Parallel Solution-Adaptive Method for Predicting Solid Propellant Rocket Motor Core Flows

Doctor of Philosophy

Jai Singh Sachdev

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Aerospace Science and Engineering
University of Toronto

Copyright © 2007 by Jai Singh Sachdev

Abstract

PARALLEL SOLUTION-ADAPTIVE METHOD FOR PREDICTING SOLID PROPELLANT ROCKET MOTOR CORE FLOWS

Jai Singh Sachdev

Doctor of Philosophy

Graduate Department of Aerospace Science and Engineering

University of Toronto

2007

A computational framework has been developed for predicting two-dimensional axisymmetric turbulent multi-phase (gas-particle) solid propellant rocket motor core flows. The cornerstone of this numerical scheme is a parallel block-based adaptive mesh refinement (AMR) algorithm. AMR methods are effective in resolving the multiple solution scales typical of such complex fluid flow while minimizing computational expense. The use of body-fitted mesh blocks makes the application of the block-based AMR more amenable to flows with thin boundary layers and permits anisotropic refinement as dictated by initial mesh stretching. The block-based data structure lends itself naturally to domain decomposition and thereby enables efficient and scalable implementations of the algorithm on distributed-memory multi-processor architectures. A mesh adjustment scheme has been devised in which the body-fitted multi-block mesh is locally adjusted to arbitrarily embedded boundaries that are not necessarily aligned with the mesh. Not only does this scheme allow for rapid and robust mesh generation involving complex embedded boundaries, it also enables the solution of unsteady flow problems involving bodies and interfaces moving relative to the flow domain. This allows for the modelling and treatment of the pressure dependent burning of the solid propellant. The level set method is used to treat the evolution of the burning propellant surface. A Godunov-type finite-volume scheme has been developed to accurately predict the interaction between the gas and solid particles injected into the rocket chamber at the combustion interface. Standard fourth-order Runge-Kutta and second-order predictor-corrector time-marching schemes are used for time-accurate temporal discretization. An explicit optimally-smoothing multi-stage time-stepping scheme with multigrid convergence acceleration is used for obtaining steady state solutions. The mathematical characteristics of an Eulerian formulation for the particle-phase required the development of a new solution method for the particle-phase that allows for a more physically realistic solution than was previously attainable. A novel multi-velocity formulation is proposed that can account for crossing particle trajectories by splitting the particle-phase into distinct velocity families which are transported separately. The overall proposed methodology provides a state-of-the-art framework for accurately and efficiently predicting rocket motor core flows.

Acknowledgements

The experience of pursuing and completing this Ph.D. degree has been enriched through the interaction with many people. I have been very fortunate to have had Profs. Clinton Groth and James Gottlieb as my Ph.D. supervisors. I would like to thank Prof. Groth for sharing his passion for the field of our study. I am indebted to Prof. Gottlieb for providing an opportunity to pursue a graduate education when others may not have. This thesis would not have been possible without their excellent supervision, teaching, encouragement, and friendship. My thesis research benefited greatly from the questions, comments, and suggestions raised by Profs. David Zingg and Ömer Gulder at my review meetings.

To my colleagues at UTIAS, I would like to thank all of you for the many technical and non-technical discussions over the years. However, I feel it is necessary to acknowledge several colleagues who have been greatly influential during my tenure at UTIAS. In the early days, Kevin Linfield, Peter Ess, Jurgen Schumacher, and Giovanni Fusina were a great source of assistance and a potent group of soccer players. From the CFD group, Marian Nemec, Luis Manzano, and Jason Lassaline were always willing to provide technical assistance, regardless of the level of my questions. In the later years, the academic and non-academic interactions I had with Peterson Wong, Scott Northrup, and John Gatsis made my time at UTIAS intellectually stimulating and highly enjoyable. In addition, I am greatly indebted to Scott Northrup for his work as the system administrator.

To my friends and family, I would like to thank all of you for your support and inspiration. Most of all, I would like to express my deepest appreciation to my wife, Jo Gardiner Sachdev. She never once faltered in her belief in me and was quick to provide the love and support I needed in order to complete my research.

Financial assistance provided by the Ontario Graduate Scholarship program is gratefully acknowledged.

Jai Singh Sachdev
University of Toronto Institute for Aerospace Studies
March 29, 2007

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Motivation and Background

1.1.1 Internal Ballistics of Solid Propellant Rocket Motors

Solid propellant rocket motors (SRMs) are chemical rockets in which the fuel and oxidizer are combined into a hardened grain. They are relatively simple to design (few structural components) and to use (easily ignited). SRMs are commonly employed as booster rockets for launch vehicles and in tactical rocket systems. A schematic of a solid propellant rocket motor is shown in Figure 1.1. The essential components of a SRM are the motor casing, the propellant grain, the igniter, and the nozzle. Although conceptually straight-forward, the internal flow dynamics of a solid propellant rocket motor are very complex and not yet completely understood. A brief description of SRM operation is provided here. Refer to the textbooks by Oates [134] and Humble *et al.* [84] and the lecture notes by Kuentzmann [99] for more details.

The combustion of the propellant occurs in a thin, high temperature layer between the propellant grain and the main flow cavity, known as the combustion interface. This topologically complex surface evolves as the propellant burns and the hot gaseous combustion products are injected into the volume in the centre of the rocket known as the combustion chamber. The burning rate of the propellant is dependent on the local gas pressure, convective gas flow across the grain, and rate of heat transfer through the propellant. Injection of the combustion products into the chamber core occurs at low subsonic Mach numbers, $M \leq 0.1$. As the pressure in the combustion chamber increases, the gases are propelled through the nozzle. When the chamber pressure is high enough (relative to the ambient pressure), the combustion products are accelerated through a converging-diverging nozzle from subsonic flow speeds in the combustion chamber, to sonic conditions at the throat of the nozzle, $M = 1$, and on to supersonic speeds at the nozzle exit, $M \approx 2-3$, resulting in both high mass flow rates and thrust. Typical conditions in the combustion chamber are pressures around 3-5 MPa and temperatures up to 3500 K, and the motor casing must be able to withstand these high pressures and temperatures.

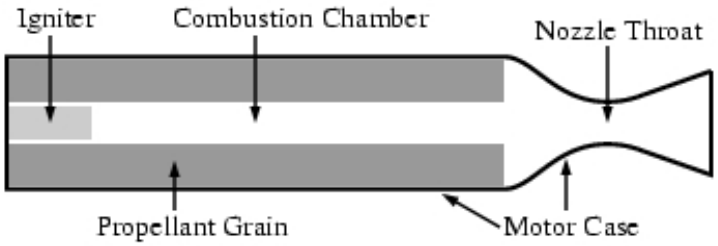


Figure 1.1: *Schematic of a solid propellant rocket motor.*

Control of the burning rate is critical during SRM operation. During normal steady-state combustion the burning rate is dependent on the pressure and the initial grain temperature. If the combustion products are moving rapidly over the propellant grain, then the turbulent flow enhances the convective heat transfer to the combustion interface causing an increase in the burning rate. This phenomenon is known as erosive burning. Unsteady burning rates occur when the fluid conditions above the combustion interface are undergoing rapid changes, such as during ignition and at the end of the motor operation. The burning rate generally increases with the chamber pressure and the chamber pressure increases with the burning rate. The coupling of pressure fluctuations to the combustion of the propellant and the subsequent growth of unstable modes are known as combustion instabilities. There are two main types of combustion instabilities: the volume or L^* instability and the acoustic instability. The L^* instability is of low frequency and typically occurs during ignition [154]. This instability is not dangerous but it can lead to quenching of the propellant flame. Acoustic instabilities occur when the fluctuating pressure field inside the combustion chamber corresponds to a resonant acoustic mode of the cavity defined by the topology of the propellant grain, the motor case, and nozzle. Note that the resonant modes of the rocket motor change as the propellant grain regresses. Any unstable modes can lead to catastrophic failure of the rocket motor (propellant or case failure). See the work of Culick and Yang [35], Vuillot and Casalis [208], and Culick [34] for more details on acoustic combustion instabilities.

Solid particles are often added to the propellant to enhance combustion and burning stability. The added mass of the particles tends to damp out unsteady waves present in the combustion chamber. The particles, however, can have detrimental effects on the rocket motor, causing excessive erosion of the nozzle surface, and altering the effective thrust and choking behaviour of the rocket motor. The particles added in many SRMs, such as the booster rockets on the Space Shuttle, are typically composed of aluminum. These reactive particles are used both to stabilize the internal flow due to possible combustion instabilities and to also act as fuel, producing additional hot gas into the combustion chamber as they burn. The aluminum particles can account for as much as

20% of the propellant by mass. Non-aluminized propellants are, however, common in some military rocket applications, such as the CRV-7 rocket system, and result in reduced smoke and thermal signatures. These inert particles, such as aluminum oxide, Al_2O_3, are typically added to dampen combustion instabilities in the flow chamber and account for 2–3% of the solid propellant by mass.

Ignition of the propellant is a transient process that is accomplished using a pyrotechnic device that contains boron-potassium nitrate pellets and are ignited by an electrical signal. Note that in larger rockets the igniter consists of a small SRM as well as the $BKNO_3$ pellets. The igniter provides a hot gas which raises the surface temperature of the propellant grain sufficiently for ignition to occur.

In summary, the fluid flow from the combustion chamber, through the nozzle, and the plume of the rocket is a high temperature, chemically reactive, multi-phase, turbulent flow having a wide range in Mach number. It is inherently three-dimensional due to turbulent flow, mass injection, and propellant grain geometry. In addition to the complexity of the internal flow, the flight mechanics of a rocket motor is inherently unstable. Rocket motors have no innate pitch/yaw stability, therefore, perturbations during the flight of a rocket can cause end-over-end rotation of the rocket altering the path of the rocket motor [98]. This instability can be provoked by a mis-aligned thrust vector. Spinning the rocket motor about its longitudinal axis will dampen the side components of the velocities due to the mis-aligned thrust. A stability analysis of rocket motors can be found in the book by Kolk [98]. The spinning motion of the rocket adds to the complexity of the internal flow field, significantly altering the flow structure due to the Coriolis and centrifugal forces created by the rocket rotation.

1.1.2 Numerical Prediction of SRM Internal Ballistics

Modern numerical methods are a potential tool for studying the complex characteristics of solid propellant rocket motor internal ballistics. In particular, high-fidelity simulations, in terms of numerical solution accuracy and modelling of physical phenomena would seem to offer a way forward to improving the understanding of the fluid processes involved in SRM core flows; however, due to the complex nature of SRM internal ballistics, these numerical simulations can be computationally expensive. Nevertheless, a number of engineers and scientists have successfully designed and applied numerical schemes to study these problems. Much of the recent numerical SRM research has been directed toward the investigation of the characteristics of the turbulent flow structure, the modelling of reactive and non-reactive particles (with and with-out particle-particle collision processes), and the interaction between the turbulent flow and the particle trajectories. Some of these recent numerical achievements are now summarized.

Solution of the Favre-averaged Navier-Stokes equations where turbulence closure is achieved using two-equation models is a common approach for the numerical solution of turbulent flows [211]. Tseng *et al.* [194], Ciucci and co-workers [28, 29], Orbekk [136], Sabnis and co-workers [167, 168], and Cai *et al.* [23] have all computed turbulent rocket motor core flows using several different variants of the k-ϵ turbulence model. A difficulty that arises in turbulence modelling of internal SRM flows is the specification and modelling of the mass injection and the propellant grain due to the combustion of the propellant. Tseng *et al.* [194] and Cai *et al.* [23] used the standard k-ϵ turbulence model away from solid and transpiring walls and the Wolfshtein one-equation model in the near wall region [212]. A correction to the eddy-viscosity length scale was proposed by Tseng *et al.* to account for the mass injection at the burning surface [194]. Sabnis *et al.* [167] and Ciucci and co-workers [28, 29] devised mass injection modifications to the damping functions of the low Reynolds number k-ϵ formulation [92, 101].

To provide a better understanding of the turbulent phenomena in SRM core flows, large eddy simulations (LES) have been conducted by Apte and Yang [6, 7], Berglund and Fureby [19], and Najjar *et al.* [129]. The work by Apte and Yang involved the prediction of the internal flow in a nozzleless rocket with surface mass-injection and was limited to two-dimensions. However, it is shown three flow regimes exist, fully laminar at the head of the rocket, a transitional section beginning roughly at the mid-point, and a fully turbulent region further downstream as the flow Reynolds number increases [6, 7]. The turbulent injection of mass causes early transition to turbulence. The work of Najjar *et al.* [129] was carried out at the Center for Simulation of Advanced Rockets (CSAR) at the University of Illinois at Urbana-Champaign. The goal of CSAR is the detailed, whole-system simulation of solid propellant rockets under both normal and abnormal operating conditions [79, 41, 40]. This includes modelling the fluid dynamics ([145, 127, 8, 157, 206, 50, 51, 128, 209, 20, 52, 129]), structural mechanics [130], and propellant combustion ([80, 96, 97, 4, 183]) as well as advances in computer science. Their research focuses on the Space Shuttle booster rockets. Algorithms for the coupled simulations are outlined by Parsons *et al.* [145] and Fielder *et al.* [53, 54], including computational fluid dynamics (CFD) for predicting the core flow and finite-element methods for simulating the heat transfer and stresses in the propellant grain.

Numerical simulation of the particle phase in SRM core flows has been achieved using both Lagrangian [28, 29, 23, 168] and Eulerian [207, 36, 136] methods. Lagrangian tracking methods more easily allow for the inclusion of complex particle processes (such as combustion, break-up, and agglomeration); however, Eulerian techniques can be less computationally expensive. Dupays *et al.* [44] have performed reactive multi-phase computations using both Lagrangian and Eulerian methods and their findings tend to support these statements. At CSAR, a Lagrangian formulation

is used to follow combusting aluminum droplets and aluminum oxide particles through the flow-field [127]. A fast Eulerian method was formulated to track the concentration of smoke particles through the flow-field for particles with small momentum relaxation times [8, 50, 51, 52]. In a full Eulerian approach, an additional system of partial differential equations must be solved for the particle phase simultaneously with the governing equations for the gas phase. The fast Eulerian method assumes that differences in gas-phase and particle-phase velocities are small for particles with a small momentum transfer relaxation time scale. To determine the particle-phase velocity, a perturbation analysis was conducted on a Lagrangian formulation of particle motion based on the momentum transfer relaxation time, giving the particle-phase velocity in terms of the gas-phase velocity with some acceleration effects. This simplification requires the solution of only one additional partial differential equation for the particle phase concentration. Rani and Vanka performed DNS of a particle laden pipe flow [157]. It is shown that the presence of the particle phase has a significant augmentation of the turbulence at the smaller dissipative scales near the wall and pipe centreline.

Through a simple analysis based on the conservation of mass, Kuentzmann [99] showed that the injection velocity of the combustion products from the burning surface into the combustion chamber is at least two orders of magnitude greater than the rate of propellant erosion. This justifies using a fixed combustion interface and fixed grids at different times to model the unsteady motion of the combustion interface for SRMs operating under normal conditions. However, algorithms that can account for the motion of moving interfaces can be used to model propellants burning under abnormal operating conditions, propellant slumping effects, and flexing inhibitors. At CSAR, an arbitrary Lagrangian-Eulerian (ALE) formulation is used to allow for grid deformations due to moving boundaries [54, 20]. Karimian and Amoli also performed SRM internal ballistic simulations with a moving combustion interface using an ALE approach [93]. In their work, additional triangular cells were inserted when the elements became highly stretched.

1.2 Thesis Objectives

The long-term objective of the SRM research program at the University of Toronto Institute for Aerospace studies is the development of a CFD tool that will allow for the study of the characteristics and structure of three-dimensional turbulent multi-phase SRM core flows, as well as aid rocket design and optimization. Of particular interest are the structure and characteristics of swirling core flow, the grain geometry effects on the swirl structure, the fluid and solid particle transport through the swirling core flow, and the effective thrust and choking of the rocket motor due to

rocket rotation and particle density. Other potential research directions include propellant ignition transients and igniter design, propellant combustion modelling, and the study of L^* and acoustic instabilities.

In order to achieve the longer-term research plan, the goal of the current research is to develop a computational framework for predicting two-dimensional axisymmetric turbulent multi-phase SRM core flows. The numerical algorithms presented here provide insight into the requirements for developing a high-fidelity, efficient, and robust three-dimensional tool. The key feature of this numerical scheme includes the use of a parallel block-based adaptive mesh refinement algorithm, a mesh adjustment algorithm for dynamic and stationary embedded boundaries, and a multi-velocity Eulerian formulation for the particle-phase to account for the complex physical processes associated with SRM core flows. The proposed parallel algorithm has been devised with a view to enabling the computation of rocket motor flows on a more routine basis.

1.3 Overview

This dissertation is delivered in four main Chapters. The equations governing the behaviour of the gas-particle flow are presented in the next Chapter. It begins with a discussion of the dynamics and heat transfer of single isolated solid particle moving through a background gas. This provides a Lagrangian perspective of particle motion. From there an Eulerian formulation of the particle phase is presented along with the Navier-Stokes equations for the gas phase. Eigensystem and dispersion analyses of the one-dimensional form of the Eulerian gas-particle flow equations are conducted to gain the necessary insight into the wave structure and dynamic behaviour of the system equations required for constructing an algorithm for their numerical solution. The Favre-averaged gas-particle equations used here to describe fully turbulent core flows and a discussion of the two-equation model used for turbulence closure are given in the next section. The Chapter is concluded by describing the pressure-dependent propellant combustion model used here to specify the gas and particle products at the combustion interface as well as determine the rate of regression of the propellant grain.

Algorithms for the numerical prediction of the system of equations outlined in Chapter 2 are described in Chapter 3. In this work, a Godunov-type finite volume scheme has been developed to solve the governing equations in integral form over a domain discretized into computational cells. Explicit time-marching schemes are used for time evolution for steady and unsteady flows. A parallel multigrid method is used to accelerate convergence for steady problems. The spatial discretization schemes for the gas-phase are reviewed. A multi-velocity formulation for the solution

of the particle-phase is presented. Standard single-velocity formulations are capable of predicting regions of zero particle concentration but are problematic for flows involving crossing particle trajectories or compression waves. The multi-velocity formulation described here can account for crossing particle trajectories by splitting the particle-phase into distinct velocity families which are transported separately in the flow. Switching of the particle families at solid boundaries and due to momentum transfer with the gas-phase is conducted in a manner that enforces conservation of mass, momentum, and energy. This numerical method is combined with a parallel block-based adaptive mesh refinement algorithm that is very effective in treating problems with disparate length scales while minimizing computational expense. For SRM core flows, adaptive mesh refinement (AMR) schemes will provide the required mesh density to resolve the thin boundary layers along the nozzle walls as well as at regions of interest with respect to the particle-phase, such as peak concentration zones and the interfaces between zero and non-zero concentrations. The block-based data structure lends itself naturally to domain decomposition and thereby enables efficient and scalable implementations of the algorithm on distributed-memory multi-processor architectures. Validation and numerical results are presented for a number of test cases throughout this chapter.

In Chapter 4, a scheme is described in which the body-fitted multi-block mesh is locally adjusted to arbitrarily embedded boundaries that are not necessarily aligned with the mesh. Not only does this scheme allow for rapid and robust mesh generation involving complex embedded boundaries, it also enables the solution of unsteady flow problems involving bodies and interfaces moving relative to the flow domain. The motivation for developing a mesh adjustment scheme stemmed from the desire to numerically predict flow involving complex bodies which can be moving relative to the computational domain, such as the evolution of the combustion interface in solid propellant rocket motors. Rigid body motion or evolving motion due to a physical process are considered. Evolving embedded boundaries are modelled using a parallel block-based AMR level set method. In the case of SRM core flows, this allows for the modelling and treatment of the pressure dependent burning of the solid propellant. Again, validation and demonstration numerical test cases are presented throughout this chapter.

Predicted rocket motor core flow results for a configuration typical of CRV-7 rocket systems are described in Chapter 5 using the numerical methods described in the previous two chapters. The CRV-7 rocket system contains a non-aluminized composite propellant composed of 80% oxidizer (ammonium perchlorate, AP) and 20% fuel (hydroxyl terminated polybutadiene, HTPB). As is typical in some tactical rocketry, inert aluminium oxide particles account for 2-3% of the solid propellant by mass. Therefore, the modelling of burning particles, particle-particle collision processes (such as agglomeration), and the transport of smoke are not required for the present work and are

left for future research. In addition, the propellant ignition is assumed to be instantaneous and the burning rate dependent only on the local pressure.

Finally, concluding remarks and recommendations on future work are presented in Chapter 6. Some supporting algorithms are also described in Appendices A–E.

Chapter 2

Gas-Particle Flow Equations

Both Lagrangian and Eulerian formulations are commonly used within the scientific community when treating the motion of an inert (non-reacting), dilute (negligible volume fraction), and disperse (particle-particle collisions are negligible) particle phase coupled with a fluid. As a consequence of the assumptions of an inert and dilute particle phase, there is no interaction between the phases due to mass transfer or volume effects. However, interactions between the phases occur due to momentum and heat transfer caused by the collisions between the solid particles and the gas molecules. The collision processes result in viscous drag on the particles and thermal conduction and convection of heat to and from the particle, which together tend to relax the system towards an equilibrium state in which the two phases have equal velocity and temperature.

In a Lagrangian formulation, the trajectories of individual particles are tracked (*e.g.*, Najjar *et al.* [127] and Patankar and Joseph [147, 146]). A set of ordinary differential equations is used to determine the translational and rotational motion of the particle as well as the particle's temperature. Although conceptually straight-forward, Lagrangian methods can be prohibitively expensive since a large number of particles is required to achieve realistic concentrations with a distribution of characteristics (such as mass, diameter, and specific heat). Ad hoc puff models, where simulated particles represent a cloud of actual particles, are often employed in order to achieve realistic concentrations in Lagrangian simulations.

In an Eulerian formulation, the particle-phase is treated as a continuum and a set of partial differential equations representing the conservation of mass (concentration), momentum, and energy is used to prescribe the motion of the particle-phase [173, 179, 171, 87]. The Eulerian framework can readily cope with particles having a distribution of sizes and characteristics. This can be accomplished by considering multiple families of different particles, each with their own mass, momentum, and energy. A set of governing equations is then required for each particle family. Nevertheless, it has been well-established that the set of partial differential equations governing dilute and disperse particle flow is both hyperbolic and degenerate [173, 179, 172]. As a result of the degeneracy, numerical predictions of flows involving the compression of the particle phase, such as at reflection boundary conditions, can be problematic and must be dealt with when constructing

solutions via numerical methods.

In this Chapter, the governing equations for an inert, disperse, and dilute gas-particle flow are presented. The momentum and heat transfer interactions are examined in the next section in the context of a Lagrangian formulation for the particle-phase. The dominant momentum transfer term, Stokes drag, is discussed. Corrections to the drag coefficient are presented for higher slip-Reynolds number and rarefied non-equilibrium flows (the latter is important for very small particles). Higher-order drag terms, such as the Basset history force, are addressed. Heat transfer due to conduction and convection is also discussed. An Eulerian formulation for the particle-phase is presented in the following section along with the compressible Navier-Stokes equations for the gas phase. Eigensystem and dispersion analyses of the inviscid one-dimensional form of the equations are conducted to gain a better understanding of the dynamic behaviour and wave structure of the coupled gas-particle flow equations as well as investigate the interaction mechanisms between the gas and particle phases. Turbulence modelling is discussed in the third section of this chapter. The Favre-averaged gas-particle flow equations are presented and the eddy-viscosity-based k-ω turbulence closure model, used herein to model the influences of unresolved turbulence, is discussed. The pressure-dependent burning law used in this study for the combustion of a solid propellant is described in the final section.

