow standard practices established in previous studies (*e.g.*, [67, 167, 13, 164, 20, 17]) and do not introduce new algorithms. The appendix serves mainly to ensure completeness and reproducibility of the numerical framework.

4.4 Computational Results

4.4.1 Advection Examples

In this section, we use a set of analytic examples to compare the characteristic bending method with similar advection schemes (*e.g.*, semi-Lagrangian, Coupled Level Set Reference Map, and Volume-Preserving Reference Map) in two dimensions. The examples used consist of a Gaussian distribution and a level set function subject to a constant angular velocity field and a level set function subject to an incompressible, but

deformational flow. For each example, we examine the performance of the numerical schemes for the purely incompressible field as well as the incompressible field coupled with a compressible component of varying strength. Finally, we present a simple nonlinear example in the form of the incompressible Euler equations.

Gaussian Advection

The purpose of this example is to verify that our characteristic bending scheme, as well as the comparative methods, yields the correct order of accuracy when applied to the advection of an infinitely smooth scalar field. The benefit of using a scalar field defined by a Gaussian distribution is that we easily measure the mass loss by computing the first moment of the advected quantity. Furthermore, by using a rotational velocity field defined over a Cartesian grid, we eliminate any serendipitous aliasing[4] that could artificially enhance our results. Hence, the Gaussian Rotation test is a simple, but non-trivial testbed to compare our characteristic bending scheme against standard methods.

For this example, we consider the advection of the scalar function,

$$\phi(x,y) = \frac{1}{\sqrt{2\sigma\pi}} \exp\left(-\frac{(x-0.5)^2+y^2}{2\sigma^2}\right), \tag{4.25}$$

in the domain, $\Omega = [-1,1]^2$. This function represents a radial Gaussian centered at $(0.5,0)$ and we advect this function using the incompressible velocity field, $\mathbf{u} = (-y,x)$ from time $t=0$ to $t=2\pi$. We also consider adding a compressible component to the velocity field, in the form of artificial expansion, to simulate a scenario in which the advecting velocity is not entirely divergence free (*e.g.* as a result of an approximation projection method). We scale this compressible component by the maximum refinement level of our mesh, Δx_{min}, with parameters, α and β, to represent either a first- or second-order deviation from the incompressible space. We center the compressible component at the origin, leading to the velocity field,

$$\mathbf{u}(x,y,t) = \begin{cases} -y \;+\; \alpha\Delta x_{min}\, x + \;\beta\left(\Delta x_{min}\right)^2 \; x \\ \quad x \;+\; \alpha\Delta x_{min}\, y + \;\beta\left(\Delta x_{min}\right)^2 \; y \end{cases} \quad \text{for } t \in [0,2\pi]. \tag{4.26}$$

We measure the accuracy of our numerical method using the L^∞ norm

$$L^\infty = \max_{\Omega}\left(|\phi(x,y,0) - \phi(x,y,2\pi)|\right) \tag{4.27}$$

[4]This is, for example, what happens when using the first-order upwinding scheme for a one-dimensional advection equation with a CFL of 1 defined on a uniform, periodic grid.

Table 4.1: Description of the acronym used for each method.

Acronym	Method
SL	Second-order semi-Lagrangian Method
CLSRM	Coupled Level Set Reference Map [13]
VPRM	Volume-Preserving Reference Map [164]
CB	Characteristic Bending (Figure 4.5)
RMCB	Reference Map Characteristic Bending (Figure 4.6)

and measure the relative mass loss, M, by computing,

$$M = \left| 1 - \frac{\int_\Omega \phi(x, y, 2\pi)\ d\Omega}{\int_\Omega \phi(x, y, 0)\ d\Omega} \right|. \tag{4.28}$$

The scalar field, ϕ, and the velocity field, $\mathbf{u}$, are both infinitely smooth and the characteristics of the velocity field can be well resolved by the reference map based methods (*i.e.*the Coupled Level Set Reference Map and the Volume Preserving Reference Map). We use a second-order Runge-Kutta method to compute the departure point and the quadratic WENO interpolation (see, *e.g.* Equation (2.42)) to interpolate the scalar, ϕ, and the map, ξ, between grid points. Additionally, for this example we use Dirichlet boundary conditions and allow for extrapolation outside of the domain to remove any accuracy penalties from boundary clipping effects[5]. Because the characteristics of this velocity field can be well resolved, the map, ξ, does not require any restarts.

We compare the results of our CB scheme with the SL, the CLSRM in [13], and the VPRM in [164] using this example. While the CLSRM and VPRM methods were specifically developed in the level set context, they can naturally be applied to the advection of scalar fields. Additionally, we include results for our the augmented reference map characteristic bending method (RMCB).

We test each of these numerical methods on a uniform grid by setting the minimum and maximum refinement levels equal (we consider adaptive meshes in the subsequent examples). We test each scheme with no artificial expansion (*i.e.*$\alpha = 0, \beta = 0$), a second-order deviation from the incompressible space (*i.e.*$\alpha = 0, \beta = 1$), and a first-order deviation from the incompressible space (*i.e.*$\alpha = 1, \beta = 0$). Each of these methods inherit the stability properties of the semi-Lagrangian scheme (see [51]) and we demonstrate this by using a CFL number of 5. We present the results for SL, CLSRM, VPRM, CB, and RMCB in Figures 4.9 and 4.10 for the solution error and mass loss, respectively. For reference, Table 4.1 provides a description of each acronym.

[5]For example, if the root of a characteristic of lies slightly outside of the domain.

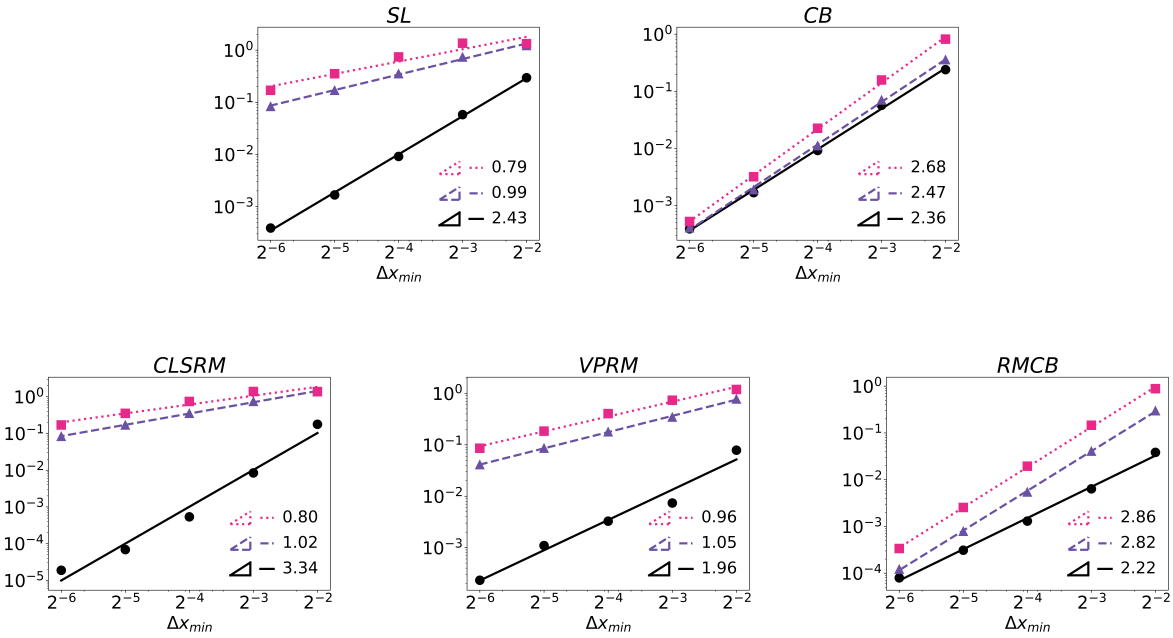


Figure 4.9: Solution error $\|\phi - \phi_e\|_\infty$ for the Gaussian advection example with various level of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

Starting with SL, we see that without any artificial expansion we see second-order convergence in the L^∞ norm and third-order convergence in mass loss, M. When we introduce the artificial expansion, however, the order of accuracy for the semi-Lagrangian experiences a significant drop. This is entirely expected as the SL method does not have any machinery to filter the added compressible modes. Similarly, we see the CLSRM perform exceptionally well with the analytically divergence-free velocity field and then suffer once the artificial expansion is added. Again, the CLSRM does not include any way to filter out compressible modes. The VPRM does perform slightly better than the previous methods, however, here we see the effect of projecting long-time map and the accumulation of errors.

The results for the CB scheme present a much more robust handling of the artificial expansion. Without artificial expansion, both the SL and CB methods yield nearly identical results. CB continues to yield similar results even after the artificial expansion is added. Here, we do see an impact when the first order expansion is added at coarser levels of

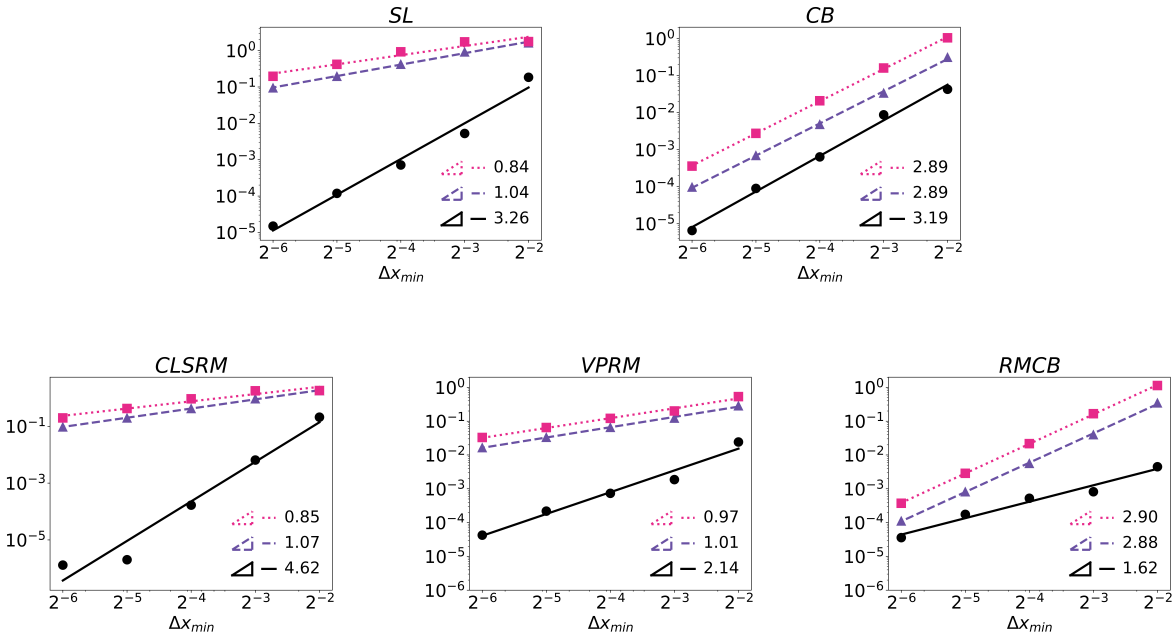


Figure 4.10: Mass loss for the Gaussian advection example with various level of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

refinement. However, this impact decreases as the grid is refined. We see a similar dynamic with the RMCB and see that the error associated with the expansion decrease as the grid is refined.

Slotted Cylinder

The purpose of this example is to explore how CB, in comparison to other methods, preserves sharp interface features. A canonical example problem for demonstrating this is the slotted cylinder of [189], where the profile of a circle with a slot is embedded into an initial distribution function by setting the value of the distribution to 3 inside of the circle and 1 outside of the circle. This profile is then advected by a field with constant angular velocity for one full revolution around the axis of rotation. After one full revolution, we can measure the accuracy of the numerical scheme by comparing the final advected profile with the initial profile, making this a particularly effective example for visually demonstrating

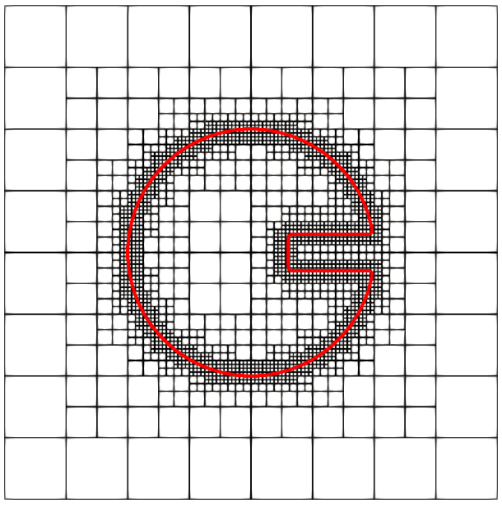


Figure 4.11: Visual representation of the initial profile (*red*) for the Slotted Cylinder example using an adaptive quadtree mesh, with a minimum and maximum refinement level of 4 and 7, respectively. The circle is centered at the origin, $(0, 0)$ with a radius of $R = 0.5$. The slot has a width of $w = 0.15$ and is placed at a height of $h = 0.15$ from the origin.

numerical diffusion.

A modern variation of this problem, and what we do herein, embeds the slotted cylinder profile using a level set function, centered in the computational domain. Specifically, we consider the advection of an interface defined by the zero-contour of the level set function shown in Figure 4.11. For this example, we use the domain $\Omega = [-1, 1]^2$ and the velocity field, $\mathbf{u} = (-y, x)$. As with the previous example, we include a compressible component in the velocity field centered at the origin, which leads to the same velocity field defined by Equation (4.26).

We use an adaptive quadtree mesh and refine the mesh near the interface (also shown in Figure 4.11). Specifically, we use the refinement criteria introduced in Section 3.2.4 with the uniform band set to $8\Delta x_{min}$[6]. This value is based on a heuristic, but is loosely chosen to ensure that the foot of the characteristics originating from nodes near the interface lie in region of maximum refinement. Additionally, we set the span between the minimum and maximum refinement levels to be 4 (*i.e.* minimum level 3, maximum level 7) and increment both the minimum and maximum refinement levels to compute the order of

[6]This has been a common practice for level set methods since the early work of Adalsteinsson and Sethian in [2].

accuracy. Finally, we set the finest resolution for this example to a minimum level of 7 and a maximum level of 11.

In this example, the interface of the slotted cylinder is embedded as the zero contour of a signed distance function, ϕ. While the level set formulation does not strictly require ϕ to remain a signed distance function (see, e.g., Sethian [152] and Osher and Fedkiw [127]), maintaining this property is important for the accurate computation of interface geometry, such as normals and curvature.

In practice, the signed distance property can be maintained through a variety of reinitialization strategies. One common approach is to solve the Eikonal equation $|\nabla\psi| = 1$ with boundary conditions prescribed at the interface, for which the Fast Marching Method (FMM) provides an efficient numerical solver [151]. An alternative strategy, introduced by Chopp [34], recomputes the signed distance function by explicitly determining the minimal distance between grid points and the zero contour of ϕ, yielding improved accuracy near the interface. Hybrid approaches have also been developed, in which closest-point or minimal-distance calculations are used in a neighborhood of the interface and marching methods are employed further away [33]. More recently, Saye [144] proposed a high-order closest-point method that achieves improved geometric accuracy and reduced numerical diffusion and is particularly effective for complex or highly curved interfaces.

In the present example, we adopt a PDE-based reinitialization strategy for consistency with earlier works and to facilitate direct comparison with existing results. Following [13] and [164], we reinitialize ϕ using the PDE-based approach proposed in [160] by solving the auxiliary equation

$$\frac{\partial\phi}{\partial\tau} = \operatorname{sign}(\phi)\left(|\nabla\phi| - 1\right) \qquad \forall x \in \Omega, \tag{4.29}$$

for a fictitious time τ until steady state. This equation is integrated using a Total Variation Diminishing second-order Runge–Kutta (TVD-RK2) method [153]. For a detailed review of level set methods and reinitialization strategies, we refer the reader to [127, 152].