2.1 Lagrangian Formulation

2.1.1 Momentum Equation

A Lagrangian formulation involves tracking the trajectory of individual particles. The equation for the translational motion of the particle is described in this section. Additional ordinary differential equations are required to track the angular velocity of the particle and heat transfer between the particle and the surrounding fluid. Particle-particle interaction effects on momentum transfer, due to collisions and interacting wakes, are neglected.

The translational motion for an isolated inert, spherical, and rigid particle of diameter, d_p, moving through a background gas with velocity vector $\vec{v}$ is given by the sum of the forces acting on the particle,

$$m_p \frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = (m_p - m_f)\vec{g} + \oint_{\Omega} \vec{\sigma}_n \cdot \hat{n} \, \mathrm{d}\Omega, \tag{2.1}$$

where m_p and $\vec{u}$ correspond to the mass and velocity of the particle, respectively, and m_f is the mass of the fluid displaced by the particle. The term on the left-side of equation (2.1) is the Lagrangian acceleration of the particle and the two terms on the right side correspond to the force

due to gravity and the force due to the fluid on the surface of the particle, respectively. The surface force acts in a normal direction, $\hat{n}$, to the surface of the particle, Ω. Note that the position of the particle can be found by integrating

$$\frac{\mathrm{d}\vec{r}}{\mathrm{d}t} = \vec{u}. \tag{2.2}$$

In the case of an incompressible fluid, the stress tensor, $\overleftrightarrow{\sigma}_n$, is given by

$$\overleftrightarrow{\sigma}_n = -p\overleftrightarrow{I} + \mu\Big(\vec{\nabla}\vec{v} + (\vec{\nabla}\vec{v})^T\Big), \tag{2.3}$$

where p and μ are the pressure and dynamic viscosity of the gas. An analytical expression can be derived for the surface integral of equation (2.1) for an unsteady, incompressible flow by splitting the force contributions into the steady and unsteady components. The steady component can be integrated directly, however, the unsteady component requires a lengthy derivation in which a complex function is assumed to satisfy the unsteady component of the flow field. Fourier transforms can be used to evaluate the resulting integral. The derivation process is outlined by Maxey and Riley [121] and Landau and Lifshitz [102] and is presented in full in Ref. [169]. The ordinary differential equation governing the motion of an isolated particle is then given by

$$\begin{aligned} m_p\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} \;=\; & (m_p - m_f)\vec{g} + m_f\frac{\mathrm{D}\vec{v}}{\mathrm{D}t}\bigg|_{\vec{r}(t)} \\ & + \frac{1}{2}m_f\frac{\mathrm{d}}{\mathrm{d}t}\left(\vec{v}(\vec{r}(t),t) - \vec{u}(t) + \frac{1}{40}d_p^2\nabla^2\vec{v}\Big|_{\vec{r}(t)}\right) \\ & + 3\pi d_p\mu\left(\vec{v}(\vec{r}(t),t) - \vec{u}(t) + \frac{1}{24}d_p^2\nabla^2\vec{v}\Big|_{\vec{r}(t)}\right) \\ & + \frac{3}{2}\pi d_p^2\mu\int_{t_0}^{t}\left(\frac{\mathrm{d}/\mathrm{d}\tau\big(\vec{v}(\vec{r}(t),t) - \vec{u}(t) + \frac{1}{24}d_p^2\nabla^2\vec{v}|_{\vec{r}(t)}\big)}{\sqrt{\pi\nu(t-\tau)}}\right)\mathrm{d}\tau. \end{aligned} \tag{2.4}$$

The terms on the right-side of this equation correspond to buoyancy force, the force due to fluid acceleration, the force due to added mass, the Stokes drag force, and the Basset history force. These terms have been derived using a low slip Reynolds numbers approximation, $\mathrm{Re_p} \ll 1$, where

$$\mathrm{Re_p} = \rho d_p|\vec{v} - \vec{u}|/\mu. \tag{2.5}$$

The fluid acceleration force incorporates the effects of the fluid-stress gradients on the particle. The added mass term represents the virtual mass added to the particle as the acceleration of the particle requires the surrounding fluid to accelerate. The volume of the added mass is equal to half of the volume of the particle. The Stokes drag force corresponds to the transfer of momentum between the fluid and the solid particle. The Basset history force is associated with the unsteady flow-field

surrounding the particle and depends on the past particle motion, weighted by the kernel $(t-\tau)^{\frac{1}{2}}$, where $(t-\tau)$ represents the elapsed time since the past acceleration. It acts as an augmented viscous drag and depends on the viscosity of the fluid, the acceleration of the particle, and the acceleration of the fluid. The terms involving the Laplacian of the flow-field $\nabla^2\vec{u}$ are known as the Faxen terms. Simply stated, the Faxen terms account for the non-uniformity (or curvature) of the flow-field. The Faxen terms are generally small when compared with the other terms and so are usually neglected.

A scale analysis of equation (2.4), as conducted by Mei *et al.* [122], indicates that the Stokes drag and buoyancy terms are the dominant forces acting on the particle. The influence of the fluid acceleration, added mass, and Basset history force are negligible for heavy particles. These forces also become negligible for slip Reynolds numbers greater than one. In this limit, equation (2.4) can be reduced to

$$m_p \frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = (m_p - m_f)\vec{g} + 3\pi d_p \mu \big(\vec{v}(\vec{r}(t),t) - \vec{u}\big) f(\mathrm{Re_p})/\mathrm{G}. \tag{2.6}$$

The function $f(\mathrm{Re_p})$ is a non-linear correction that extends Stokes drag law, where $f(\mathrm{Re_p}) = 1$, which is strictly valid for $\mathrm{Re_p} < 1$ to higher slip Reynolds numbers. In particular, an empirical drag law can be written as

$$\mathrm{C_D} = \frac{24}{\mathrm{Re_p}} f(\mathrm{Re_p}) = \frac{24}{\mathrm{Re_p}} \left(1 + d_1 \mathrm{Re_p}^{d_2} + \frac{\mathrm{Re_p}}{24} \frac{d_3}{1 + d_4/\mathrm{Re_p}^{d_5}} \right), \tag{2.7}$$

where the coefficients for five drag laws are listed in Table 2.1 for rigid, spherical particles. The first set of coefficients correspond to Stokes drag. The Klyachko-Putnam drag law extends Stokes drag to moderate slip Reynolds numbers [166]. The Turton and Levenspiel [196] and Haider and Levenspiel [71] drag laws are valid for slip Reynolds numbers up to one-hundred thousand, $\mathrm{Re_p} < 10^5$. These drag laws are compared with experimentally observed values [158] for slip Reynolds numbers ranging from 0.1 to 70,000 in Figure 2.1.

Table 2.1: *Coefficients for various drag laws.*

	d_1	d_2	d_3	d_4	d_5
Stokes [166]	0	0	0	0	0
Oseen [166]	3/16	1	0	0	0
Klyachko-Putnam [166]	1/6	2/3	0	0	0
Turton and Levenspiel [196]	0.173	0.657	0.413	16300	1.09
Haider and Levenspiel [71]	0.1806	0.6459	0.4251	6880.95	1

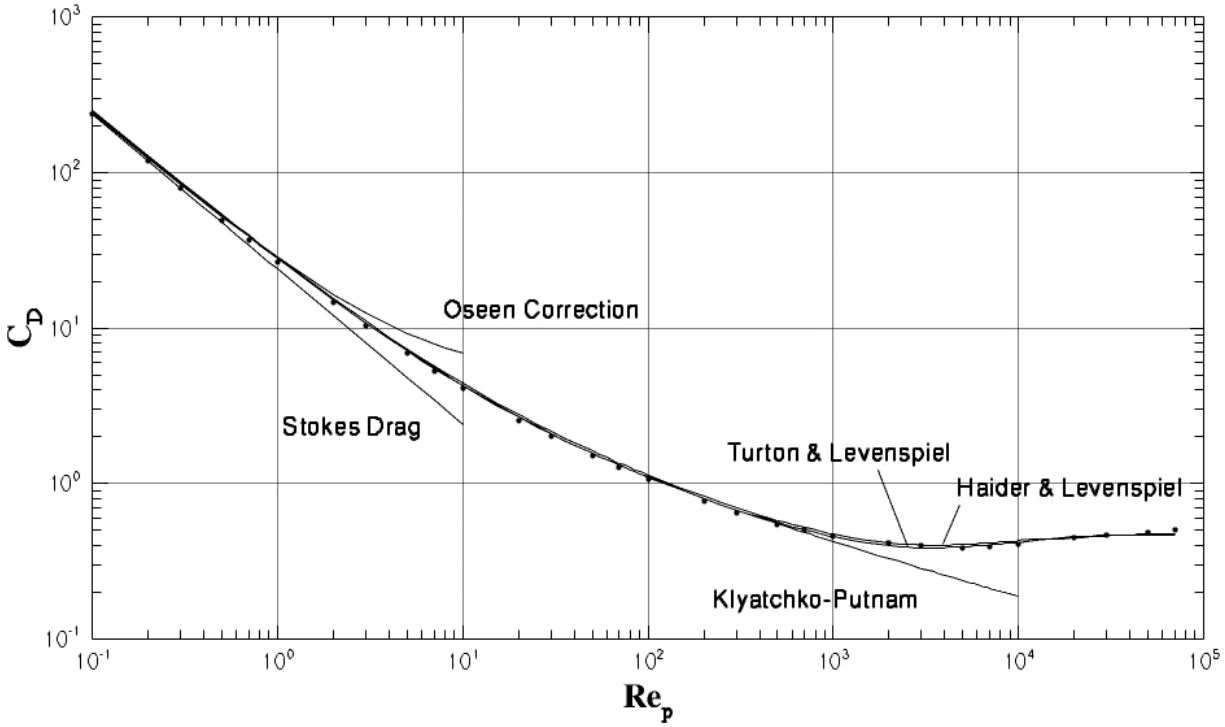


Figure 2.1: *Comparison of experimental and computed drag coefficients.*

If the particle size is small or comparable to the mean free path of the fluid, λ, then the assumption of continuum flow around the particle is invalid. In this regime, the drag laws will over predict the drag force on the particle. A correction factor, known as the Cunningham correction, G, is included in equation (2.6) to account for this effect [166], and it is given by

$$\mathrm{G} = 1 + \mathrm{Kn}\left[2.492 + 0.84\exp\left(-\frac{1.74}{\mathrm{Kn}}\right)\right], \tag{2.8}$$

where Kn is the Knudsen number which is defined as $\mathrm{Kn} = \lambda/d_p$. This correction factor increases the momentum transfer relaxation time which leads to a decreased drag force. Note that $\mathrm{G} = 1$ for continuum flows ($\mathrm{Kn} \to 0$).

2.1.2 Heat Transfer Equation

The rate of heat transfer between the single isolated particle and the surrounding fluid depends on the instantaneous temperature difference [166]. The ordinary differential equation governing the temperature of the particle is given by

$$m_p c_m \frac{\mathrm{d}T_p}{\mathrm{d}t} = \pi d_p \kappa \mathrm{Nu}(T - T_p), \tag{2.9}$$

where T_p, T, c_m, and κ are the particle's temperature, the local fluid temperature, the specific heat of the particle (assuming constant particle material), and the thermal conductivity of the fluid, respectively. The Nusselt number, Nu, accounts for the increase in heat transfer due to convection. The Nusselt number can be expressed in terms of the slip Reynolds number and the Prandtl number, $\mathrm{Pr} = \mu c_p/\kappa$, as

$$\mathrm{Nu} = 1 + 0.3\mathrm{Pr}^{1/3}\mathrm{Re}_{\mathrm{p}}^{1/2}, \tag{2.10}$$

where a Nusselt number of unity corresponds to purely conductive heat transfer.

2.2 Eulerian Formulation

2.2.1 Inert, Dilute, and Disperse Gas-Particle Equations

In order to model multi-phase flows, Lagrangian methods require the tracking of a large number of particles. In these situations, Eulerian methods, which instead provide a description of the mass, momentum, and energy of the particles in a given volume, can then be a convenient way to represent the particle dynamics. An Eulerian formulation for an inert, dilute, and disperse particle-phase coupled to a compressible fluid is now described. The compressible Navier-Stokes equations are used to govern the behaviour of the calorically perfect gas phase. The Eulerian formulation for the particle phase is presented here for N particle families. Each particle family can have distinct sizes and characteristics and is required to have their own mass, momentum, and energy. The conservation of mass (continuity equation) for the gas and particle phases are given by

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot \left(\rho \vec{v}\right) = 0, \tag{2.11}$$

$$\frac{\partial \sigma_{p_i}}{\partial t} + \vec{\nabla} \cdot \left(\sigma_{p_i} \vec{u}_i\right) = 0, \quad \text{for} \quad i = 1 \ldots N, \tag{2.12}$$

where ρ and $\vec{v}$ are the gas-phase density and velocity, and σ_{p_i} and $\vec{u}_i$ are the particle-phase concentration and velocity for the i^{th} particle family.

As noted in the previous section, a strong interaction can exist between the relatively heavy solid particles and the gas due to momentum transfer (drag) between the two phases. The conservation of momentum for the gas and particle phases are given by

$$\frac{\partial}{\partial t}\left(\rho \vec{v}\right) + \vec{\nabla} \cdot \left(\rho \vec{v}\vec{v} + p\overleftrightarrow{I}\right) = \vec{\nabla} \cdot \overleftrightarrow{\tau} - \sum_{i=1}^{N} \frac{\sigma_{p_i}}{\tau_{v_i}}\left(\vec{v} - \vec{u}_i\right), \tag{2.13}$$

$$\frac{\partial}{\partial t}\left(\sigma_{p_i} \vec{u}_i\right) + \vec{\nabla} \cdot \left(\sigma_{p_i} \vec{u}_i \vec{u}_i\right) = \frac{\sigma_{p_i}}{\tau_{v_i}}\left(\vec{v} - \vec{u}_i\right), \quad \text{for} \quad i = 1 \ldots N, \tag{2.14}$$

where p is the gas-phase pressure, and $\overline{\overline{\tau}}$ is the viscous stress tensor, $\overline{\overline{\tau}} = 2\mu\big(\vec{\nabla}\vec{v} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\overline{\overline{I}}\big)$. The momentum (drag) transfer term has been written in terms of a relaxation time, τ_v, given by

$$\tau_v = \frac{m_p}{3\pi d_p \mu}\frac{\mathrm{G}}{f(\mathrm{Re_p})}, \tag{2.15}$$

which includes both the non-linear drag and Cunningham corrections outlined in Section 2.1.1 above. Note that the momentum transfer interaction terms on the right-hand side of equations (2.13) and (2.14) are identical to the drag term for a single particle given in equation (2.6). Although the gas is assumed to be compressible, no compressibility corrections to the drag law are considered in this work.

Heat transfer between the phases can occur when the phases have different temperatures. The gas phase is taken to be calorically perfect and the total energies of the two phases are given by

$$E = \frac{p}{(\gamma-1)} + \frac{\rho}{2}(\vec{v}\cdot\vec{v}) = \rho c_v T + \frac{\rho}{2}(\vec{v}\cdot\vec{v}), \tag{2.16}$$

$$E_{p_i} = \sigma_{p_i} c_{m_i} T_{p_i} + \frac{\sigma_{p_i}}{2}(\vec{u}_i\cdot\vec{u}_i), \tag{2.17}$$

where $\gamma = c_p/c_v$ is the ratio of the specific heats for the gas and p is the gas phase pressure. The ideal gas law provides a relationship between the gas pressure, p, and temperature, T, $p = \rho \mathrm{R} T = \rho a^2/\gamma$, where $a = \sqrt{\gamma \mathrm{R} T}$ is the sound speed and R is the gas constant. The conservation of energy for the gas and solid particle phases can be expressed in terms of their total energies per unit mass and written as

$$\frac{\partial E}{\partial t} + \vec{\nabla}\cdot\Big[\vec{v}\big(E+p\big)\Big] = \vec{\nabla}\cdot\big(\vec{v}\cdot\overline{\overline{\tau}} - \vec{q}\big) - \sum_{i=1}^{N}\left[\frac{\sigma_{p_i}}{\tau_{v_i}}\big(\vec{v}-\vec{u}_i\big)\cdot\vec{u}_i + \frac{\sigma_{p_i}c_p}{\tau_{T_i}}\big(T-T_{p_i}\big)\right], \tag{2.18}$$

$$\frac{\partial E_{p_i}}{\partial t} + \vec{\nabla}\cdot\big(\vec{u}_i E_{p_i}\big) = \frac{\sigma_{p_i}}{\tau_{v_i}}\big(\vec{v}-\vec{u}_i\big)\cdot\vec{u}_i + \frac{\sigma_{p_i}c_p}{\tau_{T_i}}\big(T-T_{p_i}\big), \quad \text{for} \quad i = 1\ldots N, \tag{2.19}$$

where $\vec{q}$ is the heat transfer vector, $\vec{q} = -\kappa\vec{\nabla}T$.

Igra *et al.* [87] observed that due to the lack of the pressure-like term in the particle-phase, the particle-phase total energy equation can be simplified into a thermal energy equation while maintaining conservative form by subtracting the particle-phase continuity and momentum equations from it. The particle-phase thermal energy equation is given by

$$\frac{\partial \epsilon_{p_i}}{\partial t} + \vec{\nabla}\cdot\big(\vec{u}_i\epsilon_{p_i}\big) = \frac{\sigma_{p_i}c_p}{\tau_{T_i}}\big(T-T_{p_i}\big), \quad \text{for} \quad i = 1\ldots N, \tag{2.20}$$

where $\epsilon_{p_i} = \sigma_{p_i} c_{m_i} T_{p_i}$ is the particle-phase total thermal energy of the i^{th} family and the relaxation time associated with the heat transfer, τ_T, is given by

$$\tau_T = \frac{m_p c_p}{2\pi d_p \kappa \mathrm{Nu}}. \tag{2.21}$$

The heat transfer terms on the right-hand side of equations (2.18)–(2.20) are identical to the heat transfer term for a single particle given in equation (2.9). Note that the ratio of the relaxation times, equations (2.15) and (2.21), can be related to the Prandtl number, Pr, by $\tau_T/\tau_v = \frac{3}{2}\mu c_p/\kappa = \frac{3}{2}\mathrm{Pr}$ when the non-linear drag correction, Cunningham correction, and convective heat transfer terms are ignored.

2.2.2 Eigensystem Analysis of the Inviscid Gas-Particle Equations

Equations (2.12), (2.14), and (2.20) provide an Eulerian description for the particle phase and are very similar to the Euler equations for inviscid, compressible gases with the exception of the absence of pressure-like terms in the particle-phase momentum or energy equations. This feature is a direct result of the assumption that the particle-phase is disperse. The lack of particle-particle collisions means that there are no normal surface forces produced by the random motion of the particles themselves and hence there are no pressure-like terms (or viscous-like terms) in the particle momentum and energy equations. The eigenstructure of the inviscid one-dimensional form of the preceding two-phase flow equations is now considered in order to gain an understanding of the influence that the absence of a particle pressure has on the unsteady wave structure of this system of equations before pursuing a numerical solution.

The one-dimensional, weak conservation form of the governing equations in the case of a single-family mono-sized particle phase can be summarized by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial}{\partial x}\mathbf{F}(\mathbf{U}) = \mathbf{P}(\mathbf{U}), \tag{2.22}$$

where $\mathbf{U} = [\,\rho, \rho v, E, \sigma_p, \sigma_p u, \epsilon_p]^T$ is the one-dimensional flow state vector of conserved variables, $\mathbf{F}(\mathbf{U})$ is the flux vector, and $\mathbf{P}(\mathbf{U})$ is the phase-interaction source vector. In the "frozen" limit, the characteristic times scales of the particle drag and heat transfer, τ_v and τ_T, are assumed to be large relative to the differences in velocities and temperatures of the two phases such that the phase-interaction terms can be neglected. In this limit, the equations governing the gas and particle phases decouple. An eigensystem analysis of the system in the frozen limit provides the set of real eigenvalues, λ_k,

$$\lambda_1 = v - a, \quad \lambda_2 = v, \quad \lambda_3 = v + a, \quad \lambda_{4,5,6} = u, \tag{2.23}$$

indicating the hyperbolic nature of the equations. The first three eigenvalues correspond to the well known acoustic and convective eigenvalues of the Euler equations. The last three repeated eigenvalues are associated with the particle phase. The right eigenvectors corresponding to the

frozen-limit eigenvalues are

$$\begin{bmatrix} \vec{r}_1 \\ \vec{r}_2 \\ \vec{r}_3 \\ \vec{r}_4 \\ \vec{r}_5 \end{bmatrix} = \begin{bmatrix} 1 & -a/\rho & a^2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & a/\rho & a^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.24}$$

This is an incomplete set of eigenvectors as only five linearly independent eigenvectors can be found for the six characteristic fields. An eigenvector associated with the $\lambda = u$ eigenvalue is missing. Because of this degeneracy in the eigensystem, the set of hyperbolic conservation equations governing disperse gas-particle flows is said to to form a degenerate hyperbolic set, a result confirmed by previous studies [172, 173, 179]. The degeneracy is a direct result of the assumptions that the particle volume fraction is negligible (dilute) and that the effects of inter-particle collisions are not important (disperse). Physically, the main implications of the degeneracy are two-fold. First, particle vacuums can exist as there are no direct pressure forces to drive the particles from regions of high concentrations to those with lower concentrations. Moreover, as interactions between particles have been neglected, the paths of particles can cross. Faster moving particles can freely over-take and pass slower moving particles without being subject to any particle-particle interaction forces. As a result of this degeneracy, numerical predictions of flows involving the compression of the particle-phase, such as at reflection boundary conditions, can be problematic and this issue must be dealt with when constructing solutions via numerical methods.

2.2.3 Dispersion Analysis of the Inviscid Gas-Particle Equations

A linear dispersion analysis for initial value problems can be used to provide a better understanding of the dynamic behaviour and wave structure of the coupled gas-particle flow equations as well as the degeneracy of the particle phase equations. In particular, an analysis of initial value problems for the linearized one-dimensional form of these equations is considered here. Marble conducted a linear dispersion analysis of the boundary value problem for the same system of equations [118]. The analysis of the equilibrium, frozen, and non-equilibrium dispersion relationships follows the procedure outlined by Groth *et al.* [69].

The dispersive wave properties of the gas-particle flow equations for planar wave propagation can be assessed by considering the linearized primitive form of equation (2.22):

$$\frac{\partial \mathbf{W}}{\partial t} + \mathbf{A}_0 \frac{\partial \mathbf{W}}{\partial x} = \mathbf{Q}_0 \mathbf{W}, \tag{2.25}$$

where $\mathbf{W} = [\,\rho, v, T, \sigma_p, u, T_p]^T$ is the primitive solution vector and the matrices $\mathbf{A}_0$ and $\mathbf{Q}_0$ are the linearized coefficient matrix and phase-interaction source Jacobian determined by linearizing the primitive form of the equations about a quiescent (stationary) equilibrium state $\mathbf{W}_0 = [\,\rho_0, 0, T_0, \sigma_{p_0}, 0, T_0]^T$. For initial value problems and solutions of the form $\mathbf{W} = \exp[i(\omega t - \xi x)]$, the differential wave operator for the linearized equations is defined by the the eigenvalue problem given by $\big[i\omega\mathbf{I} - i\xi\mathbf{A}_0 - \mathbf{Q}_0\big]\mathbf{W} = 0$ where the temporal frequency, ω, can have real and imaginary components, $\omega = \omega_R + i\omega_I$, whereas the spatial frequency ξ is real valued. The parameter ξ defines the frequency content of the linearized solution, which varies from the low frequencies for near equilibrium solutions to high frequencies for non-equilibrium solutions approaching the frozen limit. The condition for the stability of the solutions is $\omega_I > 0$.

In the equilibrium limit ($\tau_v, \tau_T \rightarrow 0$), the particles and gas are driven by the drag forces and heat transfer to move with the same velocity ($u = v$) and have the same temperature ($T_p = T$). An equilibrium density can be defined by setting $\overline{\rho} = \rho + \sigma_p = \rho(1 + \chi)$ where $\chi = \sigma_p/\rho$ is the particle loading factor and is constant for uniformly distributed particles. The system of equations in the equilibrium limit can then be reduced to three variables, in terms of the primitive variables $\overline{\rho}$, v, and T. In this case, the linearized non-dimensional coefficient matrix is given by

$$\mathbf{A}_0 = \begin{bmatrix} 0 & 1 & 0 \\ \overline{\mathrm{R}}_0 T_0/a_0^2 & 0 & \overline{\mathrm{R}}_0 T_0/a_0^2 \\ 0 & \overline{\mathrm{R}}_0/\overline{c_v}_0 & 0 \end{bmatrix}, \tag{2.26}$$

where T_0 is the temperature of the quiescent equilibrium state, $a_0 = \sqrt{\gamma \mathrm{R} T_0}$ is the sound-speed of the quiescent equilibrium state and is used as a reference velocity in the dispersion analysis, the equilibrium gas constant and specific heat ratios are given by $\overline{\mathrm{R}}_0 = \overline{c_p}_0 - \overline{c_v}_0 = \mathrm{R}/(1 + \chi_0)$, $\overline{c_v}_0 = (c_v + \chi_0 c_m)/(1 + \chi_0)$, $\overline{c_p}_0 = (c_p + \chi_0 c_m)/(1 + \chi_0)$, and $\overline{\gamma}_0 = \overline{c_p}_0/\overline{c_v}_0$. The parameter χ_0 is the loading ratio of the reference equilibrium state. The solution of the eigenvalue problem, with $\lambda = \omega/\xi$, provides the equilibrium dispersion relationship

$$-\lambda\left[\,\lambda^2 - \overline{a}^2/a_0^2\right] = 0, \tag{2.27}$$

where $\overline{a} = \sqrt{\overline{\gamma}_0 \overline{\mathrm{R}}_0 T_0}$ is the mixture sound-speed. Determination of the roots of this polynomial gives the following non-dimensional equilibrium wave speeds

$$\lambda_1 = -\overline{a}/a_0 = -\sqrt{\overline{\gamma}_0/\gamma}, \quad \lambda_2 = 0, \quad \lambda_3 = \overline{a}/a_0 = \sqrt{\overline{\gamma}_0/\gamma}. \tag{2.28}$$

In the equilibrium limit, the imaginary part of the wave-speed is zero, $\lambda_I = 0$, which corresponds to non-dissipative, coherent wave propagation.

Now consider the linearized equations in the non-equilibrium limit. The linearized and non-dimensionalized coefficient and source Jacobians are

$$\mathbf{A}_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \mathrm{R}T_0/a_0^2 & 0 & \mathrm{R}T_0/a_0^2 & 0 & 0 & 0 \\ 0 & \mathrm{R}/c_v & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.29}$$

and

$$\mathbf{Q}_0 = \frac{t_0}{\tau_T} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{3}{2}\chi_0\mathrm{Pr} & 0 & 0 & \frac{3}{2}\chi_0\mathrm{Pr} & 0 \\ 0 & 0 & -\gamma\chi_0 & 0 & 0 & \gamma\chi_0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{3}{2}\mathrm{Pr} & 0 & 0 & -\frac{3}{2}\mathrm{Pr} & 0 \\ 0 & 0 & c_p/c_m & 0 & 0 & -c_p/c_m \end{bmatrix}, \tag{2.30}$$

respectively. In the frozen limit, the characteristic time scales of the viscous drag and heat transfer terms, τ_v and τ_T, are assumed to be large relative to the differences in velocity and temperature such that the source matrix $\mathbf{Q}_0$ can be neglected. The dispersion relationship in the frozen limit is found by solving

$$\det\big(\lambda\mathbf{I} - \mathbf{A}_0\big) = -\lambda^4\Big[\lambda^2 - 1\Big] = 0 \tag{2.31}$$

where $\lambda = \omega/\xi$. Solution of this polynomial equation yields the non-dimensional frozen wave speeds

$$\lambda_1 = -1, \quad \lambda_{2,4,5,6} = 0, \quad \lambda_3 = 1. \tag{2.32}$$

As in the near-equilibrium limit, the wave-speeds in the frozen limit show non-dissipative, coherent wave propagation since $\lambda_I = 0$ for each wave, including the four convective waves (three associated with the particle phase) and two acoustic waves (associated with the gas phase). These are the same eigenvalues as found in the eigensystem analysis of the previous subsection with $u = 0$.

The dispersion relationship for the general non-equilibrium system can be determined by solving the eigenvalue problem for the characteristic wave-speeds, $\det\big(i\omega\mathbf{I} - i\xi\mathbf{A}_0 - \mathbf{Q}_0\big) = 0$. The resulting dispersion relationship is

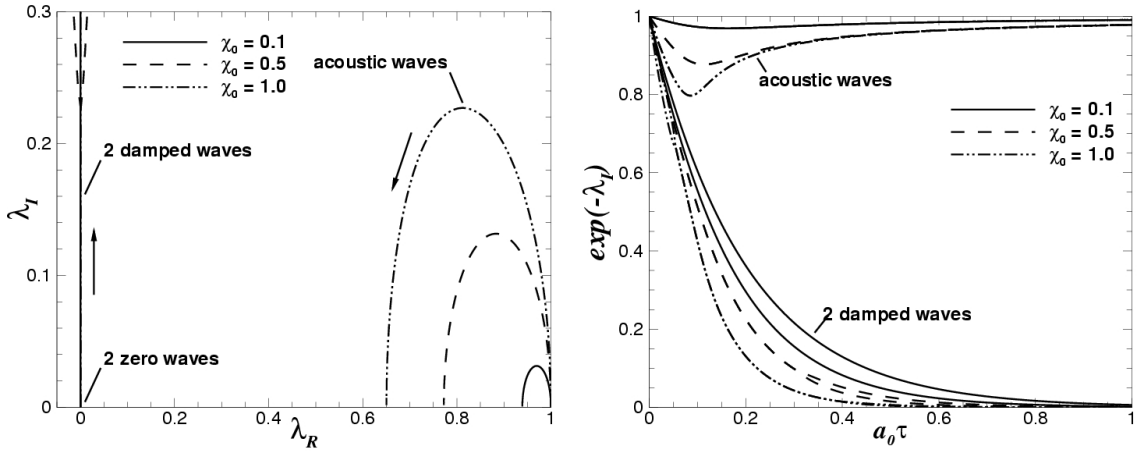
$$\lambda^2(\lambda^4 - ic_3\lambda^3 - c_2\lambda^2 + ic_1\lambda + c_0) = 0 \tag{2.33}$$

where $\lambda = \omega/\xi$. The spatial frequency, ξ, can be expressed as $\xi = 2\pi x_0/L = a_0 t_0/a_0\tau = t_0/\tau$, where t_0 is a reference time scale and $L = a_0\tau$ is the wavelength of the solution. The frozen and equilibrium limits are defined by high and low frequency solution limits (i.e., $\tau \to 0$ and $\tau \to \infty$, respectively). The coefficients c_3, c_2, c_1, and c_0 are given by

$$
\begin{aligned}
c_3 &= \tilde{\tau}(1+\chi_0)\left(\gamma\frac{\overline{c_v}_0}{c_m} + \frac{3}{2}\mathrm{Pr}\right),\\
c_2 &= 1 + \tilde{\tau}^2\left(\frac{3}{2}\mathrm{Pr}\gamma(1+\chi_0)^2\frac{\overline{c_v}_0}{c_m}\right),\\
c_1 &= \tilde{\tau}\left(\frac{3}{2}\mathrm{Pr} + (1+\chi_0)\frac{\overline{c_p}_0}{c_m}\right),\\
c_0 &= \tilde{\tau}^2\left(\frac{3}{2}\mathrm{Pr}(1+\chi_0)\frac{\overline{c_p}_0}{c_m}\right),
\end{aligned}
$$

where $\tilde{\tau} = \tau/\tau_T$ is a non-dimensional relaxation time-scale. In the general non-equilibrium case, the non-dimensional wave speeds, λ, can have real and imaginary components, $\lambda = \lambda_R + i\lambda_I$, characteristic of dispersive wave propagation. Analytic expressions for the non-dimensional wave speeds cannot be found; however, it is instructive to investigate the dispersion relationship for the non-equilibrium system by numerically calculating the waves speeds as a function of the solution frequency (as controlled by $\tilde{\tau}$). The coefficients of the non-equilibrium dispersion relationship are functions of various non-dimensional numbers: Pr, χ_0, γ, $\overline{c_p}_0/c_m$, and $\overline{c_v}_0/c_m$. The ratios of the equilibrium specific heat at constant pressure and volume with respect to the specific heat of the non-reacting particle, $\overline{c_p}_0/c_m$ and $\overline{c_v}_0/c_m$, are dependent on the gas-type, particle material, and the particle loading factor of the quiescent equilibrium state whereas the ratio of the gas-phase specific heats is dependent only on the gas. For the purpose of the current analysis, calorically perfect air, (γ, c_p) = (1.4, 1004.72 J/kg-K) at a density and temperature of 1.0 kg/m^3 and 300 K, respectively, and glass beads, c_m = 840 J/kg-K, were chosen to specify the physical properties of the gas and particle-phases. The effect of the frequency content of the linearized solution on the dispersive wave behaviour can now be investigated for various loading factors, χ_0, and Prandtl numbers, Pr.

Three loading factors were chosen for this analysis: $\chi_0 = 0.1$, 0.5, and 1. The particle volume fraction at the maximum loading factor is 4.2×10^{-4} which corresponds to an equilibrium sound speed of 65% of the frozen sound speed. The assumptions of negligible particle volume fraction (dilute) and particle-particle collisions (disperse) begin to breakdown in the vicinity of this loading factor. The Prandtl number controls the relative equilibrium of the two phases between the momentum and heat transfer rates. A Prandtl number of unity indicates that momentum and heat transfer occur at similar rates. Small Prandtl numbers correspond to a system that will reach temperature equilibrium much quicker than that of momentum. Conversely, a large Prandtl number



(a) (b)

Figure 2.2: *(a) Dispersion relationship and (b) attenuation rates for a Prandtl number of unity,* Pr*, with various loading factors.*

corresponds to a system that will reach momentum equilibrium much quicker than that of temperature. Therefore, the dispersion relationship is calculated for each of the loading factors listed above for Prandtl numbers of $\mathrm{Pr} = 1$, $\mathrm{Pr} = 10^{-3}$, and $\mathrm{Pr} = 10^{3}$ and are plotted in Figures 2.2(a), 2.3(a), and 2.3(b), respectively. The arrows indicate the direction of change from the frozen limit (high frequency solution content) to the equilibrium limit (low frequency solution content).

Due to the symmetry of the pair of acoustic modes in the imaginary axis, only one of the acoustic modes is shown in the dispersion relationship figures. As indicated previously in equation (2.32), four zero wave-speeds and two acoustic modes exist in the frozen limit. Two of the zero wave-speed modes remain undamped for all solution frequencies and are associated with convective transport of the gas and particle phase densities. The other two zero wave-speed modes are also convective modes associated with the transport of differences in the velocities and temperatures of the two phases. These two modes are undamped in the frozen limit but are highly attenuated in the equilibrium limit. This is confirmed by the equilibrium wave solution determined previously, equation (2.28), where the gas and solid phases achieve equal velocity and temperature. The attenuation rates for the two damped and acoustic modes as a function of the solution wavelength are plotted in Figure 2.2(b) for a Prandtl number of unity. This plot shows that the two purely damped modes quickly become highly damped as the wavelength increases (frequency decreases). The acoustic modes have finite damping for more intermediate solution frequencies and become undamped once again as the non-equilibrium frozen limit is reached. The zero damping of the acoustic modes in both the equilibrium and frozen limits is clearly depicted in Figure 2.2(b) for a Prandtl number of unity. It is seen in Figure 2.3(a) for a system nearly in temperature equilibrium

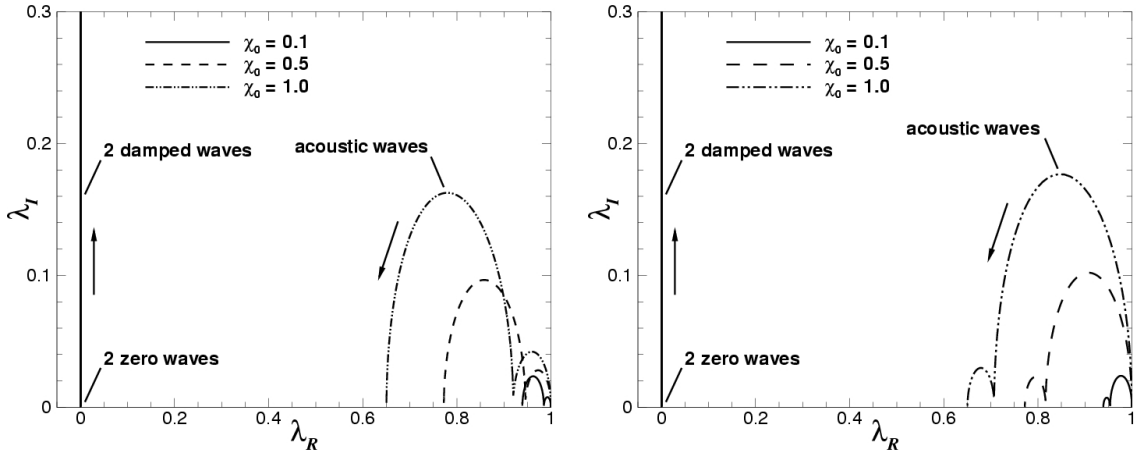


Figure 2.3: *Dispersion relationships for Prandtl numbers of (a)* $\mathrm{Pr} = 10^{-3}$ *and (b)* $\mathrm{Pr} = 10^{3}$ *with various loading factors.*

that an intermediate temperature equilibrium value of the acoustic wavespeed (with finite damping) occurs before the system reaches full equilibrium. Similarly, as shown in Figure 2.3(b) for a system nearly in momentum equilibrium an intermediate momentum equilibrium value of the acoustic wavespeed (with finite damping) occurs before the system fully reaches equilibrium.

The preceding dispersion analysis provides a complete description of how the two-phase flow equations progress from a degenerate hyperbolic system in the frozen limit to a strictly hyperbolic system in the equilibrium limit. This more complete picture of the degenerate nature of the Eulerian description was particularly helpful in formulating the numerical approach described in the next Chapter. Note also the interleaving of the non-equilibrium acoustic wave speeds between the frozen and equilibrium limits. This ensures that the frozen acoustic wavespeeds are the maximum attainable wavespeed. Therefore, in spite of the existence of the degeneracy in the non-equilibrium limit, stable solutions of the two-phase flow equations using upwind numerical schemes are attainable.

2.3 Turbulence Modelling

The Reynolds number, Re, gives an indication of whether the fluid flow under consideration is dominated by viscous or inertial effects. Viscous forces dominate the fluid motion at low Reynolds numbers and the fluid flow is said to be laminar. At higher Reynolds numbers the interaction between the non-linear inertial and viscous terms leads to instability and the fluid flow becomes irregular. This irregular condition corresponds to turbulent flow in which the flow quantities can be closely approximated as random fluctuations about a mean flow. Turbulent motion can be

characterized by an intense local swirling or eddying motion. These turbulent eddies appear in a wide range of sizes, the largest of which carry the bulk of the turbulent energy. The large eddies are responsible for the increased mixing and stresses present in turbulent motion. The energy of a turbulent flow is cascaded from the largest eddies to the smallest eddies which, in turn, dissipate energy into heat through viscous interaction. Though the Navier-Stokes equations fully describe the turbulent motion of a fluid (the smallest turbulent eddy length-scale, the Kolmogorov scale, is still much greater than that of the mean-free path of the gas) it is not practical to directly perform numerical simulations of problems of engineering interest due to the computational resources required to resolve those smallest turbulent eddies. Therefore, a statistically-averaged form of the Navier-Stokes equations is solved for the mean flow quantities. The averaging procedures, and turbulence modelling in general, is thoroughly described in the textbook by Wilcox [211]. Nevertheless, for completeness a brief description is provided next.

In the averaging procedure for the Navier-Stokes equations, an instantaneous flow quantity, $\phi(\vec{x},t)$, can be expressed as the sum of mean, $\overline{\phi}(\vec{x},t)$, and fluctuating components, $\phi'(\vec{x},t)$, such that

$$\phi(\vec{x},t) = \overline{\phi}(\vec{x},t) + \phi'(\vec{x},t). \tag{2.34}$$

The Reynolds-average, or time-average, of the state quantity is given by

$$\overline{\phi}(\vec{x},t) = \frac{1}{T}\int_{t}^{t+T} \phi(\vec{x},\tau)\mathrm{d}\tau, \tag{2.35}$$

where the time-average of the fluctuating part is zero. The time-scale, T, must be much greater than the time-scale of the turbulent fluctuations of the flow quantity but at the same time much less than the time-scale of any temporal variations of the mean flow quantities (i.e., there is a requirement for a separation between the scales of the turbulent fluctuations and mean flow variations). Reynolds-averaging the Navier-Stokes equations leads to a set of equations with correlations that must be closed via additional modelling. For example, the Reynolds average of the product of two quantities, ϕ and ψ, is $\overline{\phi\psi} = \overline{\phi}\,\overline{\psi} + \overline{\phi'\psi'}$. The quantities ϕ' and ψ' are said to be correlated if $\overline{\phi'\psi'} \neq 0$ and uncorrelated otherwise.

For the compressible Navier-Stokes equations, *mass-averaging* or *Favre-averaging* is introduced to provide a mathematical simplification of the Reynolds-averaged equations. Favre-averaging removes the density fluctuations from the averaged equations by performing a mass-average of velocity and temperature instead of a time-average. Here, these flow properties are decomposed as

$$\phi(\vec{x},t) = \tilde{\phi}(\vec{x},t) + \phi''(\vec{x},t), \tag{2.36}$$

where $\tilde{\phi}(\vec{x},t)$ is the mass-averaged component and $\phi''(\vec{x},t)$ is the fluctuating part. The mass-average is defined by

$$\overline{\rho}\tilde{\phi}(\vec{x},t) = \frac{1}{T}\int_{t}^{t+T} \rho\phi(\vec{x},\tau)\mathrm{d}\tau, \tag{2.37}$$

where $\overline{\rho}$ is the Reynolds-averaged density.

Favre-averaging of the equations governing the gas-particle flow equations is used herein and is presented in the next section. The statistical correlations that result from the averaging require a turbulence model to close the system of equations. The two equation k-ω turbulence model [211] is used in this work and is also described in what follows. The values for the closure coefficients, turbulence boundary conditions, and the methods for determining the dimensionless wall distance are also outlined.

2.3.1 Favre-Averaged Gas-Particle Equations

Favre-averaging of the equations governing the motion of the dilute and disperse gas-particle flow is performed by density-averaging the gas-phase variables as is normally done for the compressible Navier-Stokes equations and concentration-averaging the particle-phase variables. Note that when applying this averaging procedure, the phase-interaction relaxation time-scales are assumed to be based solely on the mean flow quantities. For the sake of clarity, only a single particle-phase family is considered here. The result of this averaging procedure is

$$\frac{\partial\overline{\rho}}{\partial t} + \vec{\nabla}\cdot\left(\overline{\rho}\tilde{\vec{v}}\right) = 0, \tag{2.38}$$

$$\frac{\partial}{\partial t}(\overline{\rho}\tilde{\vec{v}}) + \vec{\nabla}\cdot\left[\overline{\rho}\tilde{\vec{v}}\tilde{\vec{v}} + \overline{p}\vec{\vec{I}}\right] = \vec{\nabla}\cdot\left[\overline{\vec{\tau}} - \overline{\rho\vec{v}''\vec{v}''}\right] - \frac{\overline{\sigma}_p}{\tau_v}\left(\tilde{\vec{v}}-\tilde{\vec{u}}\right) - \frac{1}{\tau_v}\overline{\sigma_p\vec{v}''}, \tag{2.39}$$

$$\begin{aligned}\frac{\partial\tilde{E}}{\partial t} + \vec{\nabla}\cdot\left[\tilde{\vec{v}}\left(\tilde{E}+\overline{p}\right)\right] = \vec{\nabla}\cdot\Big[\tilde{\vec{v}}\cdot\left(\overline{\vec{\tau}} - \overline{\rho\vec{v}''\vec{v}''}\right) - \overline{\vec{q}} - c_p\overline{\rho\vec{v}''T''} + \overline{\vec{v}''\cdot\vec{\tau}} - \overline{\vec{v}''\tfrac{1}{2}\rho\vec{v}''\cdot\vec{v}''}\Big] - \\ \left[\frac{\overline{\sigma}_p}{\tau_v}\left(\tilde{\vec{v}}-\tilde{\vec{u}}\right)\cdot\tilde{\vec{u}} + \frac{1}{\tau_v}\overline{\sigma_p\left(\vec{v}''-\vec{u}''\right)\cdot\vec{u}''} + \frac{1}{\tau_v}\overline{\sigma_p\vec{v}''\cdot\tilde{\vec{u}}} + \frac{\overline{\sigma}_p c_p}{\tau_T}\left(\tilde{T}-\tilde{T}_p\right) + \frac{c_p}{\tau_T}\overline{\sigma_p T''}\right],\end{aligned} \tag{2.40}$$

$$\frac{\partial\overline{\sigma}_p}{\partial t} + \vec{\nabla}\cdot\left(\overline{\sigma}_p\tilde{\vec{u}}\right) = 0, \tag{2.41}$$

$$\frac{\partial}{\partial t}(\overline{\sigma}_p\tilde{\vec{u}}) + \vec{\nabla}\cdot\left(\overline{\sigma}_p\tilde{\vec{u}}\tilde{\vec{u}}\right) = -\vec{\nabla}\cdot\left(\overline{\sigma_p\vec{u}''\vec{u}''}\right) + \frac{\overline{\sigma}_p}{\tau_v}\left(\tilde{\vec{v}}-\tilde{\vec{u}}\right) + \frac{1}{\tau_v}\overline{\sigma_p\vec{v}''}, \tag{2.42}$$

$$\frac{\partial\tilde{\epsilon}_p}{\partial t} + \vec{\nabla}\cdot\left(\tilde{\vec{u}}\tilde{\epsilon}_p\right) = -\vec{\nabla}\cdot\left(c_m\overline{\sigma_p\vec{u}''T_p''}\right) + \frac{\overline{\sigma}_p c_p}{\tau_T}\left(\tilde{T}-\tilde{T}_p\right) + \frac{c_p}{\tau_T}\overline{\sigma_p T''}. \tag{2.43}$$

The Favre-averaged total gas-phase energy and particle-phase thermal energy are

$$\tilde{E} = \frac{\overline{p}}{(\gamma-1)} + \frac{\overline{\rho}}{2}(\tilde{\vec{v}}\cdot\tilde{\vec{v}}) + \frac{1}{2}\overline{\rho\vec{v}''\cdot\vec{v}''}, \tag{2.44}$$

$$\tilde{\epsilon}_p = \overline{\sigma}_p c_m\tilde{T}_p. \tag{2.45}$$

Ten different statistical correlations result from the averaging procedure. In order to obtain the mean-flow solution of the gas-particle flow equations, additional relations for these terms must be provided.