One side effect of the PDE-based reinitialization procedure is that it tends to smooth sharp features of the level set function. As noted in [158] and [152], iterating the reinitialization procedure has the effect of perturbing the zero-contour of the level set function, which can diffuse sharp interfaces and lead to mass loss. There are a number of alternative techniques that better preserve sharp features, such as the work in [144], but avoiding reinitialization altogether tends to be one of the best strategies. This, is one of the key benefits with reference map based schemes. Because the map does not require frequent restarts, we can maintain the signed distance nature of ϕ without frequent reinitialization because $\phi(\mathbf{x}, t)$ is constructed directly from the composition of ϕ_0 and the reference map $\boldsymbol{\xi}$.

For the work herein, we follow [13] and [164] (primarily for a fair comparison) and perform systematic reinitialization using the PDE-based scheme at every iteration of the SL and CB schemes. Similarly, we reinitialize the level set function every time we restart the map for the RM, VPRM and RMCB methods. We use the criteria presented in [164] when restarting the reference map, which monitors the bijectivity of the map and performs a restart when the columns of $\nabla \boldsymbol{\xi}$ are nearly colinear. As with the previous example, the characteristics of the solid body rotation field (*i.e.* $\mathbf{u} = (-y, x)$) are well resolved using the reference map approach and no restarting is required for the analytically divergence free field. When the artificial expansion terms are incorporated, however, this is no longer guaranteed.

We measure the L^2 error on the interface using the same approach as Bellotti and Theillard [13],

$$L^2 = \left(\int_{\Gamma_h} \Big(\phi(x, y, 0) - \phi(x, y, 2\pi) \Big)^2 \, ds \right)^{1/2}, \tag{4.30}$$

where Γ_h denotes the reconstructed interface. The difference $\phi(x, y, 0) - \phi(x, y, 2\pi)$ is approximated at grid nodes using standard third-order finite-difference stencils involving neighboring grid points and then interpolated onto Γ_h. The integral along Γ_h is evaluated as a one-dimensional geometric contour integral using the second-order technique of Min and Gibou [113].

Finally, we compute the volume loss by integrating a constant density, $\rho = 1$, over the volume defined as the interior of the interface (*i.e.* $V(t) = \{\mathbf{x} \in \Omega : \phi(\mathbf{x}, t) \leq 0\}$) as follows,

$$V = \left| 1 - \frac{\int_{V(2\pi)} \rho \, dV}{\int_{V(0)} \rho \, dV} \right|. \tag{4.31}$$

The volume integrals are evaluated using the same geometric integration procedure as for the interface integrals, by reconstructing the subdomain $V(t)$ within grid cells intersected by the interface and computing the resulting cell-wise volume contributions using the second-order technique of Min and Gibou [113].

We present the numerical results for each of the tested methods, using a CFL of 5, in Figures 4.12 and 4.13 for interface error and volume loss, respectively. We immediately see that the SL method yields first-order accuracy for the interface error and between first- and second-order accuracy in volume loss. The decrease in accuracy for the SL method is directly related to the frequent interpolation of ϕ and the reinitialization of the interface at each time step. In Figures 4.14 and 4.15, we show how these effects result in a rounding effect at the corners of the slotted circle. The CLSRM avoids the accumulation of these errors by reconstructing the interface at every time step from the initial configuration

avoiding the need for reinitialization. Unfortunately, the high-order accuracy of the CSLRM is significantly degraded when the artificial expansion is applied to the velocity field. Again, the CLSRM method provides no mechanism for filtering compressible modes. Their presence introduces error both through the use of modified characteristics and through the need for frequent remapping, since $\nabla \boldsymbol{\xi}$ frequently triggers the colinear threshold.

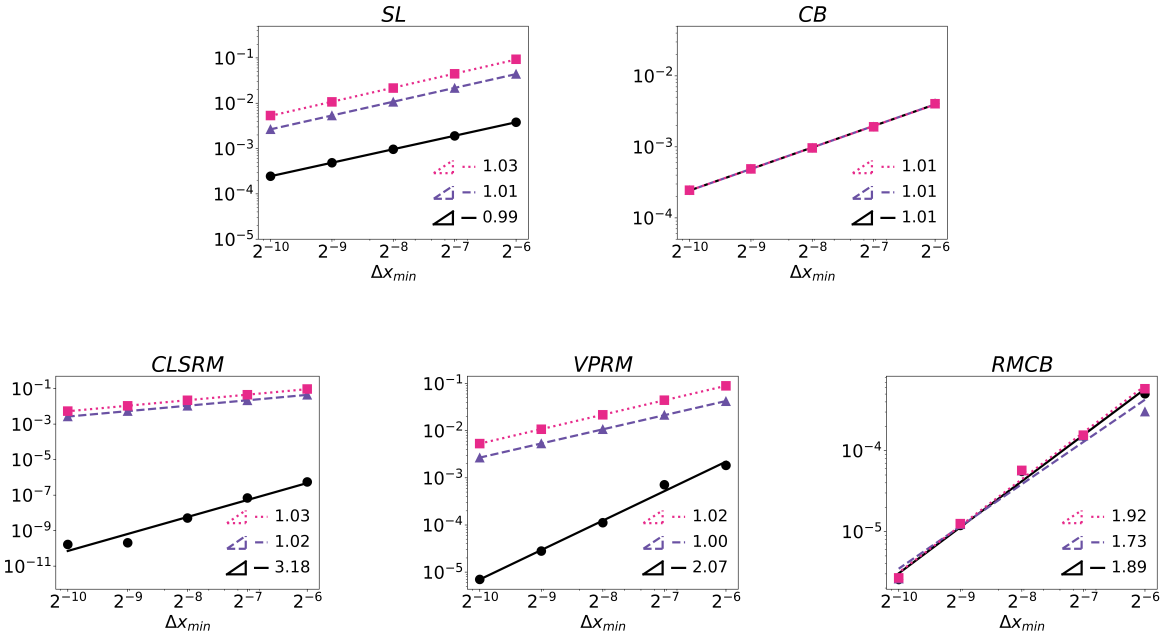


Figure 4.12: Solution error $\|\phi - \phi_e\|_2$ for the Slotted Circle example with various level of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

The VPRM performs slightly worse than the CLSRM for the incompressible case (*i.e.* $\alpha = \beta = 0$). This highlights one of the key findings from [164], where it was shown that the VPRM can result in degraded performance compared with the CSLRM for analytically divergence-free velocities. Essentially, the added volume-preserving projection can perturb the interface. Here, we use a shell[7] size equal to the uniform band

[7]Following Theillard [164], the shell denotes the band around the interface within which the volume-preserving projection is performed. The width of the shell is a tunable parameter that influences both accuracy and stability.

(*i.e.* $|\phi| < 8\Delta x_{min}$) based on the commentary in [164], but this parameter can have a significant impact on the overall performance of the method (as noted in [164]). For example, in the tests conducted here we see that increasing the shell size reduces the accuracy of the method for analytically divergence-free velocities, but improves performance (marginally) when artificial expansion is added to the advecting velocity. Because this parameter is typically chosen based on the assumption that the velocity is incompressible, we present results using the ideal shell size for the divergence-free condition.

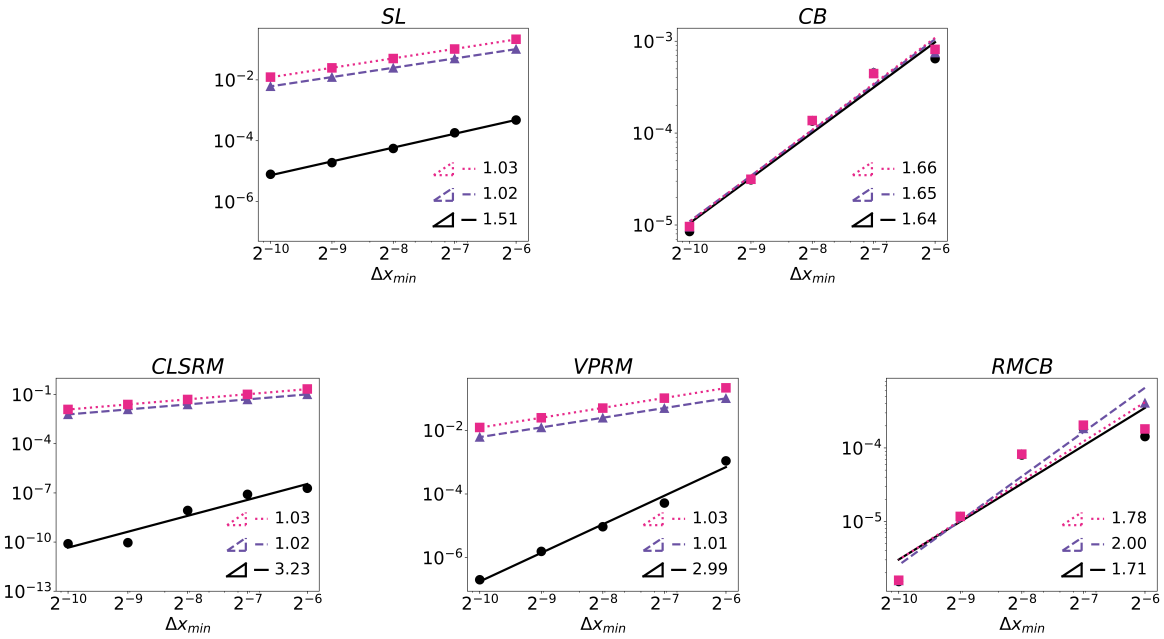


Figure 4.13: Volume loss for the Slotted Circle example with various levels of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

The results for the CB and RMCB methods demonstrate a similar invariance as in the Gaussian example. The CB method yields similar results to the SL method for the divergence-free condition, including the rounding of sharp features (see Figure 4.14). When we incorporate the artificial expansion, the CB method yields nearly identical results to the divergence-free solution. Similarly, the RMCB method yields nearly identical results with and without the addition of artificial expansion. When looking at the

volume loss error in Figure 4.13, we see similar results. The methods using CB retain performance even in the presence of artificial expansion, whereas the other methods degrade in performance. A three-dimensional example of the slotted cylinder is shown in Figure 4.16.

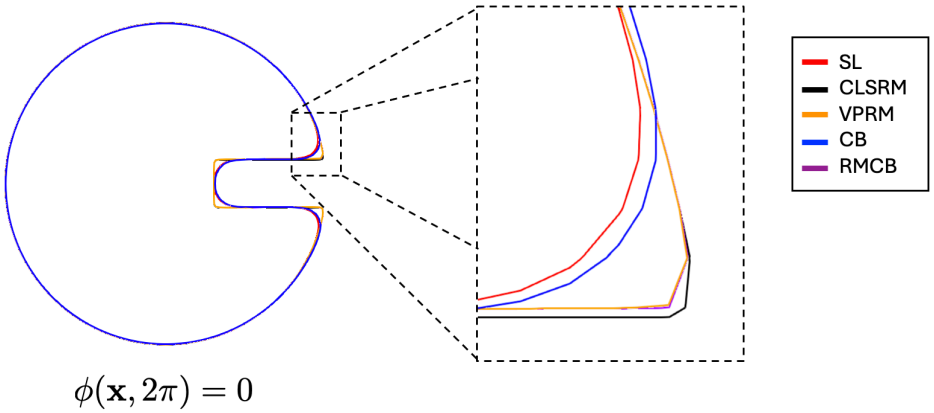


Figure 4.14: Comparison of rounded corners in the Slotted Circle example using each of the different methods and a completely incompressible velocity field. Results are shown using a quadtree grid with a minimum and maximum level of refinement of 3 and 7, respectively. The CLSRM method (black) is visually identical to the exact solution, $\phi(\mathbf{x}, 0) = 0$.

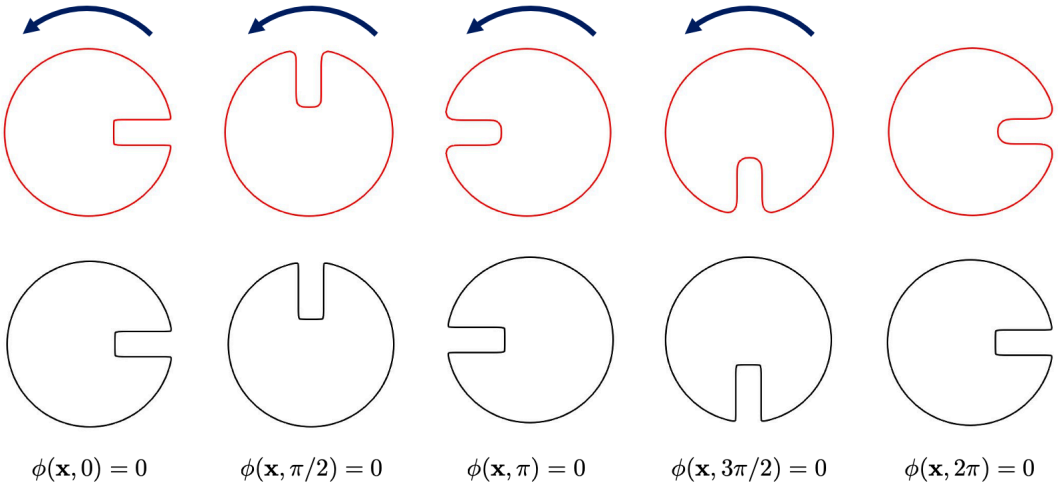


Figure 4.15: Evolution of the interface for the slotted circle example. The single time-step methods (SL and CB) round sharp features of the interface as a result of interpolation and PDE-based reinitialization. The reference map-based methods (CLSRM, VPRM, and RMCB) avoid rounding sharp features by reconstructing the interface using the initial configuration and avoiding frequent reinitialization.

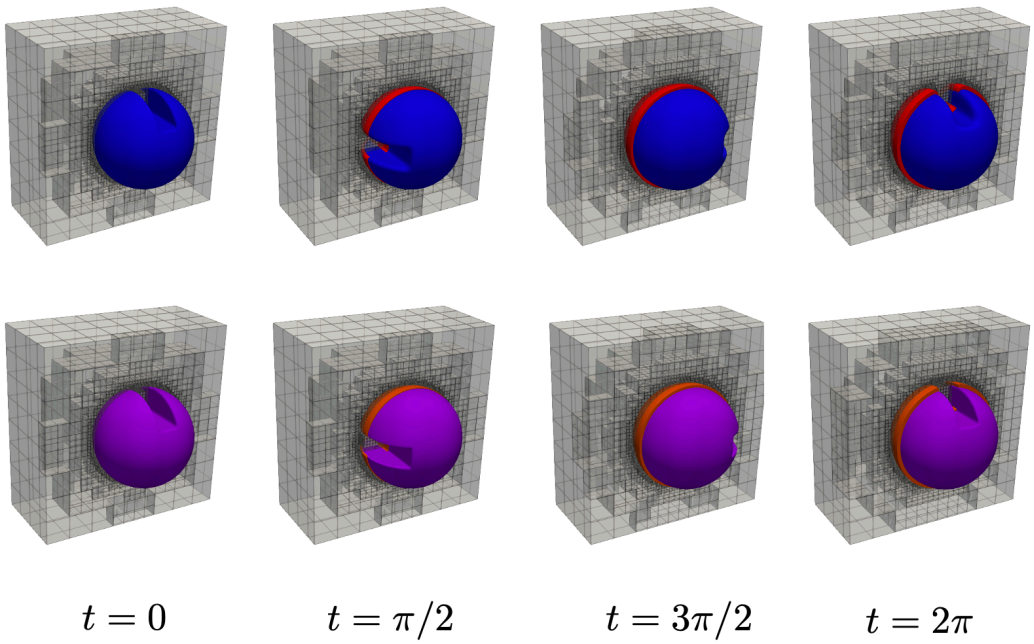


Figure 4.16: Evolution of the interface for a three-dimensional version of the slotted circle example using $\alpha = 1,\ \beta = 0$. A comparison of the SL (red) and CB (blue) methods are shown on the top and a comparison of the VPRM (orange) and RMCB (purple) methods are shown on the bottom at 4 different times. For this example, we see that the CB and RMCB methods retain their shape as the SL and VPRM methods expand. Additionally, we see that the SL and CB methods smooth the corners, whereas the reference map-based methods remain sharp.