The statistical correlation that appears in the gas-phase total energy equation, equation (2.44), corresponds to the kinetic energy due to the gas-phase velocity fluctuations,

$$\rho k = \frac{1}{2}\overline{\rho\vec{v}''\cdot\vec{v}''}. \tag{2.46}$$

An equation for the turbulent kinetic energy can be derived by taking the Favre-average of the inner product of the fluctuating velocity and the gas-phase momentum equation,

$$\begin{aligned}\frac{\partial}{\partial t}(\overline{\rho}k) + \vec{\nabla}\cdot\left(\overline{\rho}\tilde{\vec{v}}k\right) = &-\overline{\rho\vec{v}''\vec{v}''} : \vec{\nabla}\tilde{\vec{v}} - \overline{\overline{\tau} : \vec{\nabla}\vec{v}''} + \vec{\nabla}\cdot\left[\overline{\vec{v}''\cdot\overline{\tau}} - \overline{\vec{v}''\tfrac{1}{2}\rho\vec{v}''\cdot\vec{v}''} - \overline{p'\vec{v}''}\right] - \\ &\overline{\vec{v}''}\cdot\vec{\nabla}\overline{p} + \overline{p'\vec{\nabla}\cdot\vec{v}''} - \left[\frac{\overline{\sigma}_p}{\tau_v}\left(\tilde{\vec{v}}-\tilde{\vec{u}}\right)\cdot\tilde{\vec{v}} + \frac{1}{\tau_v}\overline{\sigma_p\left(\vec{v}''-\vec{u}''\right)\cdot\vec{v}''} + \frac{1}{\tau_v}\overline{\sigma_p\vec{v}''\cdot\vec{v}''}\right].\end{aligned} \tag{2.47}$$

This equation includes five additional statistical correlations that must be prescribed to achieve closure.

2.3.2 Turbulence Closure

Prescription of the fifteen statistical correlations produced by the Favre-averaged of the gas-particle flow equations are addressed in this section. In terms of the influence of the particles on the gas-phase turbulent flow, Peirano and Leckner discussed the classification of turbulent flow regimes for gas-particle flows and identified five mechanisms for turbulent modulation: (1) dissipation of turbulent kinetic energy by the particles, (2) increase of turbulent viscosity due to the presence of the particles, (3) shedding of vorticies behind the particles (turbulent production), (4) fluid moving with the particles as added fluid mass, and (5) enhancement of the velocity gradient between two particles [148]. Mechanisms (2) and (5) can be neglected for a dilute and disperse particle-phase. Particle diameters considered in this thesis are small ($d_p \sim 1$ μm) and, therefore, turbulent production mechanisms (3) and (4) can also be ignored. Due to the large relative mass of the particles the dissipation of turbulent kinetic energy by the particles may be important, however, this mechanism is also neglected in this study. Therefore, for all of the preceding reasons, the statistical correlations related to the phase-interaction terms are all neglected.

Since the phase-interaction statistical correlations can be neglected, the remaining correlations can be addressed separately with their own phase. In terms of the particle-phase, the stress tensor, $-\overline{\sigma_p\vec{u}''\vec{u}''}$, and heat flux vector, $c_m\overline{\sigma_p\vec{u}''T_p''}$, found in the Favre-averaged particle-phase momentum and energy equations, equations (2.42) and (2.43), represent the rate of momentum and heat

transfer due to fluctuations in the particle phase. Due to the high mass of the particles considered in this work, the random motions of these particles are assumed to be negligible. Therefore, these terms are also neglected.

The k-ω turbulence model of Wilcox [211] is employed to achieve closure of the gas-phase equations above. This requires the solution of two additional partial differential equations for the turbulent kinetic energy, k, and the specific dissipation rate, ω. The k-ω turbulence model makes use of the Boussinesq approximation and relates the Reynolds stresses, the $-\overline{\rho\vec{v}''\vec{v}''}$ correlation, to the strain rate via the dynamic turbulent eddy-viscosity, μ_T, by

$$\overleftrightarrow{\lambda} = -\overline{\rho\vec{v}''\vec{v}''} = 2\mu_T\left(\vec{\nabla}\vec{v} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\overleftrightarrow{I}\right) - \frac{2}{3}\overline{\rho}k\overleftrightarrow{I}, \tag{2.48}$$

where the turbulent eddy-viscosity is given by the ratio of the turbulent kinetic energy and the specific dissipation rate, $\nu_T = \mu_T/\overline{\rho} = k/\omega$. Note that the last term is there for consistency and guarantees that the contraction of the Reynolds stress tensor is $-2\overline{\rho}k$ as should be expected.

In an analogous manner, the turbulent heat-flux correlation vector, $c_p\overline{\rho\vec{v}''T''}$, is assumed to be proportional to the mean temperature gradient,

$$\vec{q}_T = c_p\overline{\rho\vec{v}''T''} = -\kappa_T\vec{\nabla}\tilde{T}, \tag{2.49}$$

where the turbulent thermal conductivity, κ_T, can be expressed via a Reynolds analogy in terms of the eddy-viscosity and a turbulent Prandtl number, Pr_T, as $\kappa_T = \mu_T c_p/\mathrm{Pr}_T$. The turbulent Prandtl number is usually assumed to be constant, and $\mathrm{Pr}_T = 0.9$ is typical for many turbulent flows.

The correlations $\overline{\vec{v}''\cdot\overleftrightarrow{\tau}}$ and $\overline{\vec{v}''\frac{1}{2}\rho\vec{v}''\cdot\vec{v}''}$ found in the Favre-averaged gas-phase energy equation, equation (2.40), and the turbulent kinetic energy equation, equation (2.47), correspond to the molecular diffusion and turbulent transport of turbulent kinetic energy, respectively. An approximation typically used for these terms is

$$\overline{\vec{v}''\cdot\overleftrightarrow{\tau}} - \overline{\vec{v}''\tfrac{1}{2}\rho\vec{v}''\cdot\vec{v}''} = \left(\mu + \sigma_k\mu_T\right)\vec{\nabla}k, \tag{2.50}$$

where σ_k is a closure coefficient that must be specified. The pressure diffusion, $\vec{\nabla}\cdot(\overline{p'\vec{v}''})$, pressure work, $\overline{\vec{v}''}\cdot\vec{\nabla}\overline{p}$, and pressure dilatation, $\overline{p'\vec{\nabla}\cdot\vec{v}''}$, correlations are typically small and, therefore, are generally ignored [211].

The first term on the right-hand side of the turbulent kinetic energy equation, equation (2.47), corresponds to the production of turbulent kinetic energy,

$$-\overline{\rho\vec{v}''\vec{v}''} : \vec{\nabla}\tilde{\vec{v}} = \overleftrightarrow{\lambda} : \vec{\nabla}\tilde{\vec{v}}. \tag{2.51}$$

The second term defines the dissipation of turbulent kinetic energy. This correlation is given in terms of the specific dissipation rate as

$$\overline{\overleftrightarrow{\tau} : \vec{\nabla}\vec{v}''} = \beta_k \overline{\rho}\omega k, \tag{2.52}$$

where β_k is another closure coefficient.

A transport equation similar to the kinetic energy equation is postulated and used to describe the time evolution of the specific dissipation rate, ω:

$$\frac{\partial}{\partial t}(\overline{\rho}\omega) + \vec{\nabla}\cdot\left(\overline{\rho}\tilde{\vec{v}}\omega\right) = \alpha\frac{\omega}{k}\overleftrightarrow{\lambda} : \vec{\nabla}\tilde{\vec{v}} - \beta_\omega\overline{\rho}\omega^2 + \vec{\nabla}\cdot\Big[\big(\mu + \sigma_\omega\mu_T\big)\vec{\nabla}\omega\Big], \tag{2.53}$$

where the terms on the right-hand side correspond to the production, dissipation, and the molecular diffusion and transport of the specific dissipation rate. Three more closure coefficients are required for this equation: α, β_ω, and σ_ω.

All of the statistical correlations appearing in equations (2.38)–(2.43) and (2.47) have now been specified. The resulting system of equations includes eight total equations: three for the gas-phase, two for the gas-phase turbulence model, and three for the particle-phase. Along with the relations for the turbulent eddy-viscosity, turbulent thermal conductivity, and the equation of state, the Favre-averaged equations governing the dilute and disperse gas-particle flow are

$$\frac{\partial\rho}{\partial t} + \vec{\nabla}\cdot\big(\rho\vec{v}\big) = 0, \tag{2.54}$$

$$\frac{\partial}{\partial t}(\rho\vec{v}) + \vec{\nabla}\cdot\Big[\rho\vec{v}\vec{v} + \big(p + \tfrac{2}{3}\rho k\big)\overleftrightarrow{I}\Big] = \vec{\nabla}\cdot\overleftrightarrow{\tau} - \frac{\sigma_p}{\tau_v}\big(\vec{v}-\vec{u}\big), \tag{2.55}$$

$$\begin{aligned}\frac{\partial E}{\partial t} + \vec{\nabla}\cdot\Big[\vec{v}\big(E + p + \tfrac{2}{3}\rho k\big)\Big] = \vec{\nabla}\cdot\Big[\vec{v}\cdot\overleftrightarrow{\tau} - \vec{q} + \big(\mu + \sigma_k\mu_T\big)\vec{\nabla}k\Big] - \\ \left[\frac{\sigma_p}{\tau_v}\big(\vec{v}-\vec{u}\big)\cdot\vec{u} + \frac{\sigma_p c_p}{\tau_T}\big(T - T_p\big)\right],\end{aligned} \tag{2.56}$$

$$\frac{\partial}{\partial t}(\rho k) + \vec{\nabla}\cdot\big(\rho\vec{v}k\big) = \overleftrightarrow{\lambda} : \vec{\nabla}\vec{v} - \beta_k\rho\omega k + \vec{\nabla}\cdot\Big[\big(\mu + \sigma_k\mu_T\big)\vec{\nabla}k\Big], \tag{2.57}$$

$$\frac{\partial}{\partial t}(\rho\omega) + \vec{\nabla}\cdot\big(\rho\vec{v}\omega\big) = \alpha\frac{\omega}{k}\overleftrightarrow{\lambda} : \vec{\nabla}\vec{v} - \beta_\omega\rho\omega^2 + \vec{\nabla}\cdot\Big[\big(\mu + \sigma_\omega\mu_T\big)\vec{\nabla}\omega\Big], \tag{2.58}$$

$$\frac{\partial\sigma_p}{\partial t} + \vec{\nabla}\cdot\big(\sigma_p\vec{u}\big) = 0, \tag{2.59}$$

$$\frac{\partial}{\partial t}(\sigma_p\vec{u}) + \vec{\nabla}\cdot\big(\sigma_p\vec{u}\vec{u}\big) = \frac{\sigma_p}{\tau_v}\big(\vec{v}-\vec{u}\big), \tag{2.60}$$

$$\frac{\partial\epsilon_p}{\partial t} + \vec{\nabla}\cdot\big(\vec{u}\epsilon_p\big) = \frac{\sigma_p c_p}{\tau_T}\big(T - T_p\big). \tag{2.61}$$

The symbols denoting the averaged quantities have been dropped for the sake of clarity and simplicity. Note that the stress tensor now includes contributions from the laminar and Reynolds stress tensors, $\overleftrightarrow{\tau} = 2(\mu + \mu_T)\big(\vec{\nabla}\vec{v} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\overleftrightarrow{I}\big)$. The kinetic energy term in the Reynolds stress

tensor, $\frac{2}{3}\rho k\vec{\vec{I}}$, has been included with the pressure in the convection term. This term behaves as an additional or "effective" pressure term and can be important when the turbulent kinetic energy is large. Similarly, the heat flux vector includes contributions from the laminar and turbulent heat flux vectors, $\vec{q} = -(\kappa + \kappa_T)\vec{\nabla}T$. Values for the six closure coefficients (Pr_T, σ_k, β_k, α, β_ω, and σ_ω) are discussed next.

2.3.3 Turbulence Model Closure Coefficients

The closure coefficients for the k-ω turbulence model used here are outlined in the textbook by Wilcox [211] and are given by

$$\alpha = 0.52, \quad \sigma_k = 0.5, \quad \sigma_\omega = 0.5, \quad \beta_k = \beta_{k_0} f_k = 0.072 f_k, \quad \text{and} \quad \beta_\omega = \beta_{\omega_0} f_\omega = 0.09 f_\omega. \tag{2.62}$$

The functions f_k and f_ω were designed to improve the accuracy of the model for predicting free shear flows and wake-like flows away from solid boundaries. These functions are given by

$$f_k = \begin{cases} 1, & \chi_k \leq 0 \\ \dfrac{1 + 680\chi_k^2}{1 + 400\chi_k^2}, & \chi_k > 0 \end{cases} \quad \text{where} \quad \chi_k = \frac{1}{\omega^3}\vec{\nabla}k \cdot \vec{\nabla}\omega, \tag{2.63}$$

$$f_\omega = \frac{1 + 70\chi_\omega}{1 + 80\chi_\omega}, \quad \text{where} \quad \chi_\omega = \frac{1}{(\beta_{k_0}\omega)^3}\left| \left(\vec{\vec{\Omega}} \otimes \vec{\vec{\Omega}}\right) : \vec{\vec{S}} \right|, \tag{2.64}$$

where $\vec{\vec{S}}$ and $\vec{\vec{\Omega}}$ are the mean strain rate and mean rotation tensors.

2.3.4 Boundary Conditions

The k and ω transport equations can be integrated through the laminar sublayer right to the solid boundary, eliminating the need of low Reynolds number formulations [101], two-layer models [26, 161], or wall functions [27]. The no-slip condition at solid walls dictates that the turbulent kinetic must go to zero at the wall. Perturbation analysis of the equations in the laminar sublayer shows that

$$\lim_{y_w \to 0} \omega = \frac{6\nu}{\beta_{\omega_o}, y_w^2} \tag{2.65}$$

where y_w is the distance to the solid wall [211]. This equation is used to specify the specific dissipation rate, ω, for all non-dimensional wall distances, y^+, less than 2.5-5.

Standard wall functions can also be used for situations where the mesh refinement does not permit direct integration of the k equation to the wall. In this case, the expressions

$$k = \frac{u_\tau^2}{\sqrt{\beta_{k_0}}}, \quad \omega = \frac{u_\tau}{\sqrt{\beta_{k_0}}\kappa y_w}, \tag{2.66}$$

are used to fully specify k and ω for $y^+ \leq 30 - 250$. The constant, $\kappa = 0.41$, is the von Kármán constant.

Application of adaptive mesh refinement to a coarse mesh may permit switching from using standard wall functions to direct integration where the resolution is available. Automated switching between boundary condition types can be accomplished by employing a smooth blending of the two approaches as described by Gao and Groth [55].

2.3.5 Calculation of y_w and y^+

Application of the boundary conditions for the turbulent kinetic energy and the specific dissipation rate requires the calculation of the dimensionless distance to the wall, y^+. This distance is dependent on the dimensional distance, y_w, the friction velocity at the wall, u_τ, and the viscosity as given by

$$y^+ = \frac{u_\tau y_w}{\nu}. \tag{2.67}$$

The friction velocity and dimensionless wall distance are determined by iterating on the empirical relations that match the boundary layer profile. Refer to Figure 2.4. The dimensionless velocity is defined as $u^+ = u/u_\tau$ where u is the tangential velocity component to the wall. Next to the wall, the flow is dominated by the effects of viscosity and the dimensionless velocity profile varies linearly with dimensionless distance,

$$u^+ = y^+. \tag{2.68}$$

This layer is known as the viscous sublayer. The next layer is the log layer which is governed by the well known *law of the wall* which is given by

$$u^+ = \frac{1}{\kappa} \ln y^+ + C, \tag{2.69}$$

where $\kappa \approx 0.41$ and $C \approx 5$. Beyond the log layer is the defect layer. Application of turbulent boundary conditions only require accurate predictions of y^+ within the sublayer and log layer.

The friction velocity and dimensionless distance are computed simultaneously. First, u_τ is determined based on the viscous sublayer relation, equation (2.68), as given by $u_\tau = \sqrt{\nu u / y_w}$ from which the first guess for the dimensionless distance to the wall can be found from equation (2.67). If this value of y^+ is less than the interface point between the sublayer and the log layer then the calculation of y^+ is complete. Otherwise, the law of the wall is iterated on using Newton's method to find the correct u_τ given u, ν, y_w, κ, and C. The function and its derivative for the iterative

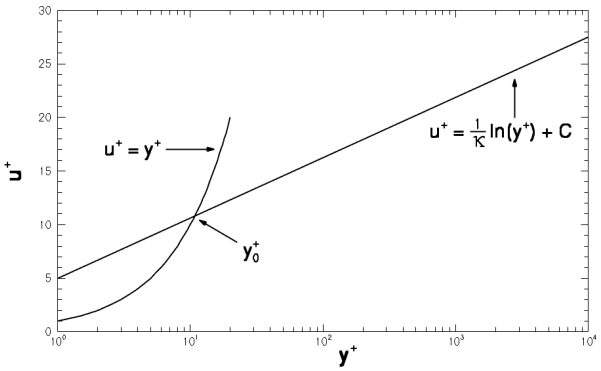


Figure 2.4: *Turbulent boundary layer profile as specified by the viscous sublayer and log-layer relations.*

scheme are given by

$$f(u_\tau) = u_\tau - \frac{\kappa u}{\ln(E u_\tau y_w/\nu)}, \quad \frac{\mathrm{d}f}{\mathrm{d}u_\tau} = 1 + \frac{\kappa u/u_\tau}{\ln(E u_\tau y_w/\nu)^2},$$

where $E = e^{\kappa C}$. The final dimensionless wall distance is computed after a converged value for u_τ is obtained. Note that the y^+ value at the interface between the sublayer and log layer, y_0^+, is computed by iterating on

$$y_0^+ = \frac{1}{\kappa} \ln y_0^+ + C, \tag{2.70}$$

given κ and C. This parameter only has to be calculated once since it depends solely on the constants κ and C.

Note that the dimensionless wall distance must be recomputed during the numerical solution. However, the dimensional wall distance, y_w, only needs to be determined after the mesh has been generated (and after each refinement). The dimensional wall distance, as well as the position, normal vector, and boundary condition type at the nearest wall, are determined by a direct exhaustive search for every cell in the mesh.

2.4 Solid Propellant Combustion Modelling

The combustion of the solid propellant of the rocket motor occurs in a thin, high temperature layer between the solid propellant and the main flow cavity, known as the combustion interface. This layer is assumed to be small relative to the diameter of the rocket motor and large relative to the product of the propellant product velocities and chemical reaction relaxation times such that the finite-rate nature of the chemical reactions can be neglected and the injected gas-particle products

can be assumed to be in chemical and dynamical equilibrium. This topologically complex surface evolves as the propellant burns and injects the hot combustion products into the rocket chamber. In general, the burning rate, r_{bs}, of the propellant is dependent on the local gas pressure, convective gas flow across the grain, and rate of heat transfer through the propellant. However, in this work, the burning rate is assumed to be pressure dependent only and can be determined by the empirical St. Robert relation [100] as given by

$$\rho_{bs} v_{bs} = (1-\alpha_s)\rho_s r_{bs} = (1-\alpha_s)\rho_s \beta\, p_{bs}^n, \tag{2.71}$$

where ρ_{bs}, v_{bs}, and p_{bs} are the density, speed, and pressure of the gas injected from the burning surface and ρ_s is the solid propellant density. The burning rate constants β and n are functions of the chemical composition of the solid and the adiabatic flame temperature, T_f. The mass fraction of solid particles in the propellant is given by α_s. Note that it is assumed here that the particles are injected into the combustion chamber at the same speed as the gas-phase and at the adiabatic flame temperature.

Various models have been proposed to account for erosive [94, 159, 63] and transient [22, 183] burning effects. Erosive burning rates can be of the same order of magnitude as the pressure dependent burning rate, particularly away from the head end of the rocket motor where the combustion products are moving more rapidly and during the initial phase of a rocket firing when the port-to-throat area is small [63]. Incorporation of these effects is beyond the scope of the current research and is left for future consideration. Ignition transients are also neglected in this work.

At the burning surface, the turbulence boundary conditions discussed above must be modified to account for the mass injection. Wilcox proposed a correction to the law of the wall as well as the asymptotic limit of the specific dissipation rate at burning surfaces [211]. However, these were found to only be valid for weak non-dimensional injection rates of order one or less, $v_{bs}^+ = v_{bs}/u_\tau \approx O(1)$. Non-dimensional mass injection rates for rocket motors are typically around 10, $v_{bs}^+ \approx O(10)$. The modifications proposed by Wilcox were found to be unstable at those levels. Stevenson also devised a modified law of the wall for flows with mass injection at the wall [181]. Contrary to Wilcox's work, Stevenson determined that constants κ and C are unaffected by injection rate. However, his work was also valid for only weak mass injection rates. Tseng and Yang [194] and Cai *et al.* [23] modified the eddy-viscosity length scale of the one-equation turbulence model of Wolfshtein [212]. The use of the Wolfshtein model allows for solutions in the near-wall or low Reynolds number regions without resorting to wall function formulations. Tseng and Yang and Cai *et al.* used the one-equation model of Wolfshtein in the near wall region and the standard k-ϵ turbulence model is used away from the wall as is done in a two-layer approach [26, 161]. Sabnis *et al.* [167] and Ciucci

and co-workers [28, 29] devised mass injection modifications to the damping functions of the low Reynolds number k-ϵ formulation [92, 101].

In the present work, the k and ω equations are integrated directly to the burning surface where the kinetic energy and specific dissipation rate are specified by

$$k_{bs} = \sigma_v^2 v_{bs}^2, \quad \omega_{bs} = \sqrt{k_{bs}}/l_{bs} \tag{2.72}$$

where σ_v is a parameter characterizing the surface roughness and l_{bs} is a length scale of the turbulence. These boundary values were used by Cai *et al.* [23] in the two-layer approach described above.

Chapter 3

FINITE-VOLUME SCHEME

In a finite-volume approach, the governing partial differential equations are solved in integral form over a domain discretized into computational cells and, therefore, naturally enforce both locally and globally conservation of the key solution quantities, such as mass, momentum, and energy in computational fluid dynamics applications. A brief introduction into finite-volume schemes is provided here. The textbooks by Hirsch [81], Laney [104], Toro [191], and Lomax *et al.* [115] offer detailed explanations of these schemes.

The weak-conservation or integral form of the two-dimensional axisymmetric multi-phase flow equations of Chapter 2 can be expressed as

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{A(t)}\mathbf{U}\mathrm{d}A+\oint_{\partial A(t)}\left[\vec{\vec{\mathbf{F}}}(\mathbf{U})-\vec{w}\mathbf{U}\right]\cdot\hat{n}\,\mathrm{d}\ell=\oint_{\partial A(t)}\vec{\vec{\mathbf{G}}}(\mathbf{U},\vec{\nabla}\mathbf{U})\cdot\hat{n}\,\mathrm{d}\ell+\int_{A(t)}\mathbf{S}(\mathbf{U})\mathrm{d}A, \tag{3.1}$$

where $\mathbf{U}$ is the vector of conserved solution quantities, A is the area of the finite-volume, and $\mathrm{d}\ell$ and $\hat{n}$ are the length and unit outward normal of the faces of the finite-volume. The flux dyads, $\vec{\vec{\mathbf{F}}}(\mathbf{U})$ and $\vec{\vec{\mathbf{G}}}(\mathbf{U},\vec{\nabla}\mathbf{U})$, represent fluxes of the solution quantities through the boundaries of the finite-volume of hyperbolic and elliptic nature, respectively. The vector $\mathbf{S}$ contains source terms such as the phase-interaction, turbulence production and dissipation, and axisymmetric geometry terms. If the area of the volume of interest varies with time, then the flux contribution due to motion of the cell interfaces must be accounted for, as given by the $\vec{w}\mathbf{U}$ term in the above equation where $\vec{w}$ is the velocity of the cell interface.

Most finite-volume schemes involve three distinct stages: projection, flux evaluation, and time evolution. The projection stage involves the determination of the cell-averaged solution state, as given by

$$\int_{A(t)}\mathbf{U}\mathrm{d}A=\overline{\mathbf{U}}A. \tag{3.2}$$

The use of cell-averaged data results in data that is discontinuous with that of neighbouring cells. Finite-volume schemes, therefore, naturally allow for discontinuities (*e.g.*, shocks) in the flow-field. The second stage of a finite-volume scheme, flux evaluation, involves resolution of the discontinuities that arise from solution reconstruction at the quadrature points on the cell interfaces for the

calculation of the hyperbolic and elliptic fluxes. The final stage of a finite-volume scheme involves the evolution of the solution through the use of a numerical integration scheme.