A Note on Level Set Reinitialization

As mentioned above, it is possible to use a different reinitialization approach for the level set method. Here, we briefly demonstrate this possibility using the reinitialization scheme proposed in [144] for the SL and CB methods[8].

For this example, we use coarse uniform grids (to magnify error) and compare the interface and volume error between the PDE-based reinitialization and the high-order reinitialization algorithms of [144], implemented in the C++ code of [138]. We use a third-order reinitialization scheme from [138] to match our third-order PDE-based scheme and perform the reinitialization procedure at every iteration (as done with the PDE-based method). Here, we also use a slightly different metric to measure the volume loss. Specifically, we measure the difference between the volume of the level set function at the final time and the exact volume of the slotted circle. This removes potential discretization artifacts resulting from the coarse representation of the level set interface and focuses on the impact of reinitialization on the exact location of the interface.

In Figure 4.17, we can see that the third-order reinitialization algorithm of [144] leads to second-order accuracy in the L^2 norm, whereas the PDE-based reinitialization in 4.29 is first-order for the semi-Lagrangian and characteristic bending schemes, which aligns with the results presented in [144]. The interesting dynamic is that, when using third-order reinitialization algorithm of [144], the volume error for the CB scheme is an order of magnitude lower than when using the PDE-based reinitialization. From these results, we can conclude that the CB method and volume-preservation in general will not correct errors resulting from reinitialization. The method, however, can see auxiliary benefits from using high-order reinitialization schemes. For the remainder of the results presented herein, we continue to use the PDE-based reinitialization scheme to compare with results from the literature. Additionally, for smooth surfaces (which are what we expect to deal with in later applications), the PDE-based reinitialization does indeed see third-order accuracy, as noted in [157].

[8] We only demonstrate the impact of reinitialization with the SL and CB schemes as the reference map-based methods, for this particular example, do not require any reinitialization.

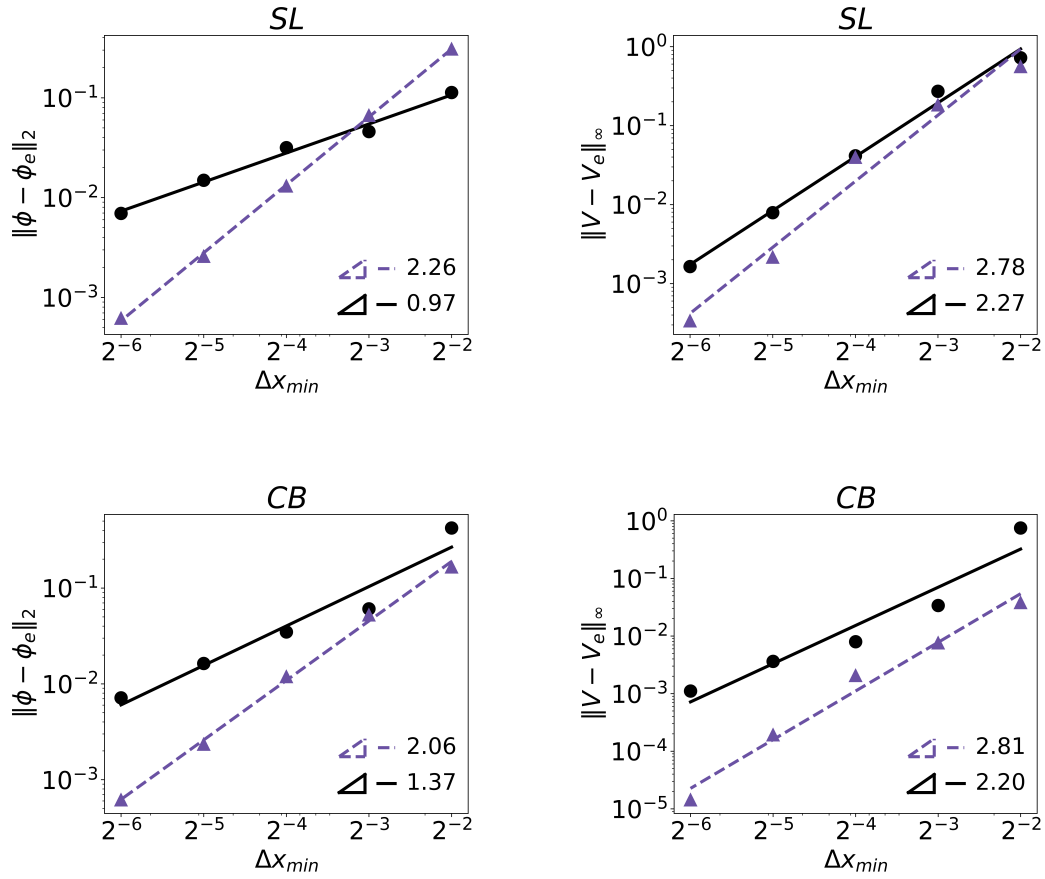


Figure 4.17: Comparison of the PDE-based reinitialization (●) and third-order reinitialization algorithm of [144] (▲) for the level set function in the Slotted Circle example. Results for the solution error are shown on the left and results for volume loss are shown on the right.

Deformational Flow

The purpose of the single vortex example is to explore how the characteristic bending method behaves when subject to a strong deformational flow. This problem was first introduced in [100][9] and is a simple method for measuring the accuracy and robustness of

[9]We also note that the incompressible field used in this example has been used in a variety of other studies (notably in [11]). However, the example in [100] seems to be the first example of reversing the direction of the flow to test conservation and interface preservation properties.

numerical schemes for scalar advection. This example is particularly useful for measuring the distortion of an interface and the conservation properties (or lack thereof) for level set methods. As such, we use this example to highlight the benefits and limitations of the CB scheme.

We consider the advection of an interface defined by the level set function

$$\phi(x, y, 0) = \sqrt{(x - 0.5)^2 + (y - 0.75)^2} - 0.2, \tag{4.32}$$

in the domain $\Omega = [0, 1]^2$. This level set function represents a circle of radius $R = 0.2$ centered at $(0.5, 0.75)$ and we advect this interface using the incompressible velocity field,

$$\mathbf{u}_0(x, y) = \begin{pmatrix} -\sin^2(\pi x)\sin(2\pi y) \\ \sin(2\pi x)\sin^2(\pi y) \end{pmatrix} \tag{4.33}$$

from time $t = 0$ until $t = \pi$. At time $t = \pi$, we reverse the direction of the velocity field by flipping the sign of each component and continue advecting the profile until time $t = 2\pi$. As this process is reversible, the initial profile should be recovered exactly at time $t = 2\pi$ and we can measure the accuracy and conservation properties of our numerical method by measuring the error between the initial profile and computed result.

Similar to the previous two examples, we consider including a compressible component to velocity field. However, we note that sign of the compressible component does not change at time $t = \pi$ as the reversal of this component could potentially negate the effects of the adding artificial expansion. Additionally, we center the compressible component at $(0.5, 0.5)$, leading to the velocity field

$$\mathbf{u}(x, y, t) = \mathbf{u}_0(x, y) + \begin{pmatrix} \alpha\Delta x_{\min}(x - 0.5) + \beta(\Delta x_{\min})^2(x - 0.5) \\ \alpha\Delta x_{\min}(y - 0.5) + \beta(\Delta x_{\min})^2(y - 0.5) \end{pmatrix} \text{ for } t \in [0, \pi] \tag{4.34}$$

and

$$\mathbf{u}(x, y, t) = -\mathbf{u}_0(x, y) + \begin{pmatrix} \alpha\Delta x_{\min}(x - 0.5) + \beta(\Delta x_{\min})^2(x - 0.5) \\ \alpha\Delta x_{\min}(y - 0.5) + \beta(\Delta x_{\min})^2(y - 0.5) \end{pmatrix} \text{ for } t \in (\pi, 2\pi]. \tag{4.35}$$

This velocity field, with $\alpha = \beta = 0$, is entirely divergence-free and, analytically, the characteristic trajectories do not cross in each time region. This may lead one to believe that the CLSRM and VPRM methods would not require restarting. However, in practice, and as noted in [13] and [164], the bijectivity of the map can be lost due to numerical errors for this deforming flow and the reference map requires frequent restarting. Following the

previous example, we reinitialize the level set function every time we restart the reference map. The SL and the CB schemes reinitialize the level set function at each iteration[10].

As in the previous example, we use an adaptive quadtree mesh for this example and use a uniform band equal to $8\,\Delta x_{min}$. We also use the L^2 interface error defined in (4.30) and the volume loss defined in (4.31).

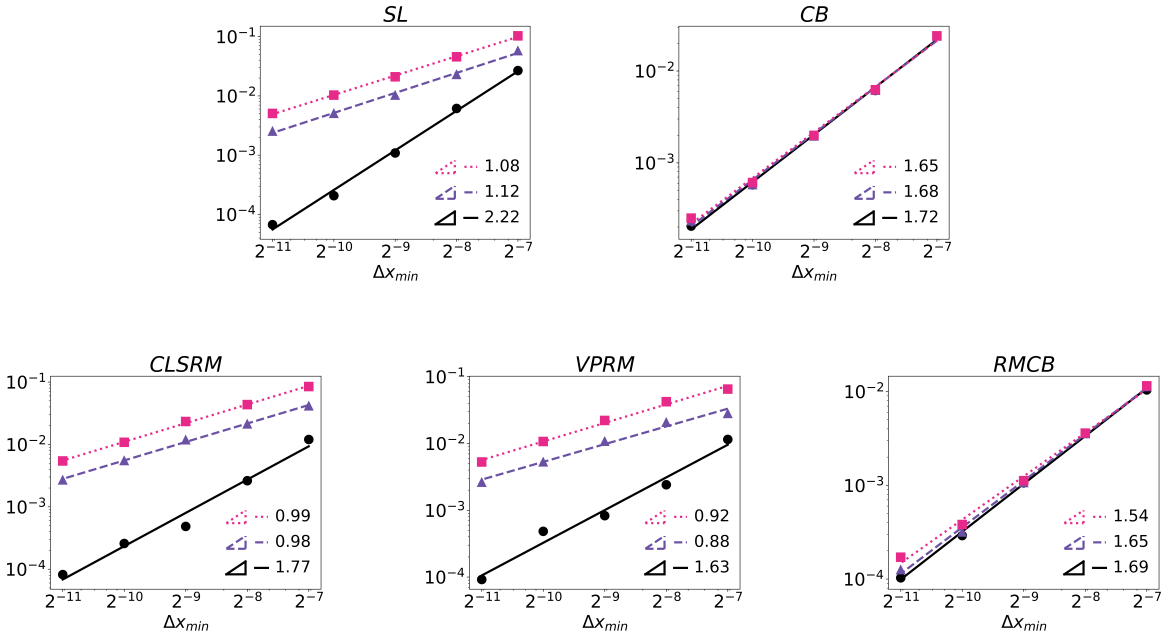


Figure 4.18: Solution error $\|\phi - \phi_e\|_2$ for the Single Vortex example with various levels of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

The results for the SL, CLSRM, VPRM, CB, and RMCB methods using a CFL of 5 are shown in Figures 4.18 and 4.19. As expected, we see between first- and second-order accuracy for the interface error for all of the methods tested. Furthermore, we see second-order convergence for the volume loss with each method. When we add artificial expansion to the velocity field, we observe the same trend noted in the previous two examples. The SL,

[10] As noted in the previous example, it is not strictly required that we reinitialize at every time step. However, we chose this approach for direct comparison with previous studies of Bellotti and Theillard [13] and Theillard [164]

CLSRM, and VPRM schemes all yield accuracy between first- and second-order, whereas the CB and RMCB schemes produce results nearly identical to the purely incompressible case.

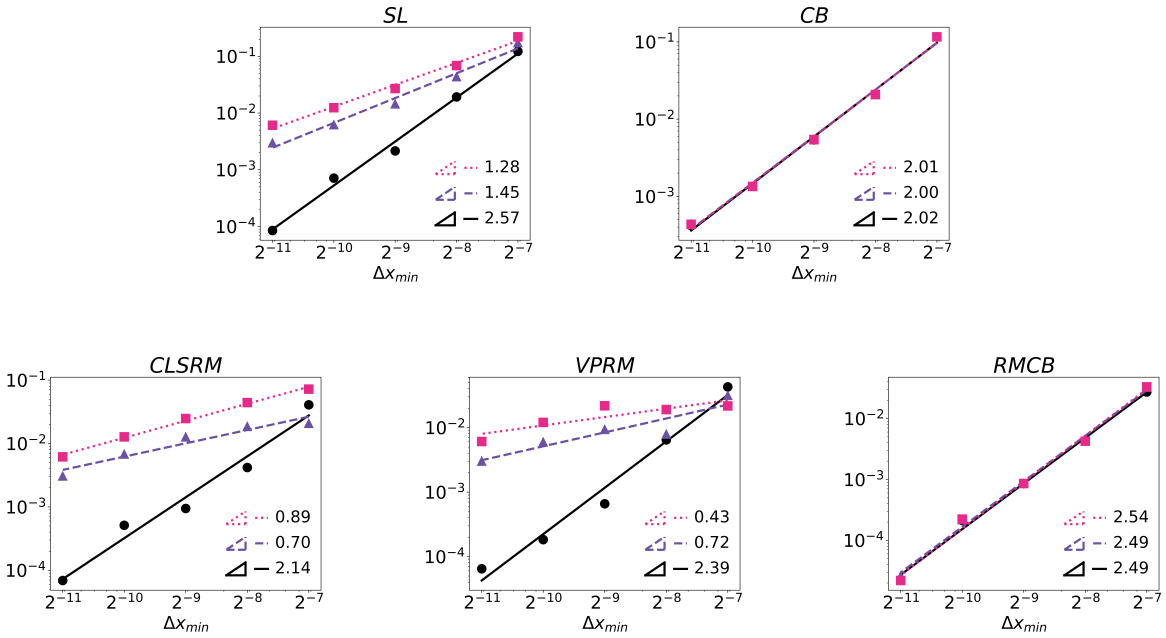


Figure 4.19: Volume loss for the Single Vortex example with various level of artificial expansion added to the velocity field. ● denotes the completely incompressible field, ▲ denotes the second-order expansion added to the incompressible velocity (*i.e.* $\alpha = 0,\ \beta = 1$), and ■ denotes the first-order expansion added to the analytic velocity field (*i.e.* $\alpha = 1,\ \beta = 0$).

Figure 4.20 provides a qualitative comparison of the SL and CB methods using a three-dimensional example from [13], originally proposed by Enright *et al.* [49][11]. Here, we advect the flow forward in time until the level set function loses a significant amount of mass, rather than reversing the velocity field as in the two-dimensional case. This visualization highlights the ability of the CB method to preserve the topology of the level set surface, particularly on coarse grids where the impact of the artificial compression[12] is more pronounced. In Figure 4.21, we show the corresponding loss in volume at the final

[11]The reference map based methods (CLSRM, VPRM, and RMCB) exhibit minimal volume loss for this example and are extensively compared in [13] and [164].

[12]For this example, adding artificial expansion increases the mass of the level set function. So we have used compression to highlight the differences more clearly.

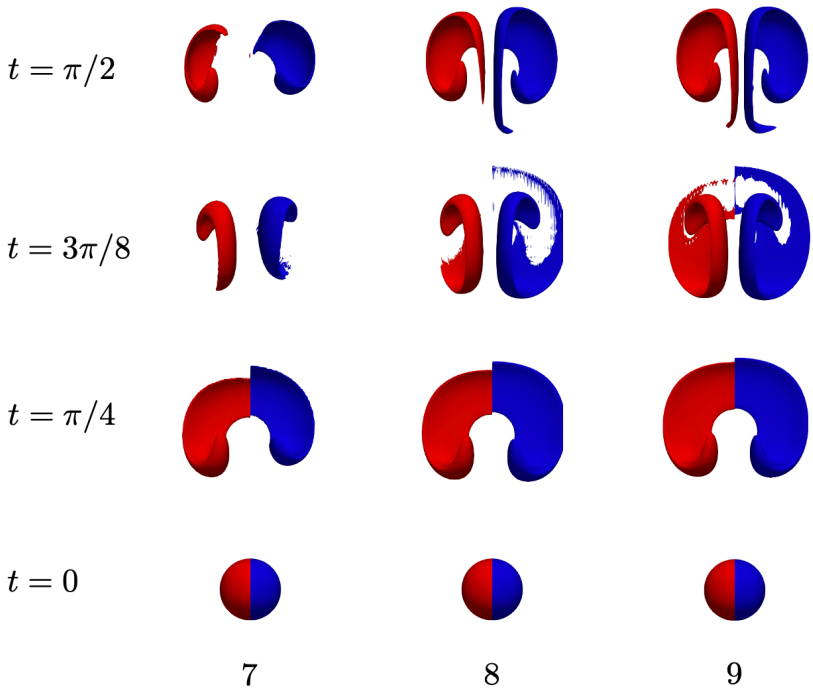


Figure 4.20: Evolution of the interface for a three-dimensional deformational flow proposed by Enright *et al.* [49] using a first-order artificial compression added to the incompressible field (*i.e.* $\alpha = -1,\ \beta = 0$). The SL (red) and CB (blue) methods are shown at four time steps with increasing maximum levels of refinement.

time for both methods using $\alpha = 0$, and -1. In contrast to the SL method, the CB scheme remains largely invariant with respect to the artificial expansion, demonstrating its robustness. Note that the topological deterioration of the interface produces a non-smooth problem, and we do not expect second-order convergence. Nevertheless, these results clearly illustrate the advantages of the CB method in this three-dimensional, deformable flow.

Incompressible Euler

For the final advection example, we consider a simple example of nonlinear advection in the form of the incompressible Euler equations,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p. \tag{4.36}$$

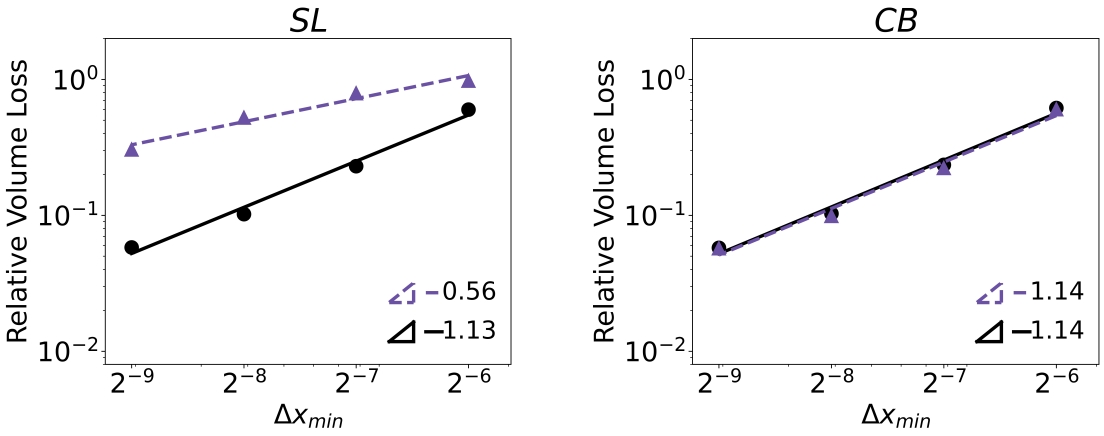


Figure 4.21: Volume loss for the three-dimensional deformational flow proposed by Enright *et al.* [49] using various levels of artificial expansion added to the velocity field. ● denotes the completely incompressible field and ▲ denotes a first-order compression added to the incompressible velocity (*i.e.* $\alpha = 1,\ \beta = 0$).

Specifically, we construct an analytical solution to (4.36) using the velocity field,

$$u(x, y) = \begin{cases} \sin(x)\cos(y) \\ -\cos(x)\sin(y) \end{cases} \tag{4.37}$$

and compute the corresponding pressure gradient, ∇p. We use the domain $\Omega = [-\pi, \pi]^2$ and prescribe Dirichlet boundary conditions at the boundaries. This yields a stationary problem, where the exact solution is simply the initial velocity field for all time. Numerically, we solve this system using the standard SL formulation by integrating the right-hand side, $-\nabla p$, along characteristic curves in the Lagrangian frame, following [24, 86].

The goal of this test is twofold. First, it provides a manufactured nonlinear problem in which to compare the CB scheme with the standard SL method. Second, it highlights how the CB scheme behaves in a Navier–Stokes setting. This problem is set up as the advection step in a projection method and by iterating the advection step, we can observe how error accumulates and how the divergence of the velocity field is affected.

We run this problem on a uniform grid with refinement levels ranging from 3 to 7, and use CFL-based time stepping with a CFL of 3. Recall that the semi-Lagrangian method is unconditionally stable for linear advection problems. However, this property does not automatically extend to nonlinear problems (see [51]) and, for this problem, large CFL

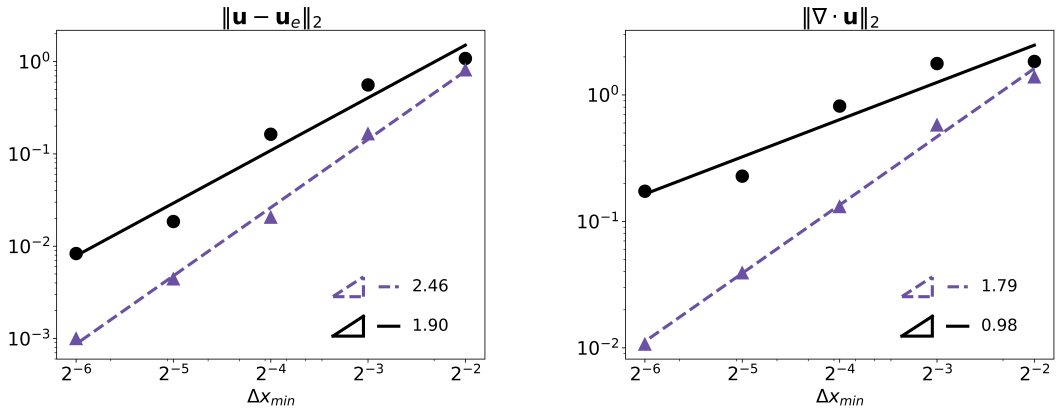


Figure 4.22: Comparison of the SL (●) and CB (▲) methods for the incompressible Euler example. Velocity error is shown on the left and the divergence is shown on the right.

numbers may cause the solution to blow up. Empirically, we found that both SL and CB remain stable with a CFL of 3 when combined with an RK2 backward time integrator and WENO-based quadratic interpolation.

The results are summarized in Figure 4.22 using a CFL of 3. We report errors at a final time $t = 8\pi$, corresponding to the point at which oscillations in the velocity error and divergence are stabilized. The SL method shows between first- and second-order convergence for the velocity field, and first-order is observed in the divergence. By contrast, the CB method exhibits clear second-order convergence in the velocity and nearly second-order convergence for the divergence. Notably, the CB method preserves incompressibility, which is consistent with its underlying volume-preserving formulation.

We also observed during testing that the CB method remains stable with higher-order interpolation, while the SL scheme quickly becomes unstable and diverges from the exact solution. This effect is most pronounced with non-symmetric stencils (e.g. Taylor polynomials rather than bicubic or tricubic interpolation, as implemented in [144]). In these cases, oscillations introduced by high-order interpolation appear to be smoothed by the volume-preserving correction, which aligns with the commentary in [164]. For comparability with prior studies, we restrict results here to second-order interpolation, but note this as a promising direction for future analysis.

Finally, we acknowledge that this test somewhat favors the CB method. Unlike the SL method, which has no mechanism to enforce incompressibility, the CB method effectively performs a divergence-free projection in the Lagrangian frame. That said, there are extensions of the SL method that address this limitation (e.g., conservative formulations

such as [96, 190]). Another possible solution is to reformulate the SL method in terms of the reference map framework and then use a Piola transform (see *e.g.*, [21]) to ensure that the transformation preserves divergence-free fields. Nevertheless, the results clearly illustrate the benefit of the characteristic bending method for maintaining a divergence-free velocity field.

4.4.2 Incompressible Multiphase Navier-Stokes

In this section, we test the characteristic bending scheme in the context of two-phase, incompressible flows. Specifically, we compare the CB implementation against the SL method in an analytic setting and then in a practical example of rising bubbles. We explain where the CB method can be incorporated into a multiphase projection method and show how these changes impact both the solution and the robustness of the modified solver.

Governing Equations

The fluid velocity and pressure are modeled by the incompressible Navier-Stokes equations[13],

$$\left.\begin{aligned} \rho_i \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) &= -\nabla p + \mu_i \Delta \mathbf{u} + \mathbf{f} \\ \nabla \cdot \mathbf{u} &= 0 \end{aligned} \right\} \text{ in } \Omega_i, \tag{4.38}$$

subject to the interfacial jump conditions,

$$\left.\begin{aligned} [\mathbf{u}] &= 0 \\ [\sigma \cdot \mathbf{n}] &= \gamma \kappa \mathbf{n} + \mathbf{q} \end{aligned} \right\} \text{ on } \Gamma, \tag{4.39}$$

where,

$$\sigma = -p\,\mathbf{I} + \mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T \right) \tag{4.40}$$

is the stress tensor, κ is the curvature of the interface, and γ is the surface tension. The jump operator, $[\,\cdot\,]$, is defined as $[\chi] = \chi^+ - \chi^-$ and represents the jump of the quantity χ across the interface Γ. We use $\mathbf{f}$ and $\mathbf{q}$ to capture any additional forcing terms acting on the fluid or at the interface, respectively. Finally, we use the standard convention where the $-$ superscript represents quantities in the interior of the interface and the $+$ superscript represents quantities in the exterior of the interface.

[13] Note, we have chosen to omit the $\pm$ on the fluid variables for the clarity of presentation.

Overview of the Navier-Stokes Solver

In order to solve the two-phase, incompressible Navier-Stokes equations, we use the solver developed in [17] and presented in Chapter 3. This solver employs a collocated, node-based framework on adaptive quadtree (2D) and octree (3D) grids, where all the fluid variables are stored at the grid nodes. This choice of variable arrangement greatly simplifies the design of data structures and algorithms for the Navier-Stokes solver, which makes it an ideal testbed for the characteristic bending advection schemes.

We modify the solver by replacing the SL method used in the SLBDF scheme with the CB method and by replacing the VPRM scheme used to advance the fluid interface with the RMCB scheme. The SLBDF scheme is used to advanced the momentum equation by discretizing the equation along the characteristics of the velocity field, treating the nonlinear advection term explicitly, and then uses a backward differentiation formula to handle the diffusive terms implicitly. The explicit treatment of the advection term is done entirely using the SL method and thus can be replaced with the CB scheme. The evolution of the interface in the previous chapter is done using the VPRM approach for which the RMCB method can be used as in the previous advection examples.

Analytic Vortex

As in Chapter 3, we verify the convergence for the full two-phase solver using the standard, analytic vortex problem (see [167, 17]). We solve the complete incompressible, two-phase Navier-Stokes equations, Eq. (4.38) - (4.39), given the exact analytic solution,

$$\mathbf{u}^{\pm}(x,y,t) = \begin{cases} \sin(x)\cos(y)\cos(t) \\ -\cos(x)\sin(y)\cos(t) \end{cases} \tag{4.41}$$

$$p^{\pm}(x,y,t) = 0, \tag{4.42}$$

with the interface, Γ, defined by the level set function,

$$\phi(x,y) = 0.1 - \sin(x)\sin(y). \tag{4.43}$$

We then set the forcing terms, $\mathbf{f}^{\pm}(x,y,t)$ and $\mathbf{q}(x,y,t)$, as

$$\mathbf{f}^{\pm}(x,y,t) = \rho^{\pm}\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) - \mu^{\pm}\Delta\mathbf{u} \tag{4.44}$$

$$\mathbf{q}(x,y,t) = -\gamma\nabla\cdot\frac{\nabla\phi}{|\nabla\phi|} + \left[\mu\nabla\mathbf{u}\cdot\frac{\nabla\phi}{|\nabla\phi|}\right]. \tag{4.45}$$

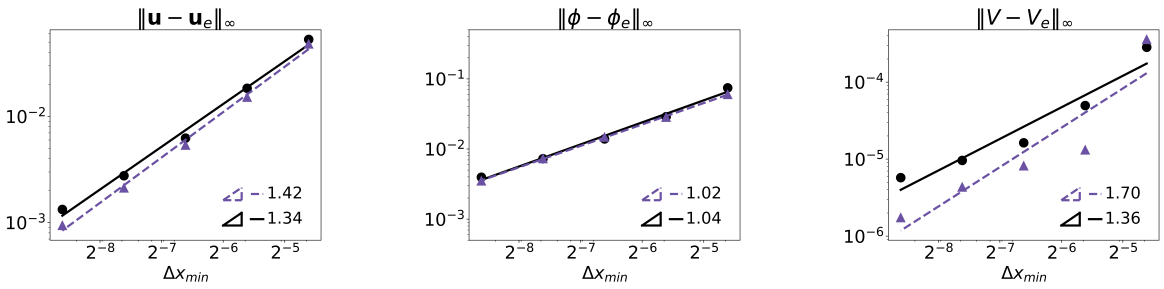


Figure 4.23: Comparison of the SL with VPRM (●) and CB with RMCB (▲) methods for the two-phase, analytic vortex example. Velocity error is shown on the left and the interface error is shown on the right. For this example, the SL method is coupled with the VPRM for the advection of the interface. For the CB scheme, the RMCB is used to advect the interface.

Finally, we set the remaining parameters as,

$$\mu^+, \rho^+ = 1, \qquad \mu^-, \rho^- = 10, \qquad \gamma = 0.1, \qquad \Omega = \left[-\frac{\pi}{3}, \frac{4\pi}{3}\right]^2, \qquad t_{\text{final}} = \pi. \tag{4.46}$$

We prescribe no-slip boundary conditions on the walls of the computational domain, Ω, and use an adaptive quadtree mesh with a span of 4 between the minimum and maximum refinement levels. For this particular example, we refine only near the interface using the criteria specified in Section 3.2.4 (*i.e.* no vorticity-based refinement is used).

In Figure 4.23, we present the results using the formulation in Chapter 3 (SL and VPRM) and our modified version of the solver using the combination of CB and RMCB. We see that our modified solver yields a slight improvement in the accuracy of the solution and the interface position. This slight improvement is expected as the interface for this particular example does not undergo large deformations and the velocity field should remain largely free of compressible modes. We do see an improvement in the volume loss, which is due to the subtle treatment of the correction in RMCB. However, the key importance of these results is that the usage of the CB/RMCB modification improves the solver in Chapter 3.

Rising Bubbles

For the final example, we consider rising bubbles in two dimensions. These examples were presented in Chapter 3 and are inspired by the experimental work in [16]. Here, we

consider Cases *a*, *e*, and *f* to serve as a qualitative demonstration of the differences between each of the different advection schemes presented herein (e.g., SL, RM, VPRM, CB, and RMCB). As in the previous example, we compare the SL scheme with CB in the SLBDF method and the CLSRM and VPRM methods with RMCB for interface evolution.

We consider an initially circular bubble suspended in another denser and more viscous fluid, where the density difference between the two fluids induces a buoyancy force causing the bubble to rise and deform. The dynamics of these rising bubbles are described by the non-dimensional Morton (Mo), Eötvös (Eo), and Reynolds (Re) numbers, defined as

$$Mo = \frac{g(\mu^-)^4}{\rho^-\gamma^3}, \quad Eo = \frac{gd^2\rho^-}{\gamma}, \quad \text{and} \quad Re = \frac{\rho^- U d}{\mu^-},$$

where U is the asymptotic rising velocity measured at the tip of the bubble, g is the acceleration due to gravity, and d is the initial diameter of the undeformed bubble. Cases a, e and f in [16] are defined using these dimensionless numbers and we summarize these parameters in Table 4.2.