Applying the projection step represented by equation (3.2) to the integral form of the conservation equations given by equation (3.1) yields the following system of semi-discrete ordinary differential equations:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\overline{\mathbf{U}}A\right) = -\sum_{k}\left[\overleftrightarrow{\mathbf{F}}_k - \vec{w}_k\mathbf{U}_k - \overleftrightarrow{\mathbf{G}}_k\right]\cdot\hat{n}_k\,\Delta\ell_k + \mathbf{S}(\overline{\mathbf{U}})A, \tag{3.3}$$

where the contour integrals have been simplified to summations over the faces of the computational cell, $\Delta\ell$ is the face length, and it has been assumed that $\int_{A(t)} S(\mathbf{U})\mathrm{d}A \approx S(\overline{\mathbf{U}})A$. The time derivative can be expanded to give the following form for the semi-discrete ordinary differential equations:

$$\frac{\mathrm{d}\overline{\mathbf{U}}}{\mathrm{d}t} = -\frac{1}{A}\sum_{k}\left[\overleftrightarrow{\mathbf{F}}_k - \vec{w}_k\mathbf{U}_k - \overleftrightarrow{\mathbf{G}}_k\right]\cdot\hat{n}_k\,\Delta\ell_k + \mathbf{S}(\overline{\mathbf{U}}) - \frac{\overline{\mathbf{U}}}{A}\frac{\mathrm{d}A}{\mathrm{d}t}. \tag{3.4}$$

The last term on the right-hand side of this equation corresponds to the rate of change of the cell area which can be approximated by the geometric conservation law, written in semi-discrete form, as

$$\frac{\mathrm{d}A}{\mathrm{d}t} = \sum_{k}\vec{w}_k\cdot\hat{n}_k\,\Delta\ell_k. \tag{3.5}$$

The geometric conservation law states that the change in cell area is equal to the area swept by the moving surfaces [190].

In order to integrate the preceding set of ordinary differential equations and construct a numerical solution the various terms appearing in equation (3.4) must be carefully evaluated. The remainder of this Chapter provides a description of the numerical methods used for flux evaluation, time evolution, mesh refinement, and parallel implementation. In the next section, the semi-discrete form of the two-dimensional axisymmetric turbulent multi-phase flow equations, equations (2.54) – (2.61), is presented in more detail. The explicit time-stepping schemes for unsteady and steady calculations are also given in this section. Evaluation of the hyperbolic and elliptic fluxes are then described for the gas-phase. Note that frozen flow conditions are assumed for the solution of the Riemann problem for the hyperbolic fluxes. In the frozen flow limit, the phase interaction terms vanish and the gas and particle-phases fully decouple. Therefore, separate Riemann problems and solutions can be formulated for the two phases. Evaluation of the particle-phase flux is discussed in Section 3.5. Specifically, the design and implementation of a multi-velocity formulation for the particle phase is outlined. The proposed approach provides a more consistent representation of the

dynamics of the particle phase discussed in the previous Chapter than that provided by current single-velocity formulations. The next section of this Chapter discusses the block-based adaptive mesh refinement (AMR) scheme that is used to automatically adapt the computational grid to the solution of the governing equations. AMR schemes are very effective in treating problems with disparate length scales while minimizing computational cost. In addition, the block-based data structure lends itself naturally to domain decomposition and thereby enables efficient and scalable implementation of the algorithm on distributed-memory multi-processor architectures. The parallel implementation is described in the last section. Various validation and demonstration numerical test cases are presented throughout the Chapter to show the capabilities of the numerical algorithm.

3.1 Semi-Discrete Form of the Governing Equations

In this work, the domain is discretized into quadrilateral cells which are contained in body-fitted structured blocks of $N_x \times N_y$ cells. Solution data associated with each block are stored in indexed array data structures. The finite-volume scheme of equation (3.4) applied to cell (i, j) of the body-fitted quadrilateral mesh results in the following semi-discrete form of the equations:

$$\left.\frac{\mathrm{d}\overline{\mathbf{U}}}{\mathrm{d}t}\right|_{ij} = -\frac{1}{A_{ij}} \sum_k \left[\overleftrightarrow{\mathbf{F}}_{ij_k} - \vec{w}_k \mathbf{U}_{ij_k} - \overleftrightarrow{\mathbf{G}}_{ij_k}\right] \cdot \hat{n}_{ij_k} \, \Delta\ell_k - \mathbf{S}_{ij} + \mathbf{T}_{ij} + \mathbf{P}_{ij} + \left.\left(\frac{\overline{\mathbf{U}}}{A}\frac{\mathrm{d}A}{\mathrm{d}t}\right)\right|_{ij}, \tag{3.6}$$

where A_{ij} is the area of cell (i, j), $\Delta\ell_k$ is the length of the cell face k, and $\hat{n}_k$ is the unit vector normal to the cell face k. The vector $\overline{\mathbf{U}}$ represents the cell-averaged conserved variable solution vector given by

$$\overline{\mathbf{U}} = \left[\, \rho,\, \rho v_r,\, \rho v_z,\, E,\, \rho k,\, \rho\omega,\, \sigma_p,\, \sigma_p u_r,\, \sigma_p u_z,\, \epsilon_p \,\right]^T, \tag{3.7}$$

where v_r and v_z are the radial and axial components of the gas velocity $\vec{v}$, u_r and u_z are the radial and axial components of the particle velocity $\vec{u}$. The cell-averaged primitive variable solution vector is given by

$$\overline{\mathbf{V}} = \left[\, \rho,\, v_r,\, v_z,\, p,\, k,\, \omega,\, \sigma_p,\, u_r,\, u_z,\, T_p \,\right]^T. \tag{3.8}$$

The dyadic quantities $\overleftrightarrow{\mathbf{F}} = [\mathbf{F}_r, \mathbf{F}_z]$ and $\overleftrightarrow{\mathbf{G}} = [\mathbf{G}_r, \mathbf{G}_z]$ are the hyperbolic and elliptic flux dyads having vector components in the radial and axial directions given by

$$\mathbf{F}_r = \Big[\rho v_r,\, \rho v_r^2 + p + \tfrac{2}{3}\rho k,\, \rho v_r v_z,\, v_r\big(E + p + \tfrac{2}{3}\rho k\big), \\ \rho v_r k,\, \rho v_r \omega,\, \sigma_p u_r,\, \sigma_p u_r^2,\, \sigma_p u_r u_z,\, u_r \epsilon_p\Big]^T, \tag{3.9}$$

$$\mathbf{F}_z = \Big[\rho v_z,\, \rho v_r v_z,\, \rho v_z^2 + p + \tfrac{2}{3}\rho k,\, v_z\big(E + p + \tfrac{2}{3}\rho k\big),$$
$$\rho v_z k,\, \rho v_z \omega,\, \sigma_p u_z,\, \sigma_p u_r u_z,\, \sigma_p u_z^2,\, u_z \epsilon_p\Big]^T, \tag{3.10}$$

$$\mathbf{G}_r = \Big[0,\, \tau_{rr},\, \tau_{rz},\, v_r\tau_{rr} + v_z\tau_{rz} - q_r + (\mu + \sigma_k\mu_T)\frac{\partial k}{\partial r},$$
$$(\mu + \sigma_k\mu_T)\frac{\partial k}{\partial r},\, (\mu + \sigma_\omega\mu_T)\frac{\partial \omega}{\partial r},\, 0,\, 0,\, 0,\, 0\Big]^T, \tag{3.11}$$

$$\mathbf{G}_z = \Big[0,\, \tau_{rz},\, \tau_{zz},\, v_r\tau_{rz} + v_z\tau_{zz} - q_z + (\mu + \sigma_k\mu_T)\frac{\partial k}{\partial z},$$
$$(\mu + \sigma_k\mu_T)\frac{\partial k}{\partial z},\, (\mu + \sigma_\omega\mu_T)\frac{\partial \omega}{\partial z},\, 0,\, 0,\, 0,\, 0\Big]^T. \tag{3.12}$$

The axisymmetric components of the combined laminar and Reynolds stress tensor are

$$\tau_{rr} = 2(\mu + \mu_T)\left(\frac{\partial u}{\partial r} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\right), \quad \tau_{rz} = (\mu + \mu_T)\left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r}\right),$$
$$\tau_{zz} = 2(\mu + \mu_T)\left(\frac{\partial v}{\partial z} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\right), \quad \tau_{\theta\theta} = 2(\mu + \mu_T)\left(\frac{u}{r} - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\right).$$

The combined laminar and turbulent heat flux vector is given by

$$\vec{q} = -(\kappa + \kappa_T)\left(\frac{\partial T}{\partial r}, \frac{\partial T}{\partial z}\right).$$

The quantities $\mathbf{S}$, $\mathbf{T}$, and $\mathbf{P}$ are source vectors. The vector, $\mathbf{S} = \big(\mathbf{S}_{hyp} - \vec{w}\mathbf{U} - \mathbf{S}_{ell}\big)/r$, represents sources due to the axisymmetric flow geometry and can be written as

$$\mathbf{S}_{hyp} = \Big[\rho v_r,\, \rho v_r^2,\, \rho v_r v_z,\, v_r\big(E + p + \tfrac{2}{3}\rho k\big),\, \rho v_r k,\, \rho v_r \omega,\, \sigma_p u_r,\, \sigma_p u_r^2,\, \sigma_p u_r u_z,\, u_r E_p\Big]^T, \tag{3.13}$$

$$\mathbf{S}_{ell} = \Big[0,\, \tau_{rr} - \tau_{\theta\theta},\, \tau_{rz},\, -q_r + v_r\tau_{rr} + v_z\tau_{rz} + (\mu + \sigma_k\mu_T)\frac{\partial k}{\partial r},$$
$$(\mu + \sigma_k\mu_T)\frac{\partial k}{\partial r},\, (\mu + \sigma_\omega\mu_T)\frac{\partial \omega}{\partial r},\, 0,\, 0,\, 0,\, 0\Big]^T. \tag{3.14}$$

The vector $\mathbf{T}$ contains the turbulent source terms and is given by

$$\mathbf{T} = \Big[0,\, 0,\, 0,\, 0,\, \overleftrightarrow{\lambda} : \vec{\nabla}\vec{v} - \beta_k\rho\omega k,\, \alpha\frac{\omega}{k}(\overleftrightarrow{\lambda} : \vec{\nabla}\vec{v}) - \beta_\omega\rho\omega^2,\, 0,\ 0,\ 0,\ 0\Big]^T. \tag{3.15}$$

Turbulence closure is discussed in a subsequent section. Finally, the vector, $\mathbf{P}$, contains the gas-particle interaction source terms and has the form

$$\mathbf{P} = \Big[0,\, -\frac{\sigma_p}{\tau_v}\big(v_r - u_r\big), -\frac{\sigma_p}{\tau_v}\big(v_z - u_z\big), -\frac{\sigma_p}{\tau_v}\big(\vec{v} - \vec{u}\big)\cdot\vec{u} - \frac{\sigma_p c_p}{\tau_T}\big(T - T_p\big),\, 0,\, 0,$$
$$0,\, \frac{\sigma_p}{\tau_v}\big(v_r - u_r\big),\, \frac{\sigma_p}{\tau_v}\big(v_z - u_z\big),\, \frac{\sigma_p c_p}{\tau_T}\big(T - T_p\big)\Big]^T. \tag{3.16}$$

3.2 Explicit Temporal Discretization Methods

The semi-discrete form of the governing two-phase flow equations of equation (3.6) can be rewritten in a more compact form as

$$\frac{\mathrm{d}\overline{\mathbf{U}}}{\mathrm{d}t} = \mathbf{R}(\overline{\mathbf{U}})\,, \tag{3.17}$$

where $\mathbf{R}(\overline{\mathbf{U}})$ is the residual defined by the right-hand side of equation (3.6). For time-accurate calculations, this set of ordinary differential equations is integrated using a predictor-corrector or fourth-order Runge-Kutta time-marching method. MacCormack's predictor-corrector method is used for second-order time-marching and is given by

$$\begin{aligned}
\tilde{\overline{\mathbf{U}}}_{n+1} &= \overline{\mathbf{U}}_n + \triangle t \overline{\mathbf{U}}'_n, \\
\overline{\mathbf{U}}_{n+1} &= \overline{\mathbf{U}}_n + \tfrac{1}{2}\triangle t\big(\overline{\mathbf{U}}'_n + \tilde{\overline{\mathbf{U}}}'_{n+1}\big),
\end{aligned} \tag{3.18}$$

where the integration is from time level n to time level $n{+}1$. The standard fourth-order Runge-Kutta time-marching method is given by

$$\begin{aligned}
\hat{\overline{\mathbf{U}}}_{n+1/2} &= \overline{\mathbf{U}}_n + \tfrac{1}{2}\triangle t \overline{\mathbf{U}}'_n, \\
\tilde{\overline{\mathbf{U}}}_{n+1/2} &= \overline{\mathbf{U}}_n + \tfrac{1}{2}\triangle t \hat{\overline{\mathbf{U}}}'_{n+1/2}, \\
\bar{\overline{\mathbf{U}}}_{n+1} &= \overline{\mathbf{U}}_n + \triangle t \tilde{\overline{\mathbf{U}}}'_{n+1/2}, \\
\overline{\mathbf{U}}_{n+1} &= \overline{\mathbf{U}}_n + \tfrac{1}{6}\triangle t\Big[\overline{\mathbf{U}}'_n + 2\hat{\overline{\mathbf{U}}}'_{n+1/2} + 2\tilde{\overline{\mathbf{U}}}'_{n+1/2} + \bar{\overline{\mathbf{U}}}'_{n+1}\Big].
\end{aligned} \tag{3.19}$$

The textbook by Lomax *et al.* [115] contains a detailed description and analysis of these and many other time-marching methods.

Time-marching methods can also be adopted for steady calculations. Here, the system of equations is marched until the transient part of the solution is removed and

$$\mathbf{R}(\overline{\mathbf{U}}) = 0. \tag{3.20}$$

For steady-state calculations, the explicit optimally-smoothing multi-stage scheme developed by van Leer *et al.* [203] is adopted. This scheme was designed to provide optimal damping of the high frequency content of the solution when using an upwind scheme for the one-dimensional linear convection equation. The M stage time-marching scheme is given by

$$\begin{aligned}
\overline{\mathbf{U}}_0 &= \overline{\mathbf{U}}_n, \\
\overline{\mathbf{U}}_k &= \overline{\mathbf{U}}_0 + \alpha_m \triangle t\, \mathbf{R}(\overline{\mathbf{U}}_{k-1}), \quad \text{for } k = 1 \ldots M, \\
\overline{\mathbf{U}}_{n+1} &= \overline{\mathbf{U}}_M,
\end{aligned} \tag{3.21}$$

Table 3.1: *Multi-stage coefficients for optimal first and second order schemes [203].*

	First Order					Second Order				
M	2	3	4	5	6	2	3	4	5	6
α_1	0.3333	0.1481	0.0833	0.0533	0.0370	0.4242	0.1918	0.1084	0.0695	0.0482
α_2	1	0.4000	0.2069	0.1263	0.0851	1	0.4929	0.2602	0.1602	0.1085
α_3		1	0.4265	0.2375	0.1521		1	0.5052	0.2898	0.1885
α_4			1	0.4414	0.2562			1	0.5060	0.3050
α_5				1	0.4512				1	0.5063
α_6					1					1

where the coefficients for schemes with 2-6 stages are given in Table 3.1. Multi-dimensional optimally smoothing schemes for the Euler and Navier-Stokes equations have been investigated by van Leer and his co-workers [117, 95], but no attempt was made to incorporate these modifications and possible improvements here.

A parallel multigrid method was used for convergence acceleration for steady-state calculations. A description of the multigrid method as used in this work can be found in Ref. [112] and a brief overview is provided in Appendix E. Both V and W multigrid cycles have been implemented and a full-multigrid cycle can be used for start-up purposes. Coarse grid cells are generated by agglomerating four fine grid quadrilateral cells into one cell. The use of the multigrid method with the explicit optimally-smoothing multi-stage temporal discretization scheme discussed above as the smoother has been shown to be very effective at improving the convergence rate of steady-state calculations [116, 112].

An estimate of the maximum allowable time-step is provided by

$$\begin{aligned}\triangle t = \min \Bigg[\beta \, \min \Bigg(& \frac{A}{(|\vec{v}-\vec{w}| + c + |\vec{w}|)\triangle \ell_k}, \frac{2A}{(\Delta \ell)^2 \max(\nu, \nu_T)} \Bigg), \\ & \beta \, \frac{A}{(|\vec{u}-\vec{w}| + |\vec{w}|)\triangle \ell_k}, \beta_p \, \min\left(\tau_T, \tau_v\right) \Bigg],\end{aligned} \tag{3.22}$$

where the index, k, refers to the faces of the cell. The first two conditions correspond to the stability criterion for the hyperbolic and elliptic terms of the gas-phase equations. The third is the stability criterion for the hyperbolic particle-phase term. The *CFL*-number, β, is used to ensure stability of the time-stepping scheme. The final criterion restricts the time-step to be a multiple of the minimum particle-relaxation time-scale. Note that the time-step restrictions associated with the hyperbolic flux account for the motion of the cell interfaces. This restriction is required to ensure

that an entire cell may not be overtaken during a single step. In addition, an augmented sound speed, c, is used to account for the turbulent kinetic energy, $c = \sqrt{a^2 + \frac{2}{3}\gamma k}$ (refer to Section 3.3.2).

3.3 Gas-Phase Hyperbolic Flux Evaluation

In this work, the hyperbolic flux, which includes the gas-phase inviscid flux, the particle-phase flux, and the transport due to the moving boundary, is determined using a high-order Godunov scheme. The methods used for evaluating the gas-phase inviscid flux are discussed in this section. Particle phase flux evaluation is discussed in section 3.5. Upwind finite-volume schemes for the inviscid gasdynamic equations were originally introduced by Godunov in 1959 [59]. The inspiration of Godunov's initial scheme was to introduce physical properties of the governing equations into the numerical discretization to prevent the growth of unwanted oscillations of the numerical solution in the neighbourhood of discontinuities (the monotonicity of the solution is preserved). For hyperbolic equations, such as the Euler equations of compressible gasdynamics, this involved the propagation of solution content along characteristic lines. To accomplish this, a coupled space-time finite-volume discretization procedure was derived in which piece-wise constant solution states are stored in the finite-volumes of the computational domain at each instance in time, permitting the existence of discontinuous solutions. The numerical fluxes at a cell interface can be found by sampling the wave structure resulting from the exact solution of the local Riemann problem (an initial value problem involving two discontinuous constant states) which allows the time evolution of the solution, producing a first-order accurate numerical scheme. In his paper, Godunov showed that to preserve monotonicity an advection scheme with constant coefficients is limited to first-order accuracy – simple application of piece-wise linear reconstruction of the solution states that provide a more accurate interface state for the solution of the Riemann problem can generate oscillations in the neighbourhood of discontinuities through the creation of new extrema. The success of Godunov's initial work led to three main areas of research to improve the algorithm: extension to high-resolution schemes (*i.e.*, higher than first-order accurate schemes) and more efficient solutions to the Riemann problem, as well as the extension to multi-dimensional flows. A brief description of higher-order schemes and approximate Riemann solvers follows. Refer to the textbooks by Hirsch [81], Laney [104], and Toro [191] for a more comprehensive review of these schemes.

3.3.1 Least Squares Gradient Reconstruction

In Godunov's initial scheme, the piecewise constant cell-centred data of the neighbouring cells were chosen as the left and right states for the solution of the Riemann problem at the interface between

the cells. For a one-dimensional problem, the left and right states for interface $i+1/2$ are

$$\begin{aligned} \mathbf{U}_L &= \overline{\mathbf{U}}_i, \\ \mathbf{U}_R &= \overline{\mathbf{U}}_{i+1}. \end{aligned} \tag{3.23}$$

This scheme results in a first-order accurate spatial discretization. However, first-order accurate solutions are impractical for engineering purposes and led researchers to determine methods for permitting more accurate solutions. Through a series of papers from 1973–1979, van Leer introduced a second-order scheme that preserves monotonicity when coupled with higher-order reconstruction [198, 199, 200, 201, 202]. This method, known as Monotone Upstream Scheme for Conservation Laws or MUSCL, makes use of slope limiters (see the papers by Sweby [189] and Munz [125]) to constrain the state reconstruction in the vicinity of discontinuities such that the scheme reduces to first order-accuracy and, therefore, is monotonicity preserving. Here, the left and right states for interface $i+\frac{1}{2}$ are

$$\begin{aligned} \mathbf{U}_L &= \overline{\mathbf{U}}_i + \phi_i \left.\frac{\partial \mathbf{U}}{\partial x}\right|_i (x_{i+\frac{1}{2}} - x_i), \\ \mathbf{U}_R &= \overline{\mathbf{U}}_{i+1} + \phi_{i+1} \left.\frac{\partial \mathbf{U}}{\partial x}\right|_{i+1} (x_{i+\frac{1}{2}} - x_{i+1}) \end{aligned} \tag{3.24}$$

where ϕ is the slope limiter which is calculated for each variable of every cell, (i, j). A centred-difference is typically used as an estimate for the gradient of the state variables. The MUSCL scheme was extended to allow for piecewise parabolic reconstruction via the Piecewise Parabolic Method (PPM) of Colella and Woodward [33]. An observation by Lax that the total variation of a solution to a scalar conservation law does not increase in time [111] led to the development of similar methods known as Total Variation Diminishing (TVD) high resolution schemes [73, 74, 163]. Another high-resolution method known as Essentially Non-Oscillatory (ENO) schemes [75, 176, 177] is based on the selection of a stencil based on the smoothest interpolants and represents a generalization of the MUSCL and PPM schemes. Of these high resolution schemes, only the ENO schemes truly applies to multi-dimensions, although they can be difficult and/or computationally expensive to implement.