Table 4.2: Dimensionless parameters, Mo, Eo, and Re, for the Bhaga-Weber Cases a, e and f [16].

Case	a	e	f
Mo	711	4.63×10^{-3}	8.20×10^{-4}
Eo	8.67	115	237
Re	7.80×10^{-2}	94	259

In order to simplify the presentation of our results, we set the rising velocity to $U = 1$ and the initial bubble diameter to $d = 4$, and then define the remaining simulation parameters as,

$$\rho^- = 1, \quad \frac{\rho^-}{\rho^+} = 10^3, \quad \mu^- = \frac{\rho^-}{Re}, \quad \frac{\mu^-}{\mu^+} = 10^2, \quad \gamma = \frac{(\mu^-)^2}{\rho^-}\sqrt{\frac{Eo}{Mo}},$$

and

$$g = \frac{(Mo)\rho^-\gamma^3}{(\mu^-)^4}.$$

We use the computational domain $\Omega = [-8, 24]^2$ and initialize the bubble at $\mathbf{x} = (0, 0)$. We chose the distance from the walls to minimize boundary effects and define no-slip boundary conditions at all but the top wall. On the top wall (*e.g.* $\mathbf{x} = (x, 24)$), we impose a no-flux

boundary condition, which allows the buoyancy-driven flow to leave the domain without introducing boundary effects that would affect the bubble dynamics. For each case, we run the simulation until the bubble reaches its asymptotic shape and rising velocity or until the interface ruptures. Similar to the previous studies, we use a minimum refinement level of 4 and adjust the maximum level of refinement to highlight differences between the advection schemes.

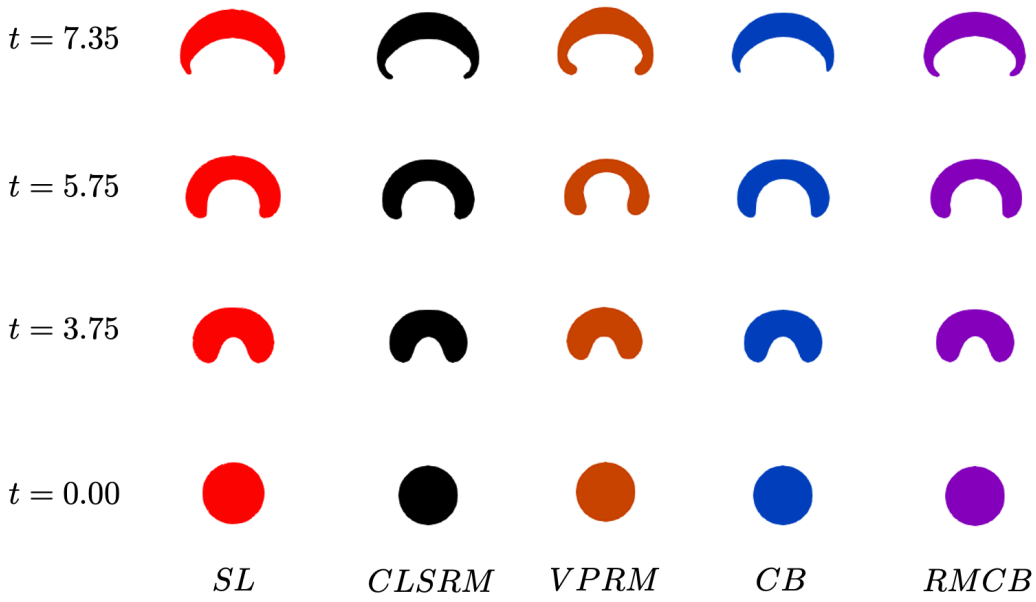


Figure 4.24: Evolution of a rising bubble in two-dimensions for Bhaga-Weber Case *e* using different advection schemes. Results are shown for a minimum and maximum refinement level of 3 and 8, respectively.

In Figure 4.24, we present the bubble shape for Bhaga-Weber Case e using a maximum refinement level of 8. This is a relatively coarse grid (*e.g.*, compared to the grids used in [17]), but this resolution highlights the major differences for each of the advection schemes. The most prominent difference for these simulations is with the skirt of the bubble (trailing edge). With the SL advection, we can see the effects of reinitialization with slightly truncated corners, similar to the effects present in the slotted circle example. This effect is slightly less pronounced with the CB method, but both the CLSRM and RMCB schemes well-preserve the sharp corners. If we focus on the VPRM scheme, we can clearly see that the volume-preserving projection results in significant deformation of the bubble shape. These discrepancies largely disappear as the refinement

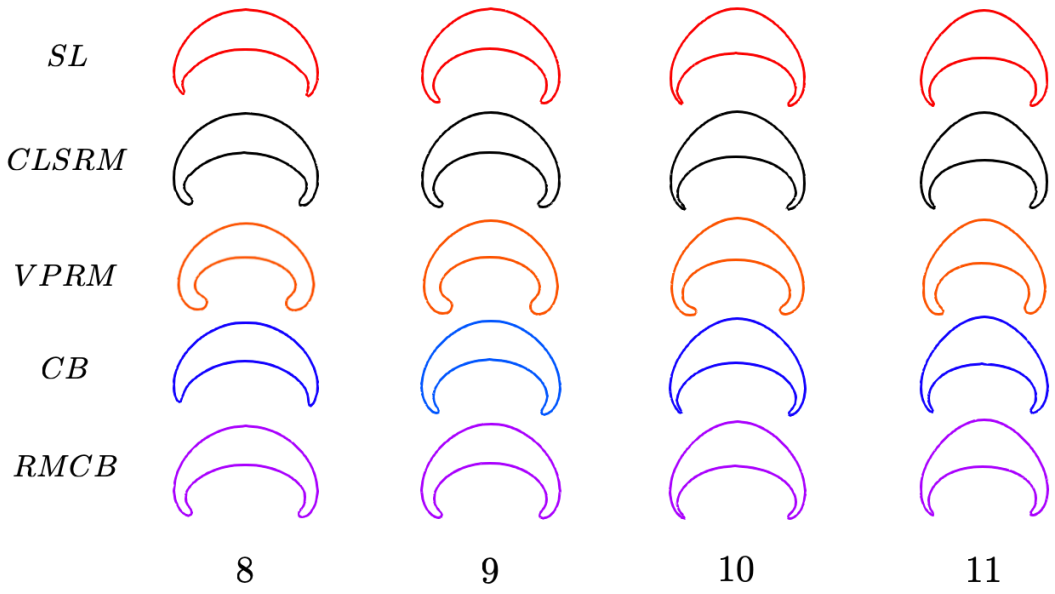


Figure 4.25: Convergence of the final interface profile for the Rising Bubble example using the parameters from Bhaga-Weber Case e. Interface profiles for each of the advection schemes (SL, RM, VPRM, CB, and RMCB) are shown at maximum refinement levels of 8, 9, 10, and 11. Here, we can see that interface profiles converge as the grid is refined and the difference between each of the methods is largely eliminated.

of the grid increases. We show how the interface profiles converge for this example in Figure 4.25.

Figure 4.26 presents the relative mass loss for each advection scheme using Case *a* from [16]. We highlight the mass loss using Case a, because this example exhibits only minor interface deformation and effectively showcases the benefits of the reference map-based methods. Additionally, the parameters of Case a closely align those presented in the mass loss tests presented in previous studies (*e.g.*, [164]).

We observe that the SL and CB methods produce nearly identical results, with the CB scheme yielding a slight reduction in mass loss. A more pronounced distinction emerges when comparing the VPRM and RMCB methods. The RMCB scheme offers a clear improvement in relative mass loss and generates noticeably smoother solutions. Comparisons among the SL, CLSRM, and VPRM schemes reveal trends consistent with those reported in [164]. This example therefore helps to situate the CB and RMCB methods within the existing literature.

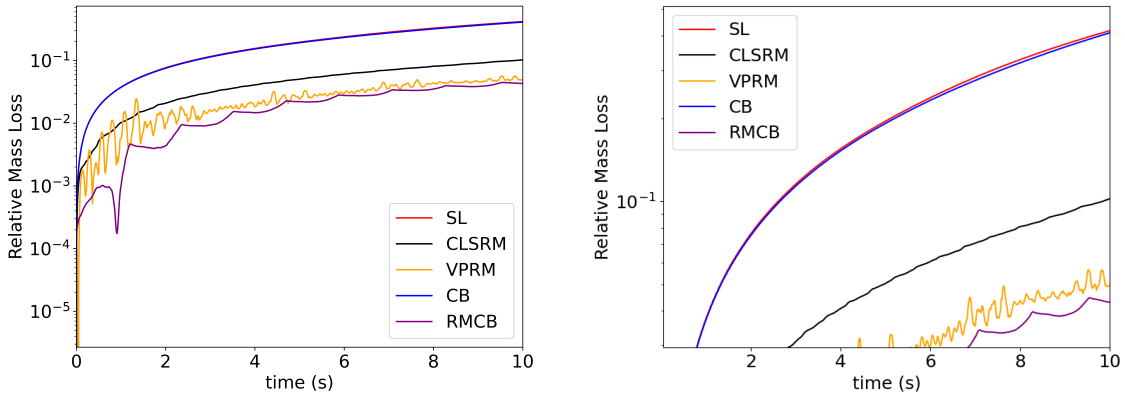


Figure 4.26: Relative volume loss for each of the different advection schemes (SL, RM, VPRM, CB, and RMCB) with a maximum refinement level of 8 for Case *a* of [16]. The right panel provides an enlarged view of the left figure, specifically highlighting the subtle differences between the SL and CB methods.

In Figure 4.27, we present a comparison of the SL, CB, VPRM, and RMCB schemes for Bhaga-Weber Case f using a maximum refinement level of 9. As noted in Chapter 3, Case f presents some challenges for the multiphase solver and we can highlight the early rupture of the interface using this two-dimensional example. Comparing the SL and CB schemes, we can see that the CB method preserves the interface shape longer than the standard SL method. We do expect this improvement as the CB is equipped with the volume-preserving projection. Focusing on the comparison of the VPRM and RMCB methods, we see that the RMCB implementation results in a significant improvement in retaining the shape of the bubble, whereas the VPRM method deforms the bubble profile. We also see that the VPRM moves mass into the concave locations of interface thickening the skirt. The notable takeaway from Figure 4.27 is that the RMCB scheme better preserves the interface of the rising bubble, whereas the VPRM scheme artificially adds mass in concave regions.

4.5 Conclusions

The characteristic bending (CB) method extends the foundational ideas of the Coupled Level Set Reference Map (CLSRM) [13] and the Volume-Preserving Reference Map (VPRM) [164] to effectively remove spurious compressible modes and accurately solve advection problems in incompressible flows. By reformulating the volume-preserving projection of [164] to operate on maps close to the identity, we obtain a generalized

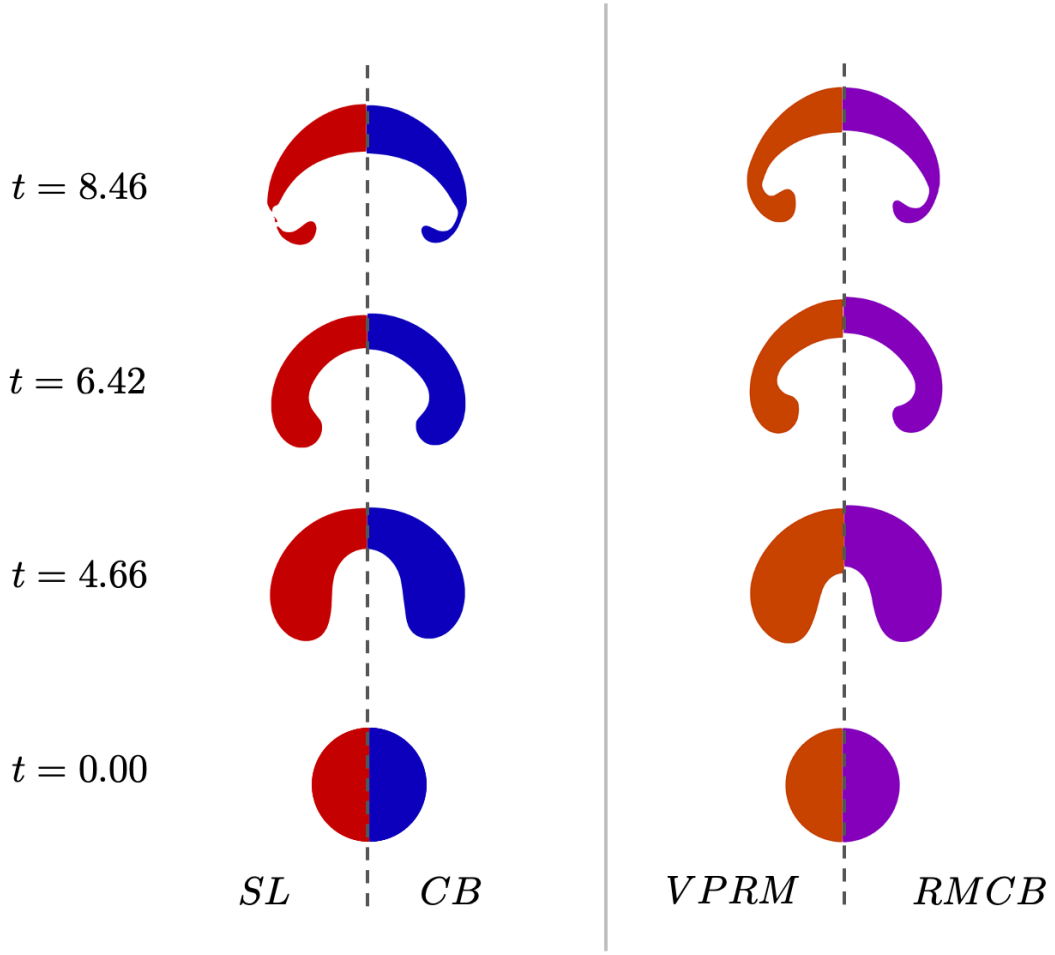


Figure 4.27: Comparison of characteristic bending advection with the traditional semi-Lagrangian method and the Volume-Preserving Reference Map [164] for Bhaga-Weber case f in two-dimensions. Results are shown for a minimum and maximum level of 4 and 9, respectively.

algorithm that improves stability and robustness across a range of test cases. Our results demonstrate that the CB method maintains accuracy even in the presence of artificial compressible modes and performs particularly well in preserving interface geometry for highly deformational flows.

Beyond scalar advection, the CB framework also performs well for nonlinear systems. In the incompressible Euler and multiphase Navier–Stokes examples, the advecting

velocity fields contain only the errors introduced by the underlying spatial and temporal discretizations. Nevertheless, the CB method yields noticeable improvements. In the Euler case, it enhances accuracy and better preserves the divergence-free structure of the velocity field. In the Navier–Stokes simulations, the RMCB approach (the CLSRM method of [13] augmented with CB) reduces mass loss while maintaining a sharp interface. This is a notable improvement over the VPRM approach in [164], which excels at preserving mass but can distort the interface.

A potential drawback of the CB scheme is the additional computational expense associated with solving a Poisson equation at each time step, on top of the underlying SL or CLSRM method. The resulting linear systems are typically well-conditioned, and both preconditioned Krylov methods and multigrid solvers perform efficiently in practice. For the simple advection problems considered herein, the use of CB increases the wall-clock solution time by a factor of approximately 2–5, which is consistent with the observations reported by Theillard [164]. This overhead is comparable to the cost of increasing the grid resolution by a single refinement level (*i.e.*, a factor of four increase in degrees of freedom in two dimensions, or eight in three dimensions). However, as demonstrated in the examples presented above, when compressible modes are present in the advecting velocity field, CB-based schemes consistently outperform non-CB schemes, even when the latter are run at higher spatial resolution.