MUSCL type schemes were extended to multi-dimensions by Barth and his co-workers [11, 10, 9] by using a k-exact least-squares error minimization technique to construct solution gradients of order $k+1$. In two-dimensions, linear ($k = 1$) reconstruction of the left and right states for interface $(i+\frac{1}{2}, j)$, for example, are given by

$$\begin{aligned} \mathbf{U}_L &= \mathbf{U}_{ij} + \phi_{ij} \left[\left.\frac{\partial \mathbf{U}}{\partial x}\right|_{ij} (x_{i+\frac{1}{2},j} - x_{ij}) + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_{ij} (y_{i+\frac{1}{2},j} - y_{ij}) \right], \\ \mathbf{U}_R &= \mathbf{U}_{i+1,j} + \phi_{i+1,j} \left[\left.\frac{\partial \mathbf{U}}{\partial x}\right|_{i+1,j} (x_{i+\frac{1}{2},j} - x_{i+1,j}) + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_{i+1,j} (y_{i+\frac{1}{2},j} - y_{i+1,j}) \right]. \end{aligned} \tag{3.25}$$

In this work, the reconstruction procedure has been implemented in terms of the primitive solution variables, $\mathbf{V}$, in order to provide better control of solution monotonicity and positivity. In this case, the reconstruction is given by

$$\begin{aligned}\mathbf{V}_L &= \mathbf{V}_{ij} + \phi_{ij}\left[\left.\frac{\partial \mathbf{V}}{\partial x}\right|_{ij}(x_{i+\frac{1}{2},j} - x_{ij}) + \left.\frac{\partial \mathbf{V}}{\partial y}\right|_{ij}(y_{i+\frac{1}{2},j} - y_{ij})\right],\\ \mathbf{V}_R &= \mathbf{V}_{i+1,j} + \phi_{i+1,j}\left[\left.\frac{\partial \mathbf{V}}{\partial x}\right|_{i+1,j}(x_{i+\frac{1}{2},j} - x_{i+1,j}) + \left.\frac{\partial \mathbf{V}}{\partial y}\right|_{i+1,j}(y_{i+\frac{1}{2},j} - y_{i+1,j})\right].\end{aligned} \tag{3.26}$$

For linear least-squares reconstruction, the gradients can be found from the following system of equations

$$\begin{bmatrix} \sum_{pq}(\widehat{x}_{pq})^2 & \sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq} \\ \sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq} & \sum_{pq}(\widehat{y}_{pq})^2 \end{bmatrix} \begin{bmatrix} \left.\frac{\partial \mathbf{V}}{\partial x}\right|_{ij} \\ \left.\frac{\partial \mathbf{V}}{\partial y}\right|_{ij} \end{bmatrix} = \begin{bmatrix} \sum_{pq}\widehat{x}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij}) \\ \sum_{pq}\widehat{y}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij}) \end{bmatrix}.$$

where the reconstruction stencil is given by $p \in [i-1, i+1]$ and $q \in [j-1, j+1]$. Note that at boundary cells the reconstruction stencil can be reduced (one-sided). However, in this work the full reconstruction stencil is used where ghost cells are used and contain boundary condition information. The solution of this system of equations is given by

$$\left.\frac{\partial \mathbf{V}}{\partial x}\right|_{ij} = \frac{\sum_{pq}\widehat{x}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij})\sum_{pq}(\widehat{y}_{pq})^2 - \sum_{pq}\widehat{y}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij})\sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq}}{\sum_{pq}(\widehat{x}_{pq})^2\sum_{pq}(\widehat{y}_{pq})^2 - \left(\sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq}\right)^2}, \tag{3.27}$$

$$\left.\frac{\partial \mathbf{V}}{\partial y}\right|_{ij} = \frac{\sum_{pq}\widehat{y}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij})\sum_{pq}(\widehat{x}_{pq})^2 - \sum_{pq}\widehat{x}_{pq}(\overline{\mathbf{V}}_{pq} - \overline{\mathbf{V}}_{ij})\sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq}}{\sum_{pq}(\widehat{x}_{pq})^2\sum_{pq}(\widehat{y}_{pq})^2 - \left(\sum_{pq}\widehat{x}_{pq}\widehat{y}_{pq}\right)^2}, \tag{3.28}$$

where the geometric coefficients for linear reconstruction are given by $\widehat{x}_{pq} = x_{pq} - x_{ij}$ and $\widehat{y}_{pq} = y_{pq} - y_{ij}$.

Barth and Jespersen [11] developed a multi-dimensional slope limiter to maintain the monotonicity of the numerical scheme. The slope limiter at cell (i, j) is determined for each of the primitive state variables, v_n, by computing

$$\phi = \begin{cases} \min\left(1, \dfrac{v_{max} - v_n}{v_k - v_n}\right), & \text{if } v_k - v_n > 0, \\ \min\left(1, \dfrac{v_{min} - v_n}{v_k - v_n}\right), & \text{if } v_k - v_n < 0, \\ 1, & \text{if } v_k - v_n = 0, \end{cases} \tag{3.29}$$

at each face of the computational cell and selecting the minimum of those values, where v_k is the reconstructed variable on face f and v_{max} and v_{min} are chosen by

$$v_{max} = \max(v_n, v_{neighbours}), \tag{3.30}$$

$$v_{min} = \min(v_n, v_{neighbours}), \tag{3.31}$$

where $v_{neighbours}$ are the primitive variables of the neighbouring cells. The discontinuous nature of this limiter may stall steady-state convergence of the scheme by incorrectly limiting the solution in smooth areas of the flow due to random noise. This effect, known as limiter cycling, can be reduced by freezing the limiter after the residual has dropped to a predefined level. Venkatakrishnan modified this limiter to improve the convergence characteristics for steady-state calculations [205]. The limiter is computed in a similar fashion except that equation (3.29) is replaced with

$$\phi = \begin{cases} \Phi\left(\dfrac{v_{max} - v_n}{v_k - v_n}\right), & \text{if } v_k - v_n > 0, \\ \Phi\left(\dfrac{v_{min} - v_n}{v_k - v_n}\right), & \text{if } v_k - v_n < 0, \\ 1, & \text{if } v_k - v_n = 0, \end{cases} \tag{3.32}$$

where Φ is a smooth function given by

$$\Phi(y) = \frac{y^2 + 2y}{y^2 + y + 2}. \tag{3.33}$$

However, the use of this limiter does not guarantee the elimination of limiter cycling. Limiter cycling can be alleviated by modifying this limiter so as to make it not strictly monotone, allowing for oscillations below a specified threshold to exist [205]. However, this modification is not used here.

3.3.2 Approximate Riemann Solutions

Given the left and right solution state vectors, $\mathbf{U}_l$ and $\mathbf{U}_r$, and velocity $\vec{w}$ of the cell interface, the numerical flux is determined by

$$(\vec{\vec{\mathbf{F}}} - \vec{w}\mathbf{U}) \cdot \hat{n} = \mathcal{F}(\mathbf{U}_l, \mathbf{U}_r, \vec{w}, \hat{n}), \tag{3.34}$$

where $\mathcal{F}$ is evaluated by solving a Riemann problem in a direction defined by the normal to the face, $\hat{n}$. The exact solution to the Riemann problem for the Euler equations requires an iterative procedure and, therefore, was thought to be computationally expensive since the Riemann problem must be solved at each cell interface at every time integration step. This led researchers to develop approximate, non-iterative, solutions to the Riemann problem that give a sufficiently accurate

representation of the wave structure. Various approximate Riemann solvers have been proposed. Roe's approximate Riemann Solver is based on a local linearization of the flow equations from which the numerical flux can be constructed from the resulting eigenstructure [162, 164]. Osher developed an approximate Riemann solver by replacing the shock waves with compression waves and then integrating over the wave structure in phase-space [141, 137]. This scheme produces a smooth (differentiable) numerical flux. These methods provide an accurate representation of the wave structure. Harten, Lax, and Van Leer simplified the wave structure to two discontinuous waves allowing for simple construction of the numerical flux based on two jump conditions and has been termed the HLL class of approximate Riemann solvers [76]. Einfeldt improved the choice of wave-speed estimates for the HLL Riemann solver by including the wave-speeds produced by Roe's local linearization and is commonly referred to as the HLLE Riemann solver [46]. The HLL style of approximate Riemann solver has been extended to include a contact wave for the Euler equations by Toro *et al.*'s HLLC Riemann solver [192] and more recently for an arbitrary hyperbolic system by Linde's HLLL Riemann solver [113]. It has also been shown, however, by Gottlieb and Groth that the efficiency of the exact solution to the Riemann problem can be significantly improved by using an initial estimate based on the isentropic solution and by a judicious choice of the iteration variable [60]. In fact, for the Euler equations, the computational costs of the exact solver can be similar to those of the approximate solvers. The main advantages of the approximate approaches would then be their more ready generalization to other systems of equations.

Here, the gas-phase Riemann solutions must account for the flux contribution by the turbulent kinetic energy as well as moving boundary transport terms. Though the turbulent kinetic energy term results from the Reynolds stress tensor, it is a convective term and must solved in an upwind manner to avoid unstable solutions [204, 178, 57]. The extension of Roe and HLLE approximate Riemann solvers for the present gas-phase system of equations are presented next.

Roe's Approximate Riemann Solver

Roe's approximate Riemann solver is based on a local linearization of the flow equations. The linearization of the Riemann problem approximates all waves by discontinuous jumps. The linearized form of the one-dimensional compressible Euler equations is given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = \frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial x} \approx \frac{\partial \mathbf{U}}{\partial t} + \hat{\mathbf{A}}(\hat{\mathbf{U}})\frac{\partial \mathbf{U}}{\partial x} = 0,$$

where $\hat{\mathbf{A}}(\hat{\mathbf{U}})$ is the "linearized" flux Jacobian computed at a reference state, $\hat{\mathbf{U}}$, which is deemed to be a function of the left and right solution states of the Riemann problem, $\mathbf{U}_L$ and $\mathbf{U}_R$. Since the approximate flux Jacobian contains constant coefficients, this represents a linearized hyperbolic

system of equations for which an analytic solution is available for the intercell flux,

$$\mathbf{F}_{i+\frac{1}{2}} = \frac{1}{2}(\mathbf{F}_L + \mathbf{F}_R) - \frac{1}{2}\sum_{i=1}^{m} \hat{\alpha}_i |\hat{\lambda}_i| \hat{\mathbf{K}}^{(i)}, \tag{3.35}$$

where m is the number of conservation laws in the system of equations under consideration ($m = 4$ for the two-dimensional Euler equations). The goal now is to determine an appropriate reference state, $\hat{\mathbf{U}}$, from which the eigenvalues, $\hat{\lambda}$, eigenvectors, $\hat{\mathbf{K}}$, and wave-strengths, $\hat{\alpha}$, of the approximate flux Jacobian, $\hat{\mathbf{A}}(\hat{\mathbf{U}})$, can be constructed. This reference state was designed such that the approximate flux Jacobian, $\hat{\mathbf{A}}$, satisfies three conditions: (A) it must be strictly hyperbolic, (B) it must be consistent with the exact flux Jacobian, $\hat{\mathbf{A}}\big(\hat{\mathbf{U}}(\mathbf{U},\mathbf{U})\big) = \mathbf{A}(\mathbf{U})$, and (C) it must ensure conservation across discontinuities, $\mathbf{F}(\mathbf{U}_R) - \mathbf{F}(\mathbf{U}_L) = \tilde{\mathbf{A}}(\mathbf{U}_R - \mathbf{U}_L)$ [162]. The appropriate reference state for the gas-phase equations presented in Section 3.1 is determined from the following relations,

$$\begin{gathered} \hat{\rho} = \sqrt{\rho_L \rho_R}, \qquad \hat{v}_x = \frac{\rho_L v_{x_L} + \rho_R v_{x_R}}{\rho_L + \rho_R}, \quad \hat{v}_y = \frac{\rho_L v_{y_L} + \rho_R v_{y_R}}{\rho_L + \rho_R}, \\ \hat{h} = \frac{\rho_L h_L + \rho_R h_R}{\rho_L + \rho_R}, \quad \hat{k} = \frac{\rho_L k_L + \rho_R k_R}{\rho_L + \rho_R}, \quad \hat{\omega} = \frac{\rho_L \omega_L + \rho_R \omega_R}{\rho_L + \rho_R}, \end{gathered} \tag{3.36}$$

where the reference sound-speed and pressure are calculated from $\hat{a} = (\gamma - 1)(\hat{h} - \frac{1}{2}\hat{\vec{v}} \cdot \hat{\vec{v}} - \hat{k})$ and $\hat{p} = \gamma \hat{a}/\hat{\rho}$. There are six eigenvalues for the turbulent gas-phase equations. They are

$$\hat{\lambda}_1 = \hat{v}_x - w_x - \hat{c}, \quad \hat{\lambda}_{2,3} = \hat{v}_x - w_x, \quad \hat{\lambda}_4 = \hat{v}_x - w_x + \hat{c}, \quad \hat{\lambda}_{5,6} = \hat{v}_x - w_x, \tag{3.37}$$

for the x-component flux Jacobian where c is a modified sound speed given by $c = \sqrt{a^2 + \frac{2}{3}\gamma k}$. The corresponding conserved variable right eigenvectors are

$$\hat{\mathbf{K}} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ \hat{v}_x - \hat{c} & \hat{v}_x & 0 & \hat{v}_x + \hat{c} & 0 & 0 \\ \hat{v}_y & \hat{v}_y & \hat{\rho} & \hat{v}_y & 0 & 0 \\ \hat{h} - \hat{c}\hat{v}_x & \frac{1}{2}(\hat{v}_x^2 + \hat{v}_y^2) & \hat{\rho}\hat{v}_y & \hat{h} + \hat{c}\hat{v}_x & \frac{3\gamma-5}{3(\gamma-1)} & 0 \\ \hat{k} & 0 & 0 & \hat{k} & 1 & 0 \\ \hat{\omega} & 0 & 0 & \hat{\omega} & 0 & 1 \end{bmatrix}. \tag{3.38}$$

The wave-strengths can be determined from the solution jump condition

$$\triangle \mathbf{U} = \mathbf{U}_R - \mathbf{U}_L = \sum_{i=1}^{m} \hat{\alpha}_i \hat{\mathbf{K}}^{(i)}. \tag{3.39}$$

and are given by

$$\begin{aligned} \hat{\alpha} = \Bigg[& \frac{1}{2\hat{c}^2}\left(\frac{2}{3}\hat{k}\mathrm{d}\rho + \hat{\rho}\hat{c}\mathrm{d}v_x + \mathrm{d}p + \frac{2}{3}\hat{\rho}\mathrm{d}k\right), \ \left(1 - \frac{2}{3}\frac{\hat{k}}{\hat{c}^2}\right)\mathrm{d}\rho - \frac{\mathrm{d}p}{\hat{c}^2} - \frac{2}{3}\frac{\hat{\rho}}{\hat{c}^2}\mathrm{d}k, \ \mathrm{d}v_y, \\ & \frac{1}{2\hat{c}^2}\left(\frac{2}{3}\hat{k}\mathrm{d}\rho - \hat{\rho}\hat{c}\mathrm{d}v_x + \mathrm{d}p + \frac{2}{3}\hat{\rho}\mathrm{d}k\right), \ \mathrm{d}k, \ \mathrm{d}\omega \Bigg]. \end{aligned} \tag{3.40}$$

This approximate solution to the Riemann problem involves only discontinuous waves. Moreover, because it satisfies condition (C) above, the approximation is valid and exact for isolated shock and contact waves. However, it can be problematic for rarefaction waves where an unphysical discontinuity may form at sonic points. These expansion shocks violate the entropy condition for an physically admissible discontinuity, which dictates that the entropy must increase across a shock [191]. The solution of the linearized Riemann problem for transonic rarefaction fans, where one of the acoustic wave speeds, $\lambda_{1,4} = v_x \pm c$, is zero, exhibits this unphysical solution. To remedy this effect, a correction proposed by Harten [73], which ensures that the acoustic eigenvalue does not go to zero, can be used. The corrected magnitude of the wavespeed is given by

$$|\lambda_k^*| = \begin{cases} |\hat{\lambda}_k|, & |\hat{\lambda}_k| \geq 2(\triangle\lambda_k)^+ \\ \dfrac{\hat{\lambda}_k^2}{4(\triangle\lambda_k)^+} + (\triangle|\lambda_k)^+, & |\hat{\lambda}_k| < 2(\triangle\lambda_k)^+ \end{cases}, \quad \text{for } k = 1, 4 \tag{3.41}$$

where $\triangle\lambda_k = \lambda_k(\mathbf{U}_R) - \lambda_l(\mathbf{U}_L)$ and $(\triangle\lambda_k)^+ = \max(0, \triangle\lambda_k)$. The convective eigenvalues are not altered, $|\lambda_k^*| = |\lambda_k|$ for $k = 2, 3, 5$, and 6. The intercell flux determined from the Roe's approximate Riemann solution is now given by

$$\mathbf{F}_{i+\frac{1}{2}} = \frac{1}{2}(\mathbf{F}_L + \mathbf{F}_R) - \frac{1}{2}\sum_{i=1}^{m} \hat{\alpha}_i |\hat{\lambda}_i^*| \hat{\mathbf{K}}^{(i)}. \tag{3.42}$$

Harten-Lax-van Leer Riemann Solver

The HLL approximate solution to the Riemann problem is based on the simplification of the wave structure to the pattern shown in Figure 3.1. The left and right waves are the only non-linear waves present – the contact surface is ignored and rarefaction waves are replaced by discontinuities. Estimates of the left and right wave-speeds are determined from

$$S_R = \max\Big[\max(\lambda_{R_k}), \max(\hat{\lambda}_k)\Big] \quad \text{and} \quad S_L = \min\Big[\min(\lambda_{L_k}), \min(\hat{\lambda}_k)\Big], \tag{3.43}$$

where λ_{R_k}, λ_{L_k}, and $\hat{\lambda}$ are the wave-speeds of the left, right, and Roe-averaged states. Including the wave-speeds of the Roe-averaged state was suggested by Einfeldt [46] and, subsequently, the approximate Riemann solution is known as the HLLE Riemann solver.

The final flux calculated from the Riemann problem depends on the signs of the signal velocities. If the left signal velocity is greater than zero, $S_L > 0$, then the flux is calculated based on the left initial state, $\mathbf{U}_L$. If the right signal velocity is less than zero, $S_R < 0$, then the flux is calculated based on the right initial state, $\mathbf{U}_R$. Otherwise, the intermediate flux must be determined. Two equations involving the intermediate state and flux are determined by integrating

$$\oint \Big[\mathbf{U}\mathrm{d}x - \mathbf{F}(\mathbf{U})\mathrm{d}t\Big] = 0, \tag{3.44}$$

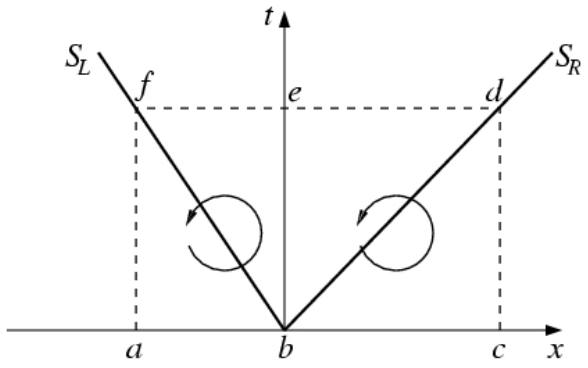


Figure 3.1: *Simplified wave pattern for the HLLE approximate Riemann solver.*

over the two paths specified in Figure 3.1. These equations can be solved to determine an equation for the intermediate flux given by

$$\mathbf{F}^* = \frac{S_R\mathbf{F}_L - S_L\mathbf{F}_R + S_L S_R(\mathbf{U}_R - \mathbf{U}_L)}{S_R - S_L}. \tag{3.45}$$

The relative wave-speeds and fluxes, $\mathbf{F} - \vec{w}\mathbf{U}$, are used in the case of a non-stationary cell interface.

3.3.3 Ringleb's Flow

In order to assess and demonstrate the accuracy of the inviscid spatial discretization scheme used here, the predictions of the numerical algorithm are considered for a test problem for which an exact analytic solution exists. Ringleb's flow is a hodograph solution to the Euler equations that is widely used in verification studies (*e.g.*, [10], [31], [77]). The flow pattern, shown in Figure 3.2(a), involves an isentropic, irrotational flow contained between two streamlines. The analytic solution can be parameterized in terms of a non-dimensional velocity, q, and the streamline number, $k = 1/\psi$:

$$x(q,k) = \frac{1}{2\rho}\left(\frac{2}{k^2} - \frac{1}{q^2}\right) - \frac{J}{2}, \tag{3.46}$$

$$y(q,k) = \pm\frac{1}{\rho k q}\sqrt{1-(q/k)^2}, \tag{3.47}$$

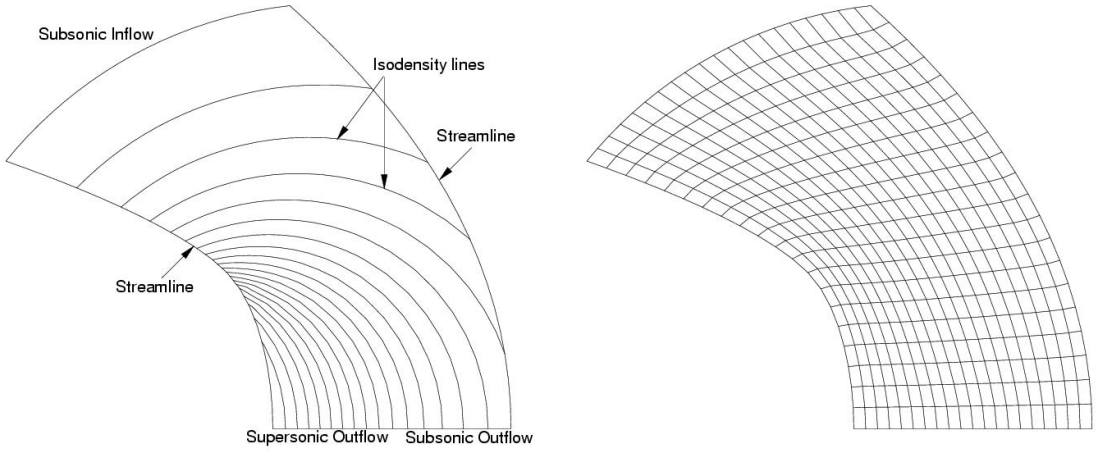
$$\overline{a} = \sqrt{1 - \frac{\gamma-1}{2}q^2}, \tag{3.48}$$

$$\overline{\rho} = \overline{a}^{2/(\gamma-1)}, \tag{3.49}$$

$$J = \frac{1}{\overline{a}} + \frac{1}{3\overline{a}^3} + \frac{1}{5\overline{a}^5} - \frac{1}{2}\ln\left(\frac{1+\overline{a}}{1-\overline{a}}\right), \tag{3.50}$$

where the density, $\overline{\rho}$, is made non-dimensional by the stagnation density and the velocity, q, and sound-speed, $\overline{a}$, are made non-dimensional by the stagnation sound-speed. The flow angle, θ, can be determined from

$$\theta = 2\pi - \sin^{-1}(q/k). \tag{3.51}$$



(a) (b)

Figure 3.2: *Ringleb's flow: (a) Flow pattern and (b) Sample mesh with 400 cells.*

The flow pattern shown in Figure 3.2(a) is defined by the streamlines corresponding to $k = 0.75$ and $k = 1.5$, and the inflow boundary is defined by the iso-tach contour, $q = 0.3$. A mixed supersonic and subsonic outflow occurs at the lower boundary.

The accuracy of the spatial discretization was assessed by comparing the computed solution on a series of meshes involving 100, 400, 1600, and 6400 cells to the analytic solution. The L_1- and L_2-norms of the difference in the solution densities were used as the measure of solution accuracy and were taken to have the form

$$L_p = \left(\frac{\sum |\rho_{ij} - \rho_e(x_{ij}, y_{ij})| A_{ij}}{\sum A_{ij}} \right)^{1/p}, \tag{3.52}$$

where A_{ij} and ρ_{ij} are the area and average density of cell (i, j). The cell-average exact density, ρ_e, is computed by performing a higher-than-second order numerical integration. The exact density at any point (x, y) can be found by iteratively solving the equation for the constant velocity lines,

$$\left(x + \frac{J}{2} \right)^2 + y^2 = \frac{1}{4\rho^2 q^4}, \tag{3.53}$$

for the non-dimensional sound-speed, $\overline{a}$, from which the non-dimensional density is computed by equation (3.49). A sample mesh with 400 cells is shown in Figure 3.2(b). The exact solution is imposed at the lateral, inflow, and outflow boundaries. The left and right states for the evaluation of the Riemann problem on the inflow and outflow boundaries are determined from a characteristic boundary condition with a specified static pressure based on the exact solution at the Gauss point. The proposed algorithm performs well for this test case. The L_1- and L_2-norms of the solution

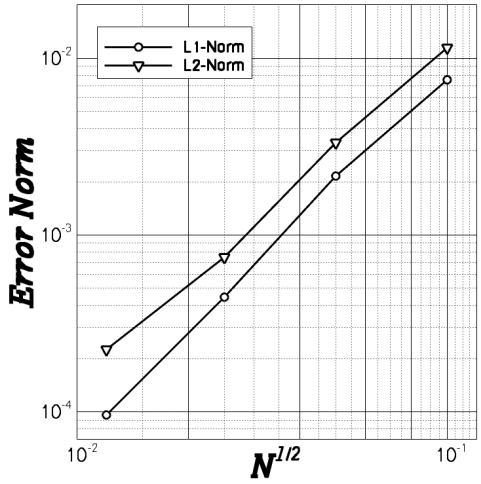


Figure 3.3: *L_1- and L_2-norms of the solution error for Ringleb's flow as a function of the number of computational cells, N.*

error are plotted in Figure 3.3. The slopes of the L_1- and L_2-norms are 2.11 and 1.94, respectively, indicating the scheme is indeed second order-accurate.

3.4 Gas-Phase Elliptic Flux Evaluation

The calculation of the viscous flux of the compressible Navier-Stokes equations requires the knowledge of the solution state as well as the gradient of the solution at cell interfaces. The numerical flux is given by

$$\vec{\mathbf{G}} \cdot \hat{n} = \mathcal{G}(\mathbf{U}, \vec{\nabla}\mathbf{U}, \hat{n}), \tag{3.54}$$

where the dependence on the gradient of the solution quantities dictates that the flux is elliptic by nature. Coirier assessed the accuracy and positivity of several formulations for evaluating the cell-interface gradient of the solution variable [30]. This assessment was conducted by analyzing the stencil of the Laplacian determined by

$$\nabla^2\phi = \frac{1}{A}\sum_k \vec{\nabla}\phi_k \cdot \hat{n}_k \Delta\ell_k, \tag{3.55}$$

where ϕ is the solution variable under consideration and $\vec{\nabla}\phi_k$ is the gradient of the solution quantity at face k. This is a valid model equation since for incompressible flow the viscous stress tensor reduces to the Laplacian operator of the velocity field. Although this is not true for compressible flows, valid reconstruction of the Laplacian is considered a minimum requirement for elliptic flux

evaluation. The Laplacian operator was assessed in terms of accuracy (by comparison with a local Taylor series expansion about the cell centroid) and positivity (does the stencil of the Laplacian satisfy a discrete maximum principle). Coirier determined that there is an incompatibility between these criteria – an accurate scheme is not positive and vice-versa. The most positive (though not strictly) gradient evaluation resulted from performing a Green-Gauss integration over the diamond-path constructed from the neighbouring cell-centres and the nodes of the cell-interface. The most accurate approach involved reconstructing a quadratic polynomial from a suitable set of support cells. This study was performed on a uniform Cartesian grid, a uni-directionally stretched Cartesian grid, and a refined Cartesian grid. Haselbacher and Blazek analyzed the positivity of the stencil created by averaging the cell-centred gradients [78]. This method resulted in a decoupled stencil – the stencil involved distance-2 neighbour cells but not distance-1 neighbour cells (the cells with non-zero entries do not share a vertex with cell (i, j)) on both quadrilateral and hexahedral grids. The authors noted that this decoupling can lead to the growth of spurious solution modes and large truncation errors [78]. They also performed this analysis using a correction to the average gradient approach proposed by Mathur and Murthy [119]. Use of this method results in a strongly coupled stencil. Note that this correction was presented by Mathur and Murthy as a method for improving the diamond-path approach. Their motivation was to eliminate the required solution information at the nodes of the cell-interface.

In this work, the Green-Gauss integration over the diamond-path is used to evaluate the required gradients at cell interfaces. This method is described next and the elliptic flux evaluation is validated by comparing the predicted solution of a laminar flat plate boundary layer with Blasius' analytical solution.

3.4.1 Green-Gauss Integration over the Diamond-Path

The gradient of the primitive variables at a cell interface can be found by the application of a Green-Gauss integral to the four-sided closed path indicated in Figure 3.4(a). The four points of the path correspond to the two cell centres and the nodes of the interface separating the two cells. The solution data is readily available at the cell-centres of the cells. However, the solution at the nodes of the interface must be determined. Coirier used two methods to determine solution data at the nodes: simple vertex weighting and a linearity-preserving weighting [30]. Simple vertex weighting is accomplished by taking an average of the data of the adjoining cells. The linearity-preserving weighting scheme of Holmes and Connell ensures that the node solution data is linearly constructed from the solution data stored at the centroid of all neighbouring cells [83]. The linearity-preserving

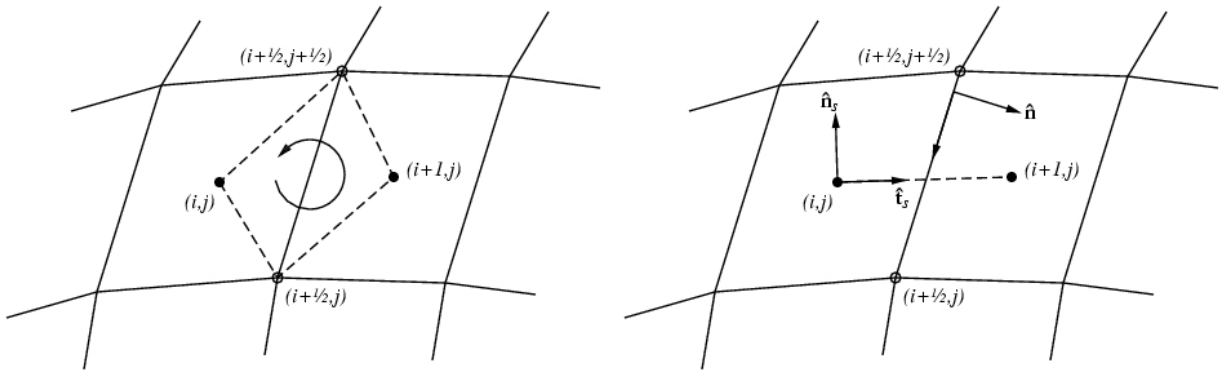


Figure 3.4: *Definition of (a) the diamond-path used for the gradient reconstruction of the east-face elliptic (viscous) flux and (b) the local coordinate systems.*

weighting scheme is used in this work. The gradient is determined by evaluating

$$\vec{\nabla}\phi = \frac{1}{A}\sum_k \phi_k \hat{n}_k \, \Delta\ell_k, \tag{3.56}$$

where A is the area of the quadrilateral specified by the contour path, $\Delta\ell_k$ is the length of face k, $\hat{n}_k$ is the unit outward normal of face k, and ϕ_k is an approximation of the solution variable at the centre of the face.

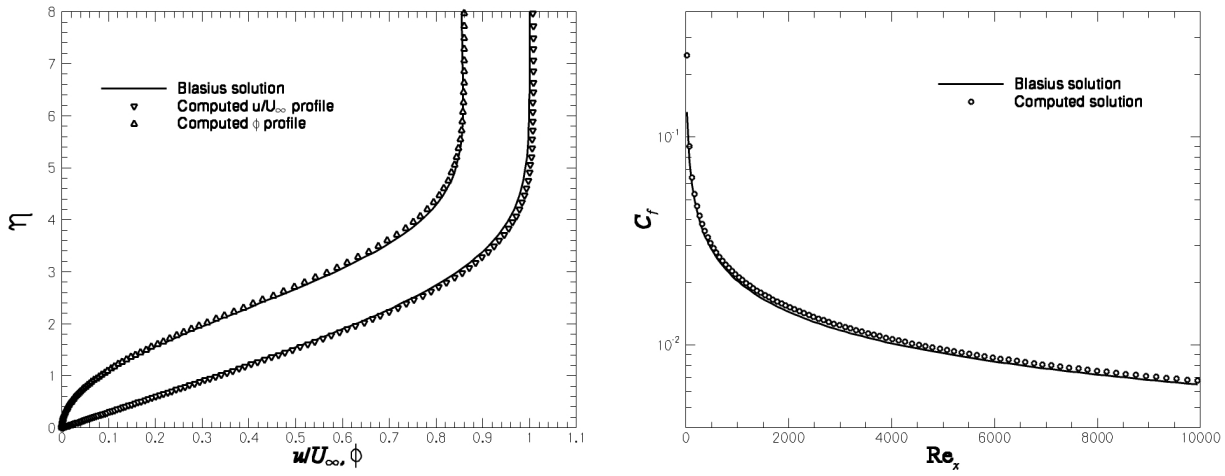
After some manipulation and simplification it can be shown that the Green-Gauss integration over the diamond-path, equation (3.56), reduces to

$$\vec{\nabla}\phi = \frac{1}{\hat{n}\cdot\hat{t}_s}\left(\frac{\phi_{i+1,j}-\phi_{ij}}{\Delta s}\hat{n} + \frac{\phi_{i+\frac{1}{2},j+\frac{1}{2}}-\phi_{i+\frac{1}{2},j-\frac{1}{2}}}{\Delta\ell}\hat{n}_s\right), \tag{3.57}$$

for the east cell-interface where Δs is the distance between the neighbouring cell centroids, $\Delta\ell$ is the length of the face, and the local coordinate systems are defined in Figure 3.4(b). Note that for a Cartesian mesh the interface unit normal and the centroid tangent are equal, therefore, their dot product is unity, $\hat{n}\cdot\hat{t}_s = 1$ and the expression reduces to those of standard second-order central finite-differencing schemes on a uniform mesh.

3.4.2 Laminar Flat Plate Boundary Layer

The computation of laminar flow over a flat plate at zero incidence is now considered to demonstrate the accuracy of the viscous spatial discretization procedure adopted herein. The free-stream Mach number and Reynolds number, based on the length of the plate, for the case considered are M = 0.2 and Re = 10,000, respectively. The exact solution of the incompressible boundary layer equations



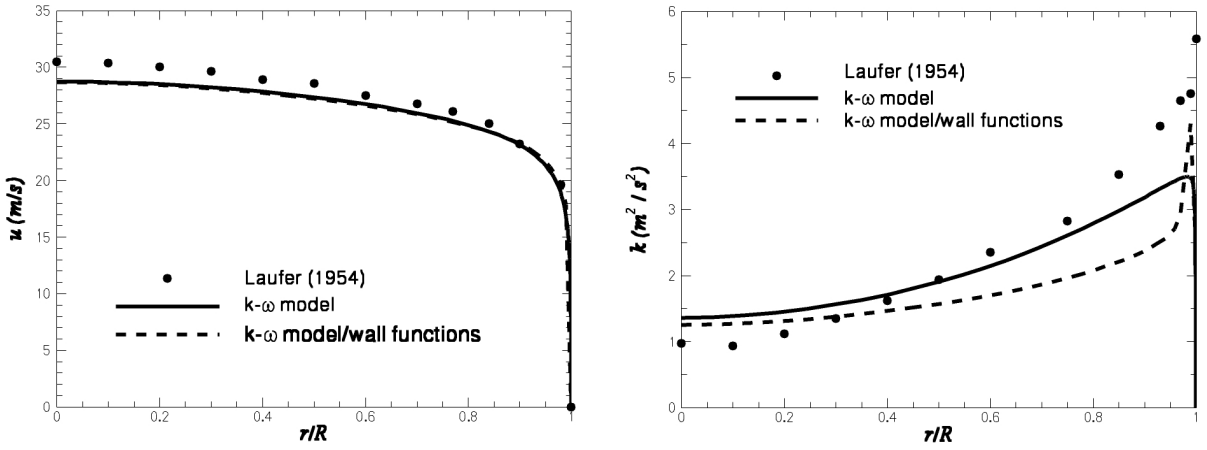
(a) (b)

Figure 3.5: *Laminar flat plate boundary layer: (a) Non-dimensional velocity components at* $\mathrm{Re_x} = 8,000$ *and (b) Estimation of the skin friction coefficient.*

first obtained by Blasius is presented in the textbook by Schlichting [174]. The predicted boundary layer velocity profiles and skin-friction coefficient are shown in Figure 3.5. The computational mesh consisted of a Cartesian mesh with 34,816 cells uni-directionally stretched towards the plate. The first node normal to the plate is located at a distance of approximately 0.2 μm off of the plate. The velocity profiles are plotted in Figure 3.5(a) at $\mathrm{Re_x} = U_\infty x/\nu = 8,000$ against the similarity variable, $\eta = y\sqrt{U_\infty/2\nu x}$. The y-component of the velocity is non-dimensionalized by $\phi = v\sqrt{\mathrm{Re_x}}/U_\infty$. The predicted skin friction coefficient is given in Figure 3.5(b). It can be seen that the velocity components and the skin friction coefficient are in excellent agreement with the Blasius solution, providing validation of the spatial discretization procedure for laminar fluid flows.

3.4.3 Fully-Developed Turbulent Pipe Flow

The validation of the numerical scheme for turbulent flow has been considered by comparing numerical results to the experimental data of Laufer [107] for fully-developed turbulent flow in a pipe with $\mathrm{Re} = 500,000$. Solutions for k-ω turbulence model with both direct integration to the wall and standard wall functions are compared to measured mean axial velocity and turbulent kinetic energy profiles in Figure 3.6. The computations with direct integration were performed using 128 cells in the radial direction with some 16-18 of those cells lying within the laminar sublayer. The first cell off the wall was located at $y^+ \approx 0.05$. The calculations with the wall function formulation were performed using just 32 cells in the radial direction with the first cell at $y^+ \approx 16$. The agree-



(a) (b)

Figure 3.6: *Turbulent pipe flow: (a) radial profiles of axial velocity, u, (b) radial profiles of turbulent kinetic energy, k.*

ment between the experimental data and numerical results for this case is generally quite good. As should be expected, the present implementations of the k-ω model with direct integration and wall functions are both capable of accurately reproducing the characteristic features of fully-developed pipe flow. Moreover, the ability of the k-ω model with direct integration to the wall to agree with the empirical relations defining the turbulent boundary layer velocity profile is evident in Figure 3.7. Although not shown in the figure, as expected the law of the wall relation is also matched perfectly when using standard wall functions.

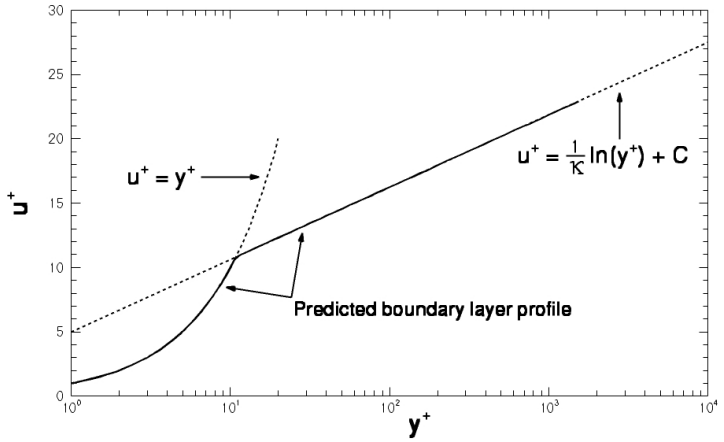


Figure 3.7: *Predicted boundary layer profile for the turbulent pipe flow compared with the empirical viscous sublayer and log-layer relations.*

3.5 Particle-Phase Flux Evaluation

In Section 2.2.2, the equations governing the gas-particle flow were identified as a degenerate hyperbolic system. From a mathematical perspective, the degenerate nature of these equations must be accounted for when designing a numerical scheme for their solution. Although Lagrangian modelling methods can readily deal with particle trajectories that cross, Eulerian finite-volume methods can produce inaccurate and physically incorrect numerical solutions. This is a result of insufficient characteristic fields from which to reconstruct the solution and therefore solution information and accuracy can be lost. The solution of the particle-phase Riemann problem proposed by Saurel *et al.* [173] allows for particle vacuums and for particle paths to cross, however, flows involving the compression of the particle-phase, such as at reflection boundary conditions, are problematic. In their paper, Saurel *et al.* discussed but did not pursue a multi-velocity treatment of the particle-phase, in which the particle-field is split up into distinct velocity families that are advected separately in the flow. Though this method requires additional computing resources in terms of both memory and computing time, compression waves are permitted and, therefore, crossing trajectories and reflection boundary conditions can be more realistically modelled.

The design and implementation of a new multi-velocity formulation for the evaluation of the numerical flux for the particle phase is now discussed in this section. For two-dimensional flow, the particle-phase is decomposed into four distinct velocity families, with velocities in the four quadrants of velocity space associated with particles travelling in the north-east, north-west, south-west, and south-east directions. Each particle family is treated separately. The flux of the particle phase at cell interfaces is then determined using a simplified version of the Riemann solution defined by Saurel *et al.* [173]. The left and right solution states for the particle-phase Riemann problem is determined using the piecewise limited reconstruction discussed in Section 3.3.1. To begin the discussion, the more usual single-velocity formulation is first described, followed by the proposed multi-velocity algorithm.

3.5.1 Single-Velocity Formulation

Determination of the flux values at cell interfaces for the particle-phase is complicated by the degeneracy of the two-phase flow equations as described in Chapter 2. In particular, as the eigen-structure is incomplete, a Roe-type linearization is not possible for evaluating the numerical flux. In an attempt to deal with this mathematical deficiency, a particle-phase Riemann solver for a single-velocity formulation was proposed by Saurel *et al.* [173]. This flux function is now briefly reviewed and its limitations discussed.

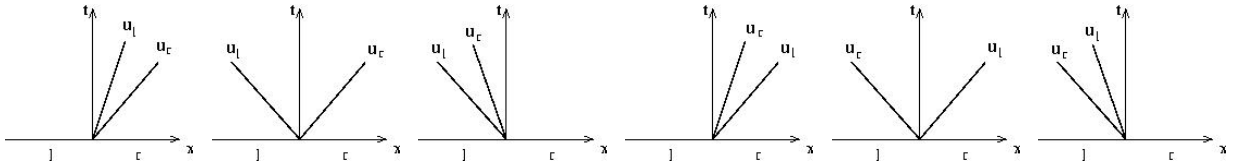


Figure 3.8: *The six wave patterns for the solution of the particle-phase Riemann problem* [173].

Given the left and right particle-phase initial conditions defined by $\mathbf{V}_l = [\,\sigma_{p_l}, u_l, T_{pl}]^T$ and $\mathbf{V}_r = [\,\sigma_{p_r}, u_r, T_{pr}]^T$, three particle expansion wave configurations ($u_l < u_r$) and three particle compression wave configurations ($u_l > u_r$) were identified by Saurel *et al.* and used to construct solutions to the Riemann problem. Refer to Figure 3.8. For the expansion configuration, if u_l, $u_r > 0$ then the interface solution provided by the solution of the Riemann problem is simply the left state. If u_l, $u_r < 0$ then the interface solution state is given by the right state. A strong expansion occurs when $u_l < 0$ and $u_r > 0$. In this case the particles of the left and right states are moving apart in opposite directions, leaving a vacuum state at the interface.

For the compressive configurations ($u_l > u_r$), if u_l, $u_r > 0$ then the interface solution for the Riemann problem is the left state and if u_l, $u_r < 0$ the solution state is the right state. A strong compression of the particles occurs when $u_l > 0$ and $u_r < 0$. In the proposed Riemann solver of Saurel *et al.*, the solution state is then given by $\sigma_p^* = \sigma_{p_l} + \sigma_{p_r}$, $u^* = (\sigma_{p_l} u_l + \sigma_{p_r} u_r)/\sigma_p^*$, and $T_p^* = (\sigma_{p_l} T_{pl} + \sigma_{p_r} T_{pr})/\sigma_p^*$, which is basically a superposition of the left and right solution states.

In all three of the compressive cases, the interface solution compromises the actual physics of the particle motions. For the mildly compressive cases where one solution state is catching up with the other, due to the degeneracy the solution information carried by the overtaken particles is partially lost. In reality, this solution information should be retained. For the more strongly compressive case, the two waves (populations of particles) should simply pass through each other. Instead, the particle concentrations are directly summed and density weighted averages are assigned to the particle velocity and temperature. An example of two crossing particle jets predicted using the single-velocity formulation is shown in the left panel of Figure 3.9. The particle jets are crossing at an angle of 45°. Vacuum conditions are assumed to exist in the channel and so the phase-interaction terms can be neglected. Although this averaging procedure provides the correct average state for the combined left and right moving particle populations, it fails to recognize the presence of the two oppositely moving populations and solution content is lost and not retained in the numerical approximation of the solution. A single particle jet is formed with no transverse momentum. This loss of information can lead to the unphysical results and difficulties near solid

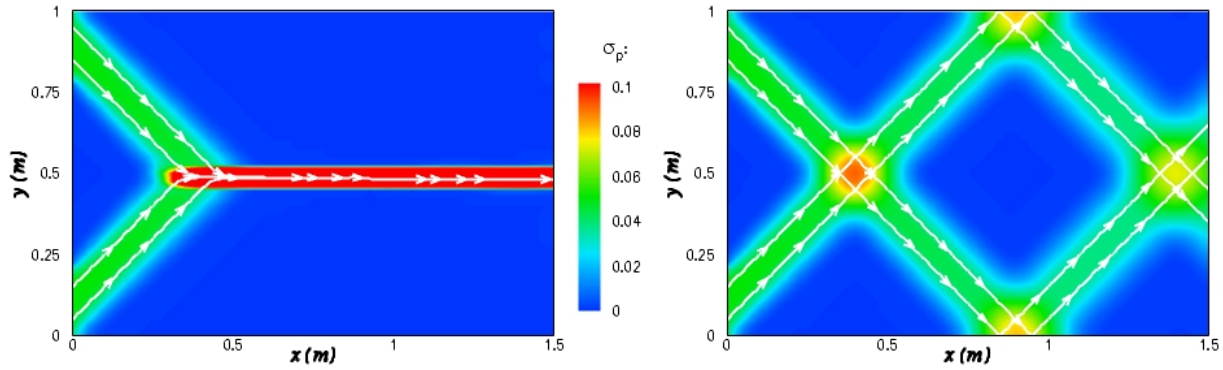


Figure 3.9: *Numerical prediction of the particle-phase concentration contours and streamlines for two impinging particle jets in a vacuum using single-velocity (left panel) and multi-velocity (right panel) formulations.*

boundaries as described by Saurel *et al.* and by Slater and Young [179] and is a basic limitation of any single-velocity Eulerian formulation.

3.5.2 Multi-Velocity Formulation

To alleviate the issues involved with compression waves that occur when using a single-velocity formulation a multi-velocity treatment of the particle phase is proposed here in which the particle-field is split up into distinct velocity families and are advected separately in the flow. For two-dimensional flow, the particle-phase is decomposed into four distinct velocity families, with velocities in the north-east ($u_x > 0, u_y \geq 0$), north-west ($u_x \leq 0, u_y > 0$), south-west ($u_x < 0, u_y \leq 0$), and south-east ($u_x \geq 0, u_y < 0$) directions of velocity space. Eight velocity families would be required for three-dimensional flow.

Each family of particles are assigned their own mass, momentum, and energy and, therefore, are governed by their own equations of motion. The flux at cell interfaces is determined individually for each particle-family using a simplified version of the single-velocity formulation particle-phase Riemann solver of Saurel *et al.* Crossing particle trajectories are inherently accounted for in this approach as seen by the particle-phase streaklines in the right panel of Figure 3.9 for two-impinging jets (interaction with the gas-phase is neglected). Moreover, reflection boundary conditions can be realistically modelled by simply reflecting the particle-phase to the appropriate particle family. For example, as shown in Figure 3.9, a particle-wave travelling in the south-east direction perfectly reflects at the solid lower boundary and continues travelling in the north-east direction without losing mass, momentum, or energy. It should be noted that the particle concentration contours plotted in Figure 3.9 are for the total concentration found by summing the concentrations of all

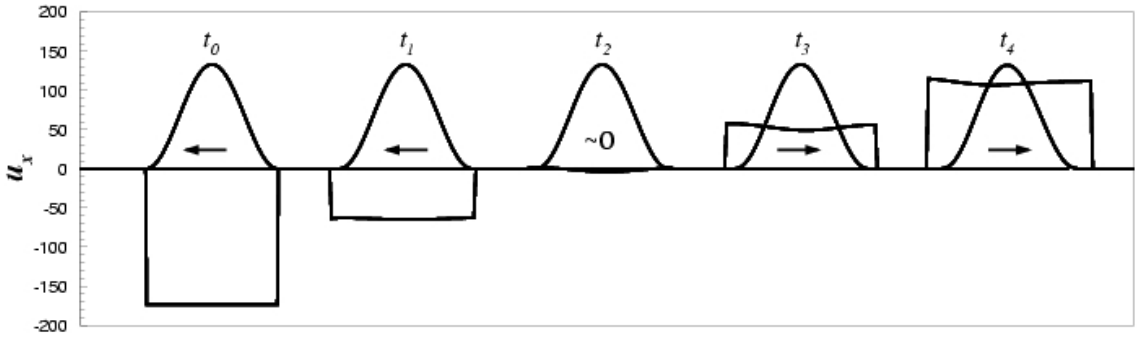


Figure 3.10: *Illustration of the change in velocity families due to momentum transfer between the two phases for a particle-wave with a sine-squared concentration distribution. The particle-wave is initially moving in the opposite direction of the gas-phase.*

four two-dimensional flow particle velocity families.