In the multiphase Navier–Stokes setting, the additional cost of using CB is often significantly smaller than that of a single viscous or pressure solve, making the expense of projecting the reference map relatively modest. Moreover, improved interface preservation can lead to better-conditioned linear systems for the viscous and pressure solves, partially offsetting the additional cost. Nevertheless, when the advecting velocity field is already well resolved and contains minimal compressible error, CB provides limited additional benefit and may not justify the extra computational expense.

There are several promising research directions for this work. A natural next step is to explore more complex advection problems, including additional nonlinear cases and further examples involving the incompressible Euler equations, such as magnetohydrodynamic (MHD) flows as in [187]. In the multiphase context, it would be valuable to examine how the CB method influences the dynamics of interacting interfaces, for instance, in simulations of merging bubbles, and to assess its performance in flows with variable density or strong stratification (*e.g.*, [107]). Finally, an exciting direction is to investigate whether characteristic bending can be applied to compressible flows, where the reference map framework might be leveraged to bend characteristics around shocks and rarefactions in a physically consistent way. In such extensions, the modest added expense of the Poisson solve may be significantly lower than that of ALE remapping.

We conclude by highlighting that the CB method presented herein is a general

approach to solving transport problems with incompressible fields. Throughout this work, we emphasized the similarities of our approach to earlier methods and highlight the key reasons why our formulation is well-suited to filtering out artificial or spurious compressible modes in the advecting velocity. By recasting the classical semi-Lagrangian method in a reference map setting, we are able to leverage a volume-preserving projection to bend the characteristics of the velocity field, yielding an accurate and robust advection scheme. We show how the CB method can be used as a drop in replacement for traditional semi-Lagrangian schemes and how CB can be used to augment existing reference map-based formulations, demonstrating that CB is a flexible approach for constructing accurate and stable methods for incompressible advection. CB provides a unified perspective that builds on previous work and offers a practical foundation for developing more accurate and robust algorithms for incompressible flows.

Chapter 5

Conclusions

This dissertation developed a unified framework for simulating incompressible flow on adaptive grids through a sequence of projection-based numerical methods. Beginning with the formulation of a stable nodal projection method on octree grids in Chapter 2, the node-based collocated framework was introduced and the stability and accuracy of the Navier-Stokes solver were demonstrated through both analytical tests and canonical benchmark problems. Next, the single-phase framework was extended to solve multiphase flows in Chapter 3, where the collocated variable arrangement simplified the development of a hybrid finite-volume/finite-difference scheme to treat interfacial jump conditions in an entirely sharp manner. Finally, in Chapter 4, a novel structure-preserving method for bending the characteristics of an incompressible flow was introduced to address multiple sources of numerical error. Together, each contribution demonstrates how projection-based methods can serve not only as numerical tools to enforce constraints, but as a broader design philosophy for building accurate and extensible solvers.

The Stable Nodal Projection Method [20] established the foundation of this framework by introducing a collocated variable arrangement on adaptive quad- and octree grids. This method removed the need for staggered data layouts, reducing implementation complexity while retaining numerical stability and improving accuracy. By connecting the collocated and staggered formulations analytically, the method provided theoretical justification for stability and verified its convergence through canonical incompressible flow benchmarks. This chapter showed that collocated, adaptive methods can achieve second-order accuracy and robustness even at high Reynolds numbers and around complex geometries.

Building on that foundation, the Sharp Collocated Projection Method for Immiscible Two-Phase Flows [17] extended the framework to handle discontinuities in density and viscosity and to capture interfacial stress jump conditions. Using a hybrid finite-volume/finite-difference discretization, this formulation treated the full stress tensor

monolithically, eliminating iterative velocity corrections common in earlier two-phase solvers. The resulting method accurately captured interfacial deformation, maintained stability under large density ratios, and demonstrated strong agreement with both analytic solutions and experimental benchmarks. Together, Chapters 2 and 3 established a general-purpose, node-based framework for incompressible flow capable of addressing a wide range of single- and multi-phase problems.

Finally, the Characteristic Bending method [18] extended the projection concept from the numerical enforcement of incompressibility to the preservation of geometric structure. By interpreting the semi-Lagrangian advection process through the reference map framework, this method introduced a volume-preserving projection that "bends" characteristics toward the divergence-free space. This approach systematically filters compressible modes in the advecting velocity, mitigating mass loss and preserving interface geometry in complex, time-dependent flows. The characteristic bending method addresses multiple sources of numerical error, including those arising from the use of approximate projections, and was demonstrated to be a robust, general-purpose advection scheme.

Collectively, these developments demonstrate that projection methods provide a cohesive and extensible foundation for simulating incompressible flows using adaptive meshes. The approaches developed herein reduce implementation barriers, promote stability across a variety of boundary and interfacial conditions, and offer new avenues for coupling fluid dynamics with other physical systems. More broadly, they highlight how geometric and analytic principles can guide the construction of algorithms that are both robust and computationally efficient.

Looking ahead, several directions naturally follow from this work. The collocated framework provides a flexible basis for incorporating fluid–structure interaction (FSI) methodologies to extend its application space. By coupling the projection-based incompressible flow solver with models for rigid or deformable solids, the framework could be applied to a wider class of problems in engineering and biological systems.

Future development of the multiphase solver should focus on addressing the accuracy limitations that arise from temporal splitting and interfacial coupling. A promising avenue is to reformulate the two-phase methodology within a predictor–corrector framework, similar to the interfacial schemes presented by Saye [147, 145]. Such an approach could eliminate the need for iterative refinement to enforce boundary conditions, reduce the magnitude of parasitic currents at the fluid interface, and enhance spatial and temporal accuracy. Similarly, the predictor–corrector approach could improve the accuracy of general advection with semi-Lagrangian based methods (including Characteristic Bending) as noted in [76]. Beyond method development, the multiphase solver can be applied to the study of droplet retention in stratified environments, extending prior work

and supporting physical experiments [107].

Finally, it would be exciting to explore how the concepts introduced through Characteristic Bending might be generalized to compressible flow. In this context, "bending" the characteristics could mean aligning them with shocks and rarefactions, providing a new geometric perspective. The reference map itself could be used as an implicit indicator of shock formation, potentially eliminating the need for explicit shock tracking. Furthermore, correction of the map via a projection could be reformulated as an interfacial jump problem to align or bend characteristics toward the physically admissible viscosity solution. Combining these ideas with principles from Arbitrary Lagrangian–Eulerian (ALE) formulations could lead to backward-in-time ALE schemes that avoid common challenges in preserving conservation and mesh quality during remap or remeshing procedures.

In summary, this dissertation presents projection-based methods as both a computational mechanism and a conceptual framework for modeling incompressible flows. From enforcing divergence-free constraints to preserving the geometric structure of advection, the methods developed herein advance the numerical treatment of incompressibility on adaptive grids, offering a pathway toward more accurate, flexible, and physically consistent simulations of complex fluid flows.

Bibliography

[1] Mridul Aanjaneya, Saket Patkar, and Ronald Fedkiw. “A monolithic mass tracking formulation for bubbles in incompressible flow”. In: *Journal of Computational Physics* 247 (2013), pp. 17–61.

[2] David Adalsteinsson and James A Sethian. “The fast construction of extension velocities in level set methods”. In: *Journal of Computational Physics* 148.1 (1999), pp. 2–22.

[3] Ann S Almgren, John B Bell, Phillip Colella, Louis H Howell, and Michael L Welcome. “A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations”. In: *Journal of Computational Physics* 142.1 (1998), pp. 1–46.

[4] Ann S Almgren, John B Bell, Phillip Colella, and Tyler Marthaler. “A Cartesian grid projection method for the incompressible Euler equations in complex geometries”. In: *SIAM Journal on Scientific Computing* 18.5 (1997), pp. 1289–1309.

[5] Ann S Almgren, John B Bell, and William Y Crutchfield. “Approximate projection methods: Part I. Inviscid analysis”. In: *SIAM Journal on Scientific Computing* 22.4 (2000), pp. 1139–1159.

[6] Ann S Almgren, John B Bell, and William G Szymczak. “A numerical method for the incompressible Navier-Stokes equations based on an approximate projection”. In: *SIAM Journal on Scientific Computing* 17.2 (1996), pp. 358–369.

[7] Ryoichi Ando and Christopher Batty. “A Practical Octree Liquid Simulator with Adaptive Surface Resolution”. In: *ACM Trans. Graph.* 39.4 (Aug. 2020).

[8] Tariq D. Aslam. “A partial differential equation approach to multidimensional extrapolation”. In: *Journal of Computational Physics* 193.1 (2004), pp. 349–355.

[9] Christopher Batty. “A cell-centred finite volume method for the Poisson problem on non-graded quadtrees with second order accurate gradients”. In: *Journal of Computational Physics* 331 (2017), pp. 49–72.

[10] Christopher Batty, Florence Bertails, and Robert Bridson. “A Fast Variational Framework for Accurate Solid-Fluid Coupling”. In: *ACM Trans. Graph.* 26.3 (July 2007), 100–es.

[11] John B Bell, Phillip Colella, and Harland M Glaz. “A second-order projection method for the incompressible Navier-Stokes equations”. In: *Journal of Computational Physics* 85.2 (1989), pp. 257–283.

[12] Pedro D Bello-Maldonado, Tzanio V Kolev, Robert N Rieben, and Vladimir Z Tomov. “A matrix-free hyperviscosity formulation for high-order ALE hydrodynamics”. In: *Computers & Fluids* 205 (2020).

[13] Thomas Bellotti and Maxime Theillard. “A coupled level-set and reference map method for interface representation with applications to two-phase flows simulation”. In: *Journal of Computational Physics* 392 (2019), pp. 266–290.

[14] Marsha Berger and Andrew Giuliani. “A state redistribution algorithm for finite volume schemes on cut cell meshes”. In: *Journal of Computational Physics* 428 (2021), p. 109820.

[15] Marsha J Berger and Joseph Oliger. “Adaptive mesh refinement for hyperbolic partial differential equations”. In: *Journal of Computational Physics* 53.3 (1984), pp. 484–512.

[16] D. Bhaga and M. E. Weber. “Bubbles in viscous liquids: shapes, wakes and velocities”. en. In: *Journal of Fluid Mechanics* 105.-1 (Apr. 1981), p. 61.

[17] Adam L Binswanger, Matthew Blomquist, Scott R West, Shilpa Khatri, and Maxime Theillard. “Sharp Collocated Projection Method for Immiscible Two-Phase Flows”. In: *arXiv preprint arXiv:2508.11107* (2025).

[18] Matthew Blomquist, Stéphane Gaudreault, and Maxime Theillard. “Characteristic Bending in Incompressible Flows”. In: *arXiv preprint arXiv:2512.06455* (2025).

[19] Matthew Blomquist, Arman Khalighi, and Abhijit Mukherjee. “Numerical Study of Bubble Instability During Microchannel Flow Boiling”. In: *International Conference on Nanochannels, Microchannels, and Minichannels*. Vol. 56871. American Society of Mechanical Engineers. 2015.

[20] Matthew Blomquist, Scott R West, Adam L Binswanger, and Maxime Theillard. “Stable nodal projection method on octree grids”. In: *Journal of Computational Physics* 499 (2024), p. 112695.

[21] Daniele Boffi, Franco Brezzi, Michel Fortin, et al. *Mixed finite element methods and applications*. Vol. 44. Springer, 2013.

[22] K Boukir, Y Maday, B Métivet, and E Razafindrakoto. "A high-order characteristics/finite element method for the incompressible Navier-Stokes equations". In: *International Journal for Numerical Methods in Fluids* 25.12 (1997), pp. 1421–1454.

[23] Karima Boukir, Yvon Maday, and Brigitte Métivet. "A high order characteristics method for the incompressible Navier—Stokes equations". In: *Computer methods in applied mechanics and engineering* 116.1-4 (1994), pp. 211–218.

[24] John P Boyd. *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

[25] J.U Brackbill, D.B Kothe, and C Zemach. "A continuum method for modeling surface tension". en. In: *Journal of Computational Physics* 100.2 (June 1992), pp. 335–354.

[26] M. Braza, P. Chassaing, and H. Ha Minh. "Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder". In: *Journal of Fluid Mechanics* 165 (1986), pp. 79–130.

[27] David L Brown, Ricardo Cortez, and Michael L Minion. "Accurate projection methods for the incompressible Navier–Stokes equations". In: *Journal of Computational Physics* 168.2 (2001), pp. 464–499.

[28] Donna Calhoun. "A Cartesian Grid Method for Solving the Two-Dimensional Streamfunction-Vorticity Equations in Irregular Regions". In: *Journal of Computational Physics* 176.2 (2002), pp. 231–275.

[29] Richard Camilli, Christopher M. Reddy, Dana R. Yoerger, Benjamin A. S. Van Mooy, Michael V. Jakuba, James C. Kinsey, Cameron P. McIntyre, Sean P. Sylva, and James V. Maloney. "Tracking Hydrocarbon Plume Transport and Biodegradation at Deepwater Horizon". en. In: *Science* 330.6001 (Oct. 2010), pp. 201–204.

[30] Lixin Cheng and Guodong Xia. "Fundamental issues, mechanisms and models of flow boiling heat transfer in microscale channels". In: *International Journal of Heat and Mass Transfer* 108 (2017), pp. 97–127.

[31] Hyuntae Cho and Myungjoo Kang. "Fully implicit and accurate treatment of jump conditions for two-phase incompressible Navier–Stokes equations". In: *Journal of Computational Physics* 445 (2021), p. 110587.

[32] Hyuntae Cho, Yesom Park, and Myungjoo Kang. “Solving incompressible Navier–Stokes equations on irregular domains and quadtrees by monolithic approach”. In: *Journal of Computational Physics* 463 (2022), p. 111304.

[33] David L Chopp. “Some improvements of the fast marching method”. In: *SIAM Journal on Scientific Computing* 23.1 (2001), pp. 230–244.

[34] David Layne Chopp. *Computing minimal surfaces via level set curvature flow*. University of California, Berkeley, 1991.

[35] Alexandre Joel Chorin. “A numerical method for solving incompressible viscous flow problems”. In: *Journal of Computational Physics* 2.1 (1967), pp. 12–26.

[36] Charles Cleret de Langavant, Arthur Guittet, Maxime Theillard, Fernando Temprano-Coleto, and Frédéric Gibou. “Level-set simulations of soluble surfactant driven flows”. In: *Journal of Computational Physics* 348 (2017), pp. 271–297.

[37] Phillip Colella, Daniel T Graves, Benjamin J Keen, and David Modiano. “A Cartesian grid embedded boundary method for hyperbolic conservation laws”. In: *Journal of Computational Physics* 211.1 (2006), pp. 347–366.

[38] Luke P Corcos, Robert I Saye, and James A Sethian. “A hybrid finite difference level set–implicit mesh discontinuous Galerkin method for multi-layer coating flows”. In: *Journal of Computational Physics* 507 (2024).

[39] Jean-François Cossette, Piotr K Smolarkiewicz, and Paul Charbonneau. “The Monge–Ampère trajectory correction for semi-Lagrangian schemes”. In: *Journal of Computational Physics* 274 (2014), pp. 208–229.

[40] Georges-Henri Cottet, Emmanuel Maitre, and Thomas Milcent. “Eulerian formulation and level set models for incompressible fluid-structure interaction”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 42.3 (2008), pp. 471–492.

[41] Olivier Desjardins, Vincent Moureau, and Heinz Pitsch. “An accurate conservative level set/ghost fluid method for simulating turbulent atomization”. In: *Journal of Computational Physics* 227.18 (2008), pp. 8395–8416.

[42] Veselin A Dobrev, Tzanio V Kolev, and Robert N Rieben. “High-order curvilinear finite element methods for Lagrangian hydrodynamics”. In: *SIAM Journal on Scientific Computing* 34.5 (2012).

[43] Jean Donea, SHJP Giuliani, and Jean-Pierre Halleux. “An arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interactions”. In: *Computer methods in applied mechanics and engineering* 33.1-3 (1982), pp. 689–723.

[44] Anshu Dubey, Ann Almgren, John Bell, Martin Berzins, Steve Brandt, Greg Bryan, Phillip Colella, Daniel Graves, Michael Lijewski, Frank Löffler, et al. “A survey of high level frameworks in block-structured adaptive mesh refinement packages”. In: *Journal of Parallel and Distributed Computing* 74.12 (2014), pp. 3217–3227.

[45] H Dütsch, F Durst, S Becker, and H Lienhart. “Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan–Carpenter numbers”. In: *Journal of Fluid Mechanics* 360 (1998), pp. 249–271.

[46] Raphael Egan, Arthur Guittet, Fernando Temprano-Coleto, Tobin Isaac, François J. Peaudecerf, Julien R. Landel, Paolo Luzzatto-Fegiz, Carsten Burstedde, and Frédéric Gibou. “Direct numerical simulation of incompressible flows on parallel Octree grids”. In: *Journal of Computational Physics* 428 (2021), p. 110084.

[47] Michael S. Engelman and Mohammad-Ali Jamnia. “Transient flow past a circular cylinder: A benchmark solution”. In: *International Journal for Numerical Methods in Fluids* 11.7 (1990), pp. 985–1000.

[48] Doug Enright, Duc Nguyen, Frédéric Gibou, and Ron Fedkiw. “Using the Particle Level Set Method and a Second Order Accurate Pressure Boundary Condition for Free Surface Flows”. In: Fluids Engineering Division Summer Meeting Volume 2: Symposia, Parts A, B, and C (July 2003), pp. 337–342.

[49] Douglas Enright, Frank Losasso, and Ronald Fedkiw. “A fast and accurate semi-Lagrangian particle level set method”. In: *Computers & structures* 83.6-7 (2005), pp. 479–490.

[50] Ercan Erturk, Thomas C Corke, and Cihan Gökçöl. “Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers”. In: *International journal for Numerical Methods in fluids* 48.7 (2005), pp. 747–774.

[51] Maurizio Falcone and Roberto Ferretti. “Convergence analysis for a class of high-order semi-Lagrangian advection schemes”. In: *SIAM Journal on Numerical Analysis* 35.3 (1998), pp. 909–940.

[52] Dixia Fan, Liu Yang, Zhicheng Wang, Michael S Triantafyllou, and George Em Karniadakis. “Reinforcement learning for bluff body active flow control in experiments and simulations”. In: *Proceedings of the National Academy of Sciences* 117.42 (2020), pp. 26091–26098.

[53] Steven J Fletcher. *Semi-Lagrangian advection methods and their applications in geoscience*. Elsevier, 2019.

[54] Marianne M. Francois, Sharen J. Cummins, Edward D. Dendy, Douglas B. Kothe, James M. Sicilian, and Matthew W. Williams. "A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework". en. In: *Journal of Computational Physics* 213.1 (Mar. 2006), pp. 141–173.

[55] Cayce Fylling, Joshua Tamayo, Arvind Gopinath, and Maxime Theillard. "Multi-population dissolution in confined active fluids". In: *Soft Matter* 20.7 (2024), pp. 1392–1409.

[56] Cédric Galusinski and Paul Vigneaux. "On stability condition for bifluid flows with surface tension: Application to microfluidics". en. In: *Journal of Computational Physics* 227.12 (June 2008), pp. 6140–6164.

[57] UKNG Ghia, Kirti N Ghia, and CT Shin. "High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method". In: *Journal of Computational Physics* 48.3 (1982), pp. 387–411.

[58] Frédéric Gibou, Ronald Fedkiw, Russel Caflisch, and Stanley Osher. "A level set approach for the numerical simulation of dendritic growth". In: *Journal of Scientific Computing* 19 (2003), pp. 183–199.

[59] Frédéric Gibou, Ronald P Fedkiw, Li-Tien Cheng, and Myungjoo Kang. "A second-order-accurate symmetric discretization of the Poisson equation on irregular domains". In: *Journal of Computational Physics* 176.1 (2002), pp. 205–227.

[60] Frédéric Gibou and Chohong Min. "Efficient symmetric positive definite second-order accurate monolithic solver for fluid/solid interactions". In: *Journal of Computational Physics* 231.8 (2012), pp. 3246–3263.

[61] Francis X Giraldo. "Lagrange–Galerkin methods on spherical geodesic grids: the shallow water equations". In: *Journal of Computational Physics* 160.1 (2000), pp. 336–368.

[62] Francis X Giraldo. "The Lagrange–Galerkin method for the two-dimensional shallow water equations on adaptive grids". In: *International Journal for Numerical Methods in Fluids* 33.6 (2000), pp. 789–832.

[63] Francis X Giraldo. "The Lagrange–Galerkin spectral element method on unstructured quadrilateral grids". In: *Journal of Computational Physics* 147.1 (1998), pp. 114–146.

[64] Andrew Giuliani, Ann S Almgren, John B Bell, Marsha J Berger, MT Henry de Frahan, and Deepak Rangarajan. “A weighted state redistribution algorithm for embedded boundary grids”. In: *Journal of Computational Physics* 464 (2022).

[65] Katuhiko Goda. “A multistep technique with implicit difference schemes for calculating two- or three-dimensional cavity flows”. In: *Journal of Computational Physics* 30.1 (1979-1).

[66] Pablo Gómez, Claudio Zanzi, Joaquín López, and Julio Hernández. “Simulation of high density ratio interfacial flows on cell vertex/edge-based staggered octree grids with second-order discretization at irregular nodes”. In: *Journal of Computational Physics* 376 (2019), pp. 478–507.

[67] Arthur Guittet, Maxime Theillard, and Frédéric Gibou. “A stable projection method for the incompressible Navier–Stokes equations on arbitrary geometries and adaptive Quad/Octrees”. In: *Journal of Computational Physics* 292 (2015), pp. 215–238.

[68] Vincenzo Gulizzi and Robert Saye. “Modeling wave propagation in elastic solids via high-order accurate implicit-mesh discontinuous Galerkin methods”. In: *Computer Methods in Applied Mechanics and Engineering* 395 (2022), p. 114971.

[69] Francis H Harlow and J Eddie Welch. “Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface”. In: *The physics of fluids* 8.12 (1965), pp. 2182–2189.

[70] Ami Harten, Stanley Osher, Björn Engquist, and Sukumar R Chakravarthy. “Some results on uniformly high-order accurate essentially nonoscillatory schemes”. In: *Applied Numerical Mathematics* 2.3-5 (1986), pp. 347–377.

[71] Dalton JE Harvie, MR Davidson, and Murray Rudman. “An analysis of parasitic current generation in volume of fluid simulations”. In: *Applied mathematical modelling* 30.10 (2006), pp. 1056–1066.

[72] Jan S Hesthaven. *Numerical methods for conservation laws: From analysis to algorithms*. SIAM, 2017.

[73] Cyril W Hirt and Billy D Nichols. “Volume of fluid (VOF) method for the dynamics of free boundaries”. In: *Journal of Computational Physics* 39.1 (1981), pp. 201–225.

[74] Cyrill W Hirt, Anthony A Amsden, and JL Cook. “An arbitrary Lagrangian-Eulerian computing method for all flow speeds”. In: *Journal of Computational Physics* 14.3 (1974), pp. 227–253.

[75] William Holman-Bissegger. *A volume-preserving Characteristic Mapping Method for the 2D incompressible Euler equations*. McGill University (Canada), 2024.

[76] Mariano Hortal. "The development and testing of a new two-time-level semi-Lagrangian scheme (SETTLS) in the ECMWF forecast model". In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 128.583 (2002), pp. 1671–1687.

[77] Ming-Chen Hsu, David Kamensky, Yuri Bazilevs, Michael S Sacks, and Thomas JR Hughes. "Fluid–structure interaction analysis of bioprosthetic heart valves: significance of arterial wall deformation". In: *Computational mechanics* 54 (2014), pp. 1055–1071.

[78] Thomas JR Hughes, Wing Kam Liu, and Thomas K Zimmermann. "Lagrangian-Eulerian finite element formulation for incompressible viscous flows". In: *Computer methods in applied mechanics and engineering* 29.3 (1981), pp. 329–349.

[79] David AB Hyde and Ronald Fedkiw. "A unified approach to monolithic solid-fluid coupling of sub-grid and more resolved solids". In: *Journal of Computational Physics* 390 (2019), pp. 490–526.

[80] G Iliadis and Petros Anagnostopoulos. "Viscous oscillatory flow around a circular cylinder at low Keulegan–Carpenter numbers and frequency parameters". In: *International journal for numerical methods in fluids* 26.4 (1998), pp. 403–442.

[81] YANG Jianming. "Sharp interface direct forcing immersed boundary methods: A summary of some algorithms and applications". In: *Journal of Hydrodynamics, Ser. B* 28.5 (2016), pp. 713–730.

[82] T. A. Johnson and V. C. Patel. "Flow past a sphere up to a Reynolds number of 300". In: *Journal of Fluid Mechanics* 378 (1999), pp. 19–70.

[83] Ken Kamrin, Chris H. Rycroft, and Jean-Christophe Nave. "Reference map technique for finite-strain elasticity and fluid–solid interaction". en. In: *Journal of the Mechanics and Physics of Solids* 60.11 (Nov. 2012), pp. 1952–1969.

[84] Kenneth Norman Kamrin. "Stochastic and deterministic models for dense granular flow". PhD thesis. Massachusetts Institute of technology, 2008.

[85] Satish G. Kandlikar. "History, Advances, and Challenges in Liquid Flow and Flow Boiling Heat Transfer in Microchannels: A Critical Review". In: *Journal of Heat Transfer* 134.3 (2012).

[86] George Karniadakis and Spencer J Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2005.

[87] George Em Karniadakis, Moshe Israeli, and Steven A Orszag. "High-order splitting methods for the incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 97.2 (1991), pp. 414–443.

[88] John D. Kessler, David L. Valentine, Molly C. Redmond, Mengran Du, Eric W. Chan, Stephanie D. Mendes, Erik W. Quiroz, Christie J. Villanueva, Stephani S. Shusta, Lindsay M. Werra, Shari A. Yvon-Lewis, and Thomas C. Weber. "A Persistent Oxygen Anomaly Reveals the Fate of Spilled Methane in the Deep Gulf of Mexico". en. In: *Science* 331.6015 (Jan. 2011), pp. 312–315.

[89] Arman Khalighi, Matthew Blomquist, and Abhijit Mukherjee. "Numerical simulation of bubble growth during nanofluid flow boiling in a microchannel". In: *International Conference on Nanochannels, Microchannels, and Minichannels*. Vol. 46278. American Society of Mechanical Engineers. 2014.

[90] John Kim and Parviz Moin. "Application of a fractional-step method to incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 59.2 (1985), pp. 308–323.

[91] Kamau Kingora and Hamid Sadat-Hosseini. "A novel interpolation-free sharp-interface immersed boundary method". In: *Journal of Computational Physics* 453 (2022).

[92] Haruhiko Kohno and Jean-Christophe Nave. "A new method for the level set equation using a hierarchical-gradient truncation and remapping technique". In: *Computer Physics Communications* 184.6 (2013), pp. 1547–1554.

[93] Anna Kucherova, Selma Strango, Shahar Sukenik, and Maxime Theillard. "Computational modeling of protein conformational changes - Application to the opening SARS-CoV-2 spike". In: *Journal of Computational Physics* 444 (2021), p. 110591.

[94] S.K. Kwok. "An improved curvilinear finite difference (CFD) method for arbitrary mesh systems". In: *Computers & Structures* 18.4 (1984), pp. 719–731.

[95] Horace Lamb. *Hydrodynamics*. University Press, 1924.

[96] Peter H Lauritzen, Ramachandran D Nair, and Paul A Ullrich. "A conservative semi-Lagrangian multi-tracer transport scheme (CSLAM) on the cubed-sphere grid". In: *Journal of Computational Physics* 229.5 (2010), pp. 1401–1424.

[97] BP Le Clair, AE Hamielec, and HR Pruppacher. “A numerical study of the drag on a sphere at low and intermediate Reynolds numbers”. In: *Journal of Atmospheric Sciences* 27.2 (1970), pp. 308–315.

[98] Minjae Lee, David Hyde, Kevin Li, and Ronald Fedkiw. “A robust volume conserving method for character-water interaction”. In: *Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2019, pp. 1–12.

[99] Randall J LeVeque. *Finite volume methods for hyperbolic problems*. Vol. 31. Cambridge university press, 2002.

[100] Randall J Leveque. “High-resolution conservative algorithms for advection in incompressible flow”. In: *SIAM Journal on Numerical Analysis* 33.2 (1996), pp. 627–665.

[101] Xu-Dong Liu, Stanley Osher, and Tony Chan. “Weighted essentially non-oscillatory schemes”. In: *Journal of Computational Physics* 115.1 (1994), pp. 200–212.

[102] Xiaohan Long and Chuanjun Chen. “General Formulation of Second-Order Semi-Lagrangian Methods for Convection-Diffusion Problems”. en. In: *Abstract and Applied Analysis* 2013 (2013), pp. 1–10.

[103] Frank Losasso, Ronald Fedkiw, and Stanley Osher. “Spatially adaptive techniques for level set methods and incompressible flow”. In: *Computers & Fluids* 35.10 (2006), pp. 995–1010.

[104] Frank Losasso, Frédéric Gibou, and Ron Fedkiw. “Simulating water and smoke with an octree data structure”. In: *Acm siggraph 2004 papers*. 2004, pp. 457–462.

[105] Emmanuel Maitre, Thomas Milcent, Georges-Henri Cottet, Annie Raoult, and Yves Usson. “Applications of level set methods in computational biophysics”. In: *Mathematical and Computer Modelling* 49.11-12 (2009), pp. 2161–2169.

[106] Marc Mancini, Maxime Theillard, and Changho Kim. “Projection method for the fluctuating hydrodynamics equations”. In: *Journal of Computational Physics* (2022), p. 111288.

[107] Tracy L. Mandel, De Zhen Zhou, Lindsay Waldrop, Maxime Theillard, Dustin Kleckner, and Shilpa Khatri. “Retention of rising droplets in density stratification”. en. In: *Physical Review Fluids* 5.12 (Dec. 2020), p. 124803.

[108] S Marella, SLHH Krishnan, H Liu, and HS Udaykumar. “Sharp interface Cartesian grid method I: an easily implemented technique for 3D moving boundary computations”. In: *Journal of Computational Physics* 210.1 (2005), pp. 1–31.

[109] Marcia K. McNutt, Rich Camilli, Timothy J. Crone, George D. Guthrie, Paul A. Hsieh, Thomas B. Ryerson, Omer Savas, and Frank Shaffer. “Review of flow rate estimates of the *Deepwater Horizon* oil spill”. en. In: *Proceedings of the National Academy of Sciences* 109.50 (Dec. 2012), pp. 20260–20267.

[110] Olivier Mercier, Xi-Yuan Yin, and Jean-Christophe Nave. “The characteristic mapping method for the linear advection of arbitrary sets”. In: *SIAM Journal on Scientific Computing* 42.3 (2020), A1663–A1685.

[111] Chohong Min and Frédéric Gibou. “A second order accurate level set method on non-graded adaptive Cartesian grids”. In: *Journal of Computational Physics* 225.1 (2007), pp. 300–321.

[112] Chohong Min and Frédéric Gibou. “A second order accurate projection method for the incompressible Navier–Stokes equations on non-graded adaptive grids”. In: *Journal of Computational Physics* 219.2 (2006), pp. 912–929.

[113] Chohong Min and Frédéric Gibou. “Geometric integration over irregular domains with application to level-set methods”. In: *J. Comput. Phys.* 226 (2007), pp. 1432–1443.

[114] Chohong Min, Frédéric Gibou, and Hector D Ceniceros. “A supra-convergent finite difference scheme for the variable coefficient Poisson equation on non-graded grids”. In: *Journal of Computational Physics* 218.1 (2006), pp. 123–140.

[115] Michael L Minion. “A projection method for locally refined grids”. In: *Journal of Computational Physics* 127.1 (1996), pp. 158–178.

[116] Rajat Mittal, Haibo Dong, Meliha Bozkurttas, FM Najjar, Abel Vargas, and Alfred Von Loebbecke. “A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries”. In: *Journal of Computational Physics* 227.10 (2008), pp. 4825–4852.

[117] KW Morton. “Generalised Galerkin methods for hyperbolic problems”. In: *Computer Methods in Applied Mechanics and Engineering* 52.1-3 (1985), pp. 847–871.

[118] Paul Mullowney, Ruipeng Li, Stephen Thomas, Shreyas Ananthan, Ashesh Sharma, Jon S Rood, Alan B Williams, and Michael A Sprague. "Preparing an incompressible-flow fluid dynamics code for exascale-class wind energy simulations". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 2021, pp. 1–16.

[119] M Nagheeby and M Kolahdoozan. "Numerical modeling of two-phase fluid flow and oil slick transport in estuarine water". In: *International Journal of Environmental Science & Technology* 7 (2010), pp. 771–784.

[120] Jean-Christophe Nave, Rodolfo Ruben Rosales, and Benjamin Seibold. "A gradient-augmented level set method with an optimally local, coherent advection scheme". In: *Journal of Computational Physics* 229.10 (2010), pp. 3802–3827.

[121] Yen Ting Ng, Chohong Min, and Frédéric Gibou. "An efficient fluid–solid coupling algorithm for single-phase flows". In: *Journal of Computational Physics* 228.23 (2009), pp. 8807–8829.

[122] Elin Olsson and Gunilla Kreiss. "A conservative level set method for two phase flow". In: *Journal of Computational Physics* 210.1 (2005), pp. 225–246.

[123] Elin Olsson, Gunilla Kreiss, and Sara Zahedi. "A conservative level set method for two phase flow II". In: *Journal of Computational Physics* 225.1 (2007), pp. 785–807.

[124] Steven A Orszag, Moshe Israeli, and Michel O Deville. "Boundary conditions for incompressible flows". In: *Journal of Scientific Computing* 1 (1986), pp. 75–111.

[125] Stanley Osher, Ronald Fedkiw, and Krzysztof Piechor. "Level set methods and dynamic implicit surfaces". In: *Appl. Mech. Rev.* 57.3 (2004), B15–B15.

[126] Stanley Osher and James A Sethian. "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations". en. In: *Journal of Computational Physics* 79.1 (Nov. 1988), pp. 12–49.

[127] Stanley. Osher and Ronald P. Fedkiw. *Level set methods and dynamic implicit surfaces*. eng. Applied mathematical sciences ; v. 153. New York: Springer, 2003.

[128] Mark Owkes and Olivier Desjardins. "A discontinuous Galerkin conservative level set scheme for interface capturing in multiphase flows". In: *Journal of Computational Physics* 249 (2013), pp. 275–302.

[129] Will Pazner and Per-Olof Persson. "Approximate tensor-product preconditioners for very high order discontinuous Galerkin methods". In: *Journal of Computational Physics* 354 (2018), pp. 344–369.

[130] J Blair Perot. “An analysis of the fractional step method”. In: *Journal of Computational Physics* 108.1 (1993), pp. 51–58.

[131] Charles S Peskin. “Numerical analysis of blood flow in the heart”. In: *Journal of Computational Physics* 25.3 (1977), pp. 220–252.

[132] Charles S Peskin. “The immersed boundary method”. In: *Acta numerica* 11 (2002), pp. 479–517.

[133] Stéphane Popinet. “Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries”. In: *Journal of Computational Physics* 190.2 (2003), pp. 572–600.

[134] Stéphane Popinet and S Zaleski. “A front-tracking algorithm for accurate representation of surface tension”. In: *International Journal for Numerical Methods in Fluids* 30.6 (1999), pp. 775–793.

[135] Jc Prairie, K Ziervogel, C Arnosti, R Camassa, C Falcon, S Khatri, Rm McLaughlin, Bl White, and S Yu. “Delayed settling of marine snow at sharp density transitions driven by fluid entrainment and diffusion-limited retention”. en. In: *Marine Ecology Progress Series* 487 (July 2013), pp. 185–200.

[136] A Priestley. “A quasi-conservative version of the semi-Lagrangian advection scheme”. In: *Monthly Weather Review* 121.2 (1993), pp. 621–629.

[137] André Robert. “A stable numerical integration scheme for the primitive meteorological equations”. In: *Atmosphere-Ocean* 19.1 (1981), pp. 35–46.

[138] Robert Saye. *Algoim: Algorithms for Implicitly Defined Geometry and Level Set Methods*. `https://algoim.github.io/`. Accessed: 2026-01-15. 2022.

[139] Giorgio Rosatti, D Cesari, and Luca Bonaventura. “Semi-implicit, semi-Lagrangian modelling for environmental problems on staggered Cartesian grids with cut cells”. In: *Journal of Computational Physics* 204.1 (2005), pp. 353–377.

[140] Chris H Rycroft, Chen-Hung Wu, Yue Yu, and Ken Kamrin. “Reference map technique for incompressible fluid–structure interaction”. In: *Journal of Fluid Mechanics* 898 (2020).

[141] Pavel Ryzhakov, Jordi Cotela, Riccardo Rossi, and Eugenio Oñate. “A two-step monolithic method for the efficient simulation of incompressible flows”. In: *International journal for numerical methods in fluids* 74.12 (2014), pp. 919–934.

[142] Hanan Samet. “An overview of quadtrees, octrees, and related hierarchical data structures”. In: *Theoretical Foundations of Computer Graphics and CAD* (1988), pp. 51–68.

[143] John Stanley Sawyer. “A semi-Lagrangian method of solving the vorticity advection equation”. In: *Tellus* 15.4 (1963), pp. 336–342.

[144] Robert Saye. “High-order methods for computing distances to implicitly defined surfaces”. In: *Communications in Applied Mathematics and Computational Science* 9.1 (2014), pp. 107–141.

[145] Robert Saye. “Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part I”. In: *Journal of Computational Physics* 344 (2017), pp. 647–682.

[146] Robert Saye. “Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid–structure interaction, and free surface flow: Part II”. In: *Journal of Computational Physics* 344 (2017), pp. 683–723.

[147] Robert Saye. “Interfacial gauge methods for incompressible fluid dynamics”. In: *Science advances* 2.6 (2016), e1501869.

[148] Robert I Saye, James A Sethian, Brandon Petrouskie, Aaron Zatorsky, Xinyu Lu, and Reza Rock. “Insights from high-fidelity modeling of industrial rotary bell atomization”. In: *Proceedings of the National Academy of Sciences* 120.4 (2023), e2216709120.

[149] Florian Schornbaum and Ulrich Rüde. “Extreme-scale block-structured adaptive mesh refinement”. In: *SIAM Journal on Scientific Computing* 40.3 (2018), pp. C358–C387.

[150] Jung Hee Seo and Rajat Mittal. “A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations”. In: *Journal of Computational Physics* 230.19 (2011), pp. 7347–7363.

[151] J. A. Sethian. “Fast Marching Methods”. In: *SIAM review* 41.2 (1999), pp. 199–235.

[152] James A Sethian et al. *Level set methods and fast marching methods*. Vol. 98. Cambridge Cambridge UP, 1999.

[153] Chi-Wang Shu and Stanley Osher. “Efficient implementation of essentially non-oscillatory shock-capturing schemes”. In: *Journal of Computational Physics* 77.2 (1988), pp. 439–471.

[154] S. A. Socolofsky and E. E. Adams. “Multi-phase plumes in uniform and stratified crossflow”. en. In: *Journal of Hydraulic Research* 40.6 (Nov. 2002), pp. 661–672.

[155] Andrew Staniforth and Jean Côté. “Semi-Lagrangian integration schemes for atmospheric models—A review”. In: *Monthly weather review* 119.9 (1991), pp. 2206–2223.

[156] M. Sussman, K.M. Smith, M.Y. Hussaini, M. Ohta, and R. Zhi-Wei. “A sharp interface method for incompressible two-phase flows”. en. In: *Journal of Computational Physics* 221.2 (Feb. 2007), pp. 469–505.

[157] Mark Sussman. “A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles”. In: *Journal of Computational Physics* 187.1 (2003), pp. 110–136.

[158] Mark Sussman and Emad Fatemi. “An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow”. In: *SIAM Journal on scientific computing* 20.4 (1999), pp. 1165–1191.

[159] Mark Sussman and Elbridge Gerry Puckett. “A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows”. In: *Journal of Computational Physics* 162.2 (2000), pp. 301–337.

[160] Mark Sussman, Peter Smereka, and Stanley Osher. “A level set approach for computing solutions to incompressible two-phase flow”. In: *Journal of Computational physics* 114.1 (1994), pp. 146–159.

[161] Tetsuya Takahashi and Christopher Batty. “Monolith: A monolithic pressure-viscosity-contact solver for strong two-way rigid-rigid rigid-fluid coupling”. In: *ACM Transactions on Graphics (TOG)* 39.6 (2020), pp. 1–16.

[162] Roger Temam. “Sur l’approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (I)”. In: *Archive for Rational Mechanics and Analysis* 32 (1969), pp. 135–153.

[163] Roger Temam. “Sur l’approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (II)”. In: *Archive for Rational Mechanics and Analysis* 32 (1969), pp. 135–153.

[164] Maxime Theillard. “A volume-preserving reference map method for the level set representation”. In: *Journal of Computational Physics* 442 (2021), p. 110478.

[165] Maxime Theillard, Roberto Alonso-Matilla, and David Saintillan. “Geometric control of active collective motion”. In: *Soft Matter* 13.2 (2017), pp. 363–375.

[166] Maxime Theillard, Landry Fokoua Djodom, Jean-Léopold Vié, and Frédéric Gibou. “A second-order sharp numerical method for solving the linear elasticity equations on irregular domains and adaptive grids – Application to shape optimization”. In: *Journal of Computational Physics* 233 (2013), pp. 430–448.

[167] Maxime Theillard, Frédéric Gibou, and David Saintillan. “Sharp numerical simulation of incompressible two-phase flows”. In: *Journal of Computational Physics* 391 (2019), pp. 91–118.

[168] Maxime Theillard, Chris H. Rycroft, and Frédéric Gibou. “A Multigrid Method on Non-Graded Adaptive Octree and Quadtree Cartesian Grids”. In: *Journal of Scientific Computing* 55 (2013), pp. 1–15.

[169] Maxime Theillard and David Saintillan. “Computational mean-field modeling of confined active fluids”. In: *Journal of Computational Physics* 397 (2019), p. 108841.

[170] John R Thome. “Boiling in microchannels: a review of experiment and theory”. In: *International Journal of Heat and Fluid Flow* 25.2 (2004), pp. 128–139.

[171] R. P. Turco, O. B. Toon, T. P. Ackerman, J. B. Pollack, and C. Sagan. “Climate and Smoke: an Appraisal of Nuclear Winter”. en. In: *Science* 247.4939 (Jan. 1990), pp. 166–176.

[172] HS Udaykumar, S Marella, and S Krishnan. “Sharp-interface simulation of dendritic growth with convection: benchmarks”. In: *International journal of heat and mass transfer* 46.14 (2003), pp. 2615–2627.

[173] HS Udaykumar, R Mittal, and P Rampunggoon. “Interface tracking finite volume method for complex solid–fluid interactions on fixed meshes”. In: *Communications in numerical methods in engineering* 18.2 (2002), pp. 89–97.

[174] HS Udaykumar, R Mittal, P Rampunggoon, and A Khanna. “A sharp interface Cartesian grid method for simulating flows with complex moving boundaries”. In: *Journal of Computational Physics* 174.1 (2001), pp. 345–380.

[175] Salih Ozen Unverdi and Grétar Tryggvason. “A front-tracking method for viscous, incompressible, multi-fluid flows”. In: *Journal of Computational Physics* 100.1 (1992), pp. 25–37.

[176] Henk A Van der Vorst. “Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems”. In: *SIAM Journal on scientific and Statistical Computing* 13.2 (1992), pp. 631–644.

[177] JJIM Van Kan. “A second-order accurate pressure-correction scheme for viscous incompressible flow”. In: *SIAM journal on scientific and statistical computing* 7.3 (1986), pp. 870–891.

[178] Arturo Vargas, Vladimir Z Tomov, M Aaron Skinner, Veselin Dobrev, Jan Nikl, Tzanio Kolev, and Robert N Rieben. “Multi-material ALE remap with interface sharpening using high-order matrix-free finite element methods”. In: *Journal of Computational Physics* (2025), p. 114367.

[179] Roberto Verzicco. “Immersed boundary methods: Historical perspective and future outlook”. In: *Annual Review of Fluid Mechanics* 55.1 (2023), pp. 129–155.

[180] Scott Robert West. “Nodal Projection Methods for Incompressible Flows and Vortex Dynamics Applications”. PhD thesis. University of California, Merced, 2025.

[181] Feng Xiao and Takashi Yabe. “Completely conservative and oscillationless semi-Lagrangian schemes for advection transportation”. In: *Journal of Computational Physics* 170.2 (2001), pp. 498–522.

[182] Shuolin Xiao and Di Yang. “Effect of oil plumes on upper-ocean radiative transfer—A numerical study”. In: *Ocean Modelling* 145 (2020).

[183] Dongbin Xiu and George Em Karniadakis. “A Semi-Lagrangian High-Order Method for Navier–Stokes Equations”. In: *Journal of Computational Physics* 172.2 (2001), pp. 658–684.

[184] Takashi Yabe, Ryotaro Tanaka, Takashi Nakamura, and Feng Xiao. “An exactly conservative semi-Lagrangian scheme (CIP–CSL) in one dimension”. In: *Monthly Weather Review* 129.2 (2001), pp. 332–344.

[185] Di Yang, Bicheng Chen, Marcelo Chamecki, and Charles Meneveau. “Oil plumes and dispersion in L angmuir, upper-ocean turbulence: Large-eddy simulations and K-profile parameterization”. In: *Journal of Geophysical Research: Oceans* 120.7 (2015), pp. 4729–4759.

[186] Tao Ye, Rajat Mittal, HS Udaykumar, and Wei Shyy. “An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries”. In: *Journal of Computational Physics* 156.2 (1999), pp. 209–240.

[187] Xi-Yuan Yin, Philipp Krah, Jean-Christophe Nave, and Kai Schneider. “A Characteristic Mapping Method with Source Terms: Applications to Ideal Magnetohydrodynamics”. In: *arXiv preprint arXiv:2411.13772* (2024).

[188] Xi-Yuan Yin, Olivier Mercier, Badal Yadav, Kai Schneider, and Jean-Christophe Nave. “A characteristic mapping method for the two-dimensional incompressible Euler equations”. In: *Journal of Computational Physics* 424 (2021), p. 109781.

[189] Steven T Zalesak. “Fully multidimensional flux-corrected transport algorithms for fluids”. In: *Journal of Computational Physics* 31.3 (1979), pp. 335–362.

[190] Mohamed Zerroukat, Nigel Wood, and Andrew Staniforth. “SLICE: A Semi-Lagrangian Inherently Conserving and Efficient scheme for transport problems”. In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 128.586 (2002), pp. 2801–2820.

[191] Weiqun Zhang, Ann Almgren, Vince Beckner, John Bell, Johannes Blaschke, Cy Chan, Marcus Day, Brian Friesen, Kevin Gott, Daniel Graves, et al. “AMReX: a framework for block-structured adaptive mesh refinement”. In: *Journal of Open Source Software* 4.37 (2019), pp. 1370–1370.

[192] Weiqun Zhang, Andrew Myers, Kevin Gott, Ann Almgren, and John Bell. “AMReX: Block-structured adaptive mesh refinement for multiphysics applications”. In: *The International Journal of High Performance Computing Applications* 35.6 (2021), pp. 508–526.