Switching of the particle-phase velocity family can be required due to the momentum phase interaction with the gas-phase as the system tends towards equilibrium. In this case, a particle-wave travelling in a different direction than the gas-phase may eventually be pulled into a different velocity family. To allow for this, the particle state for each family is averaged into the appropriate family in every computational cell of the mesh after each stage of the time-stepping scheme as required. A mass average is used, thereby enforcing conservation of mass, momentum, and energy. However, it should be noted that some solution content can be lost in this averaging process. An illustration of the change of velocity family due to momentum transfer is provided in Figure 3.10 for a one-dimensional problem which involves two velocity families (left and right travelling). Initially, at t_0, a sine-squared particle-wave is travelling at uniform velocity in the opposite direction of the gas-phase which is travelling to the right at Mach 0.5 and is contained in the left-travelling velocity family. Due to drag, the system will relax into an equilibrium state in which both phases have the same momentum. At t_1 it is seen that the particle wave has slowed down but is still associated with the left-travelling velocity family. As soon as the velocity of the particle wave switches from negative to positive, approximately at t_2, the particle state is mass-averaged into the right-travelling velocity family and the left-travelling velocity family is deleted. The particle-wave then continues to relax into an equilibrium state as seen from times t_3 and t_4.

A limitation of the multi-velocity treatment and the averaging approach described above is that although crossing particle trajectories can be accurately modelled, weakly compressive particle-phase flows with one particle-wave overtaking a slower particle-wave will produce one average wave, as found in the single-velocity formulation. This method also requires some additional computing resources in terms of both memory (twelve additional variables) and approximately one-and-a-half

times the computing time of the single-velocity formulation. However, stronger particle compression waves are easily reproduced and, therefore, crossing trajectories and reflection boundary conditions can be realistically handled.

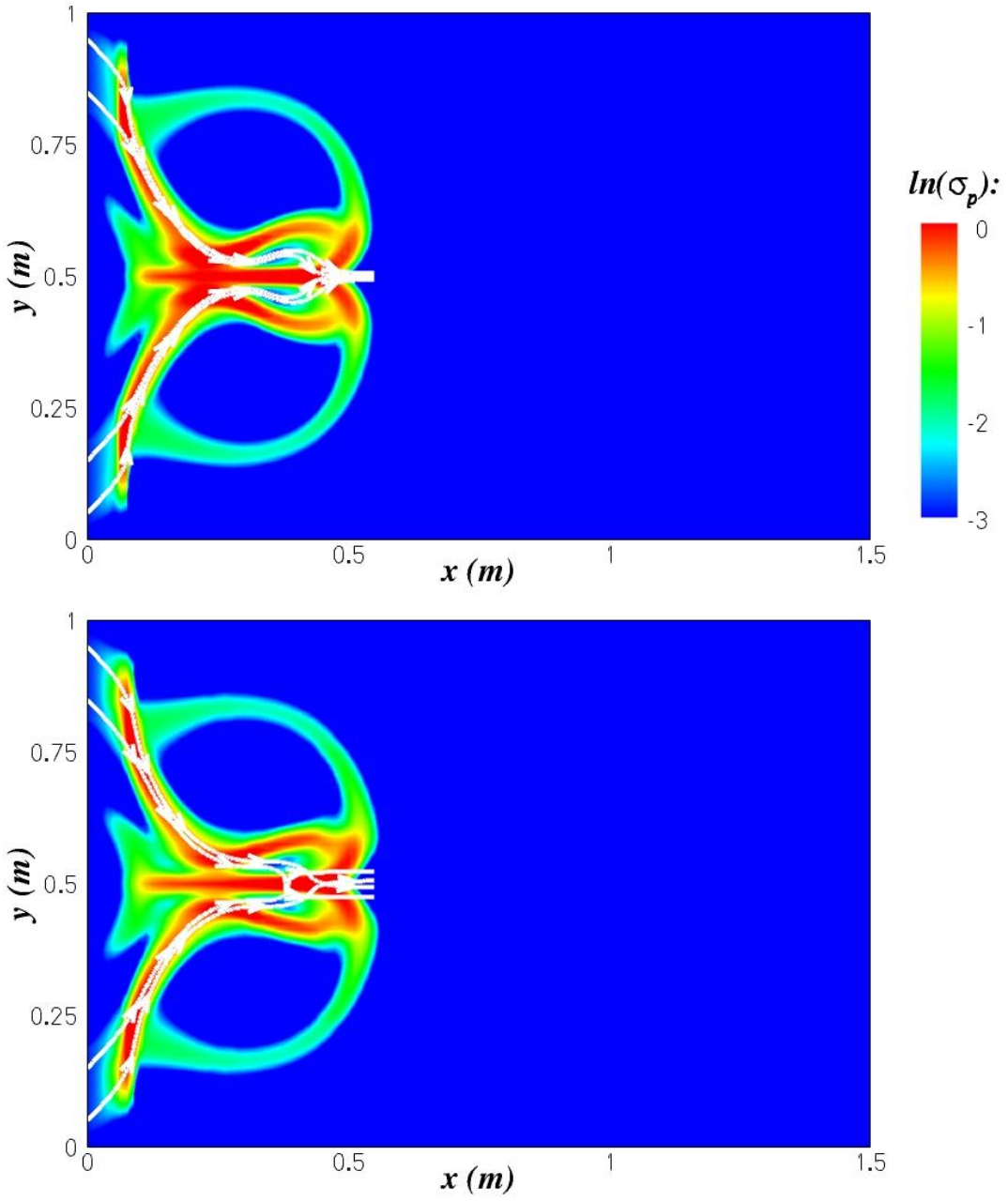


Figure 3.11: *Numerical prediction of the particle-phase concentration contours and streaklines for two impinging particle jets in a quiescent gas at* $t = 16.5$ *ms using single-velocity (top panel) and multi-velocity (bottom panel) formulations.*

3.5.3 Crossing-Particle Jets

A problem involving two impinging particle jets in a vacuum was used to demonstrate the ability of the multi-velocity formulation to accurately predict such flows. Here, the same problem is considered while including the effects of the phase-interaction terms. Initially, the channel contains a uniform quiescent gas at atmospheric conditions with zero particle concentration. As done previously, the particle jets enter at the top and bottom of the channel at a 45° angle from the walls at a speed of 80% of the gas-phase sound speed, $|\vec{u}| = 0.8a$. Reflective boundary conditions are applied at the top and bottom surfaces of the channel walls. Fixed and constant extrapolation boundary conditions are used at the left and right boundaries of the domain, respectively. The computation is performed using both the single- and multi-velocity formulations and the results are plotted in Figures 3.11 and 3.12 at 16.5 ms and 43 ms, respectively. Particle-phase concentration contours and streaklines are shown.

As shown in Figure 3.11, the prediction of the the particle-phase concentration by both formulations is nearly identical soon after the collision of the particle jets. However, the multi-velocity formulation demonstrates a definite crossing of the particle-phase streamlines that is absent from the single-velocity formulation. Moreover, the inability of the single-velocity formulation to allow the particle-phase streamlines to cross leads to an instability in the computation that is seen in Figure 3.12. This instability is related to the mathematical properties of the Eulerian description and the numerical solution scheme. It has similarities to a Kelvin-Helmholtz instability in fluid dynamics. It is experienced only when using a higher-order scheme and is a result of the higher-order reconstruction in areas with strong shear layers in the particle-phase. The multi-velocity formulation does not experience this instability since shear layers for each velocity family are avoided and the impinging jets simply pass through each other, thereby providing a symmetric solution.

3.6 Block-Based Adaptive Mesh Refinement

Adaptive mesh refinement techniques which automatically adapt the computational grid to the solution of the governing partial differential equations can be very effective in treating problems with disparate length scales. Following the approach developed by Groth *et al.* [70, 67] for computational magnetohydrodynamics, a flexible block-based hierarchical data structure has been developed and is used in conjunction with the finite-volume scheme described above to facilitate automatic solution-directed mesh adaptation on multi-block quadrilateral mesh according to physics-based refinement criteria. The current AMR formulation borrows from previous work by Berger [14, 15], Berger and Colella [16], Berger and LeVeque [17], Quirk [155], and DeZeeuw and Powell [38] and has similarities

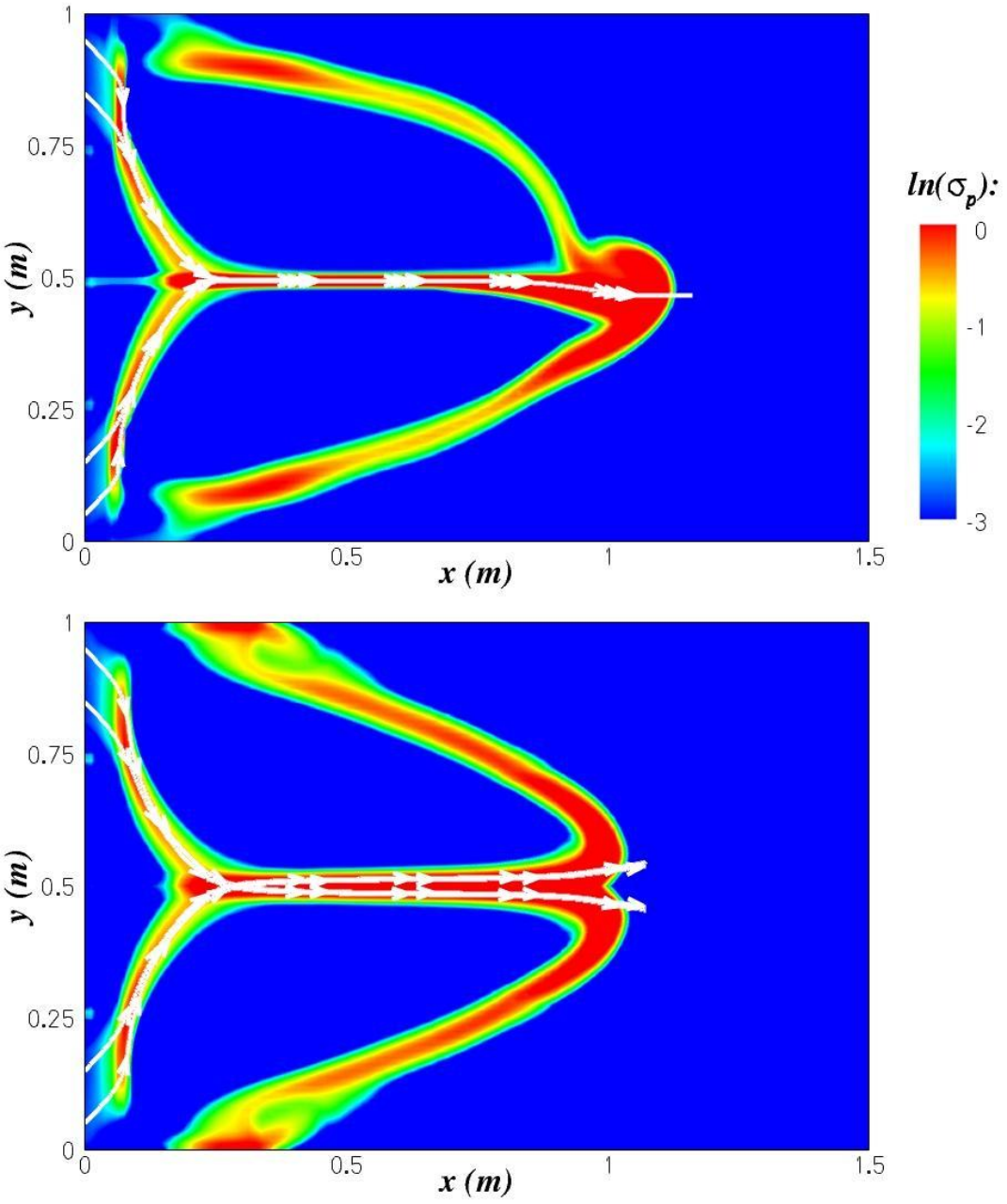


Figure 3.12: *Numerical prediction of the particle-phase concentration contours and streaklines for two impinging particle jets in a quiescent gas at $t = 43$ ms using single-velocity (top panel) and multi-velocity (bottom panel) formulations.*

with the block-based approaches described by Quirk and Hanebutte [156] and Berger and Saltzman [18]. A primary distinction of this work is the use of curvilinear (arbitrary quadrilateral) mesh as opposed to the Cartesian mesh that is used in most of the previous work. The use of quadrilateral mesh blocks makes the application of the block-based AMR more amenable to flows with thin

boundary layers and permits anisotropic refinement as dictated by the initial mesh stretching. Note that cell-based AMR schemes on curvilinear mesh have been explored in previous work by Davis and Dannenhoffer [37], Sun and Takayama [182], and Bramkamp *et al.* [21].

In this work, the governing equations are integrated to obtain area-averaged solution quantities within quadrilateral computational cells and these cells are embedded in structured blocks consisting of $N_x \times N_y$ cells, where N_x and N_y are even, but not necessarily equal integers. Solution data associated with each block are stored in indexed array data structures and it is therefore straightforward to obtain solution information from neighbouring cells within blocks. Mesh adaptation is accomplished by the dividing and coarsening of appropriate solution blocks. In regions requiring increased cell resolution, a "parent" block is refined by dividing itself into four "children" or "offspring". Each of the four quadrants or sectors of a parent block becomes a new block having the same number of cells as the parent and thereby doubling the cell resolution in the region of interest. This process can be reversed in regions that are deemed over-resolved and four children are coarsened into a single parent block. The mesh refinement is constrained such that the grid resolution changes by only a factor of two between adjacent blocks and the minimum resolution is not less than that of the initial mesh. Standard multi-grid-type restriction and prolongation operators are used to evaluate the solution on all blocks created by the coarsening and division processes, respectively. Although several approaches are possible, for this study, the coarsening and division of blocks are directed using multiple physics-based refinement criteria as previously done by Paillère *et al.* [142] and Powell *et al.* [153, 152]. Six derived flow quantities or refinement criteria, ϵ_k, are used herein,

$$\epsilon_1 \propto \left|\vec{\nabla}\rho\right| \quad \epsilon_2 \propto \left|\vec{\nabla}\cdot\vec{v}\right| \quad \epsilon_3 \propto \left|\vec{\nabla}\times\vec{v}\right| \tag{3.58}$$

$$\epsilon_4 \propto \left|\vec{\nabla}\sigma_p\right| \quad \epsilon_5 \propto \left|\vec{\nabla}\cdot\vec{u}\right| \quad \epsilon_6 \propto \left|\vec{\nabla}\times\vec{u}\right| \tag{3.59}$$

The first three quantities correspond to local measures of the density gradient, compressibility, and vorticity of the gas-phase and enable the detection of contact surfaces, shocks, and shear layers. Refinement criteria for the particle-phase are defined by the next three quantities which provide local measures of the particle concentration gradient and the divergence and vorticity of the particle velocity field. These quantities will enable the detection of high and low (vacuum) particle concentrations, particle-phase compression and expansion waves, and particle-phase shear layers. Division of a mesh block is performed if the refinement measures are greater than a user-specified threshold. Similarly, four mesh blocks are coarsened into one parent block if the refinement measures are less than a user-specified threshold in all of the candidate blocks. For most of the results reported herein, typical thresholds for division and coarsening are 0.5 and 0.1, respectively.

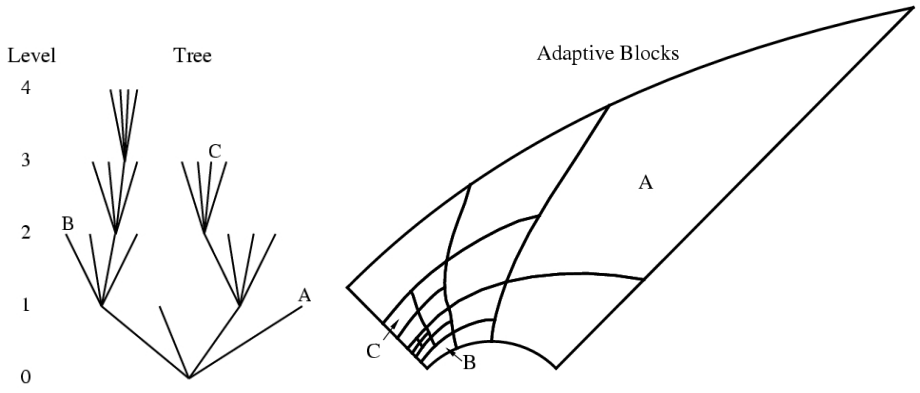


Figure 3.13: *Solution blocks of a computational mesh with four refinement levels originating from one initial block and the associated hierarchical quadtree data structure. Interconnects to neighbours are not shown.*

In order that the solution algorithm for the multi-phase flow equations can be applied to all blocks in a more independent manner, some solution information is shared between adjacent blocks having common interfaces. This information is stored in an additional two layers of overlapping "ghost" cells associated with each block. At interfaces between blocks of equal resolution, these ghost cells are simply assigned the solution values associated with the appropriate interior cells of the adjacent blocks. At resolution changes, restriction and prolongation operators, similar to those used in block coarsening and division, are employed to evaluate the ghost cell solution values. Within the AMR approach, additional inter-block communication is also required at interfaces with resolution changes to strictly enforce the flux conservation properties of the finite-volume scheme [15, 16]. In particular, the interface fluxes computed on more refined blocks are used to correct the interface fluxes computed on coarser neighbouring blocks and ensure that the fluxes are conserved at block interfaces.

A hierarchical tree-like data structure with multiple "roots", multiple "trees", and additional interconnects between the "leaves" of the trees is used to keep track of mesh refinement and the connectivity between solution blocks. This interconnected "forest" data structure is depicted in Figure 3.13. The blocks of the initial mesh are the roots of the forest which are stored in an indexed array data structure. Associated with each root is a separate "quadtree" data structure that contains all of the blocks making up the leaves of the tree created from the original parent blocks during mesh refinement. One of the advantages of the hierarchical quadtree data structure is that it readily permits local mesh refinement. Local modifications to the multi-block mesh can

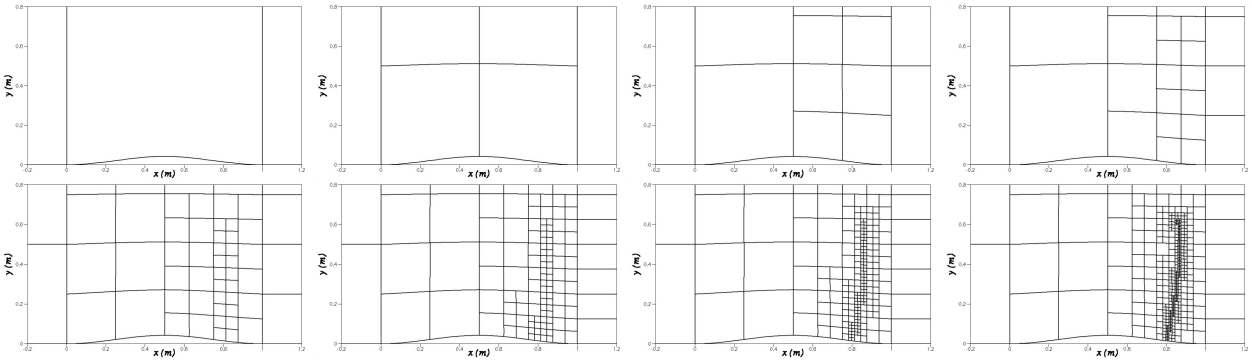


Figure 3.14: *Sequence of multi-block meshes derived by the AMR algorithm for the inviscid transonic flow over a bump. The initial mesh is given in the top left figure. Refined meshes follow. The block structure is shown for a region surrounding the bump. The entire domain and individual cells are not shown. Mesh refinement statistics are given Table 3.2.*

be performed without re-gridding the entire mesh and re-calculating solution block connectivity.

An example illustrating the adaptation of a two-dimensional multi-block quadrilateral mesh for an inviscid transonic flow over a bump, $\mathrm{M}_\infty = 0.85$, is shown in Figures 3.14 and 3.15. The geometry of the bump is defined by a sine-squared function, $y = 0.042 \sin^2(\pi x)$ for $x \in [0, 1]$. Figure 3.14 shows and initial grid (top left panel) consisting of eight blocks and 8,192 cells, $(N_x, N_y) = (32, 32)$ and seven meshes derived by applying the refinement algorithm. The solution is converged 4-5 orders of magnitude on each grid level. Note that the figures do not show the entire computational domain but only the bump section where the majority of the refinement occurs. The solution block boundaries are depicted in the figures. Some statistics corresponding to the mesh refinement are summarized in Table 3.2 where N_{blocks} is the total number of blocks, N_{cells} is the total number of cells, and η is a measure of the efficiency of the block-based AMR scheme defined by

$$\eta = 1 - N_{\text{cells}}/N_{\text{uniform}} \tag{3.60}$$

where N_{cells} is the total number of cells and N_{uniform} is the total number of cells that would have

Table 3.2: *Mesh refinement statistics for the multi-block quadrilateral mesh for the inviscid transonic flow over a bump.*

Mesh level	0	1	2	3	4	5	6	7
N_{blocks}	8	11	26	41	77	167	281	536
N_{cells}	8,192	11,264	26,624	41,984	78,848	171,008	287,744	548,864
η	0.000	0.656	0.797	0.920	0.962	0.980	0.991	0.966

been used on a uniform mesh composed of cells of the finest size on the current mesh. The efficiency of the AMR scheme improves as the number of refinement levels increase. This indicates the ability of the block-based AMR approach to deal with flows having disparate spatial scales, providing reduced numbers of cells while maintaining cell resolution in areas of interest as illustrated in Figure 3.15 for the transonic flow over a bump. The pressure contours are shown and the solution on the initial mesh is given in the left panel of the figure. A shock has formed near the trailing edge of the bump. The refinement scheme effectively clusters solution blocks on the shock. The solution on the final mesh, after seven levels of refinement, is shown in the right panel of the figure. The mesh, in this case, has an refinement efficiency of $\eta = 0.966$ and consists of 536 blocks and 548,864 cells. Note that each level of refinement in the grid introduces cells that are typically smaller by a factor two in each spatial dimension. Practical calculations may have 10-15 levels of refinement. In the case of 15 levels of refinement, the finest cells in the mesh are more than 32,000 (2^{15}) times smaller than the coarsest cells.

In the preceding example of mesh refinement, the solution was converged to steady-state on each mesh level using the five stage optimally smoothing scheme before refining the mesh. Mesh refinement during an unsteady problem is also possible by refining the mesh at a specified frequency. Typically the frequency is related to the dimensions of the individual block, for example $\min(N_x/2, N_y/2)$, which will help ensure that a neighbour block is refined before any wave phenomena of interest (such as a shock) enters the block. An example of mesh refinement during an unsteady problem is shown in Figure 3.16 for an inviscid shock reflection involving an inclined wedge. The incident shock Mach number and wedge angle are 1.36 and 25°, respectively, corresponding to a Mach reflection. The refined block structure is shown in the left panel and the density contours are shown in the right panel and are plotted at a time of 2.44 ms into the unsteady flow. It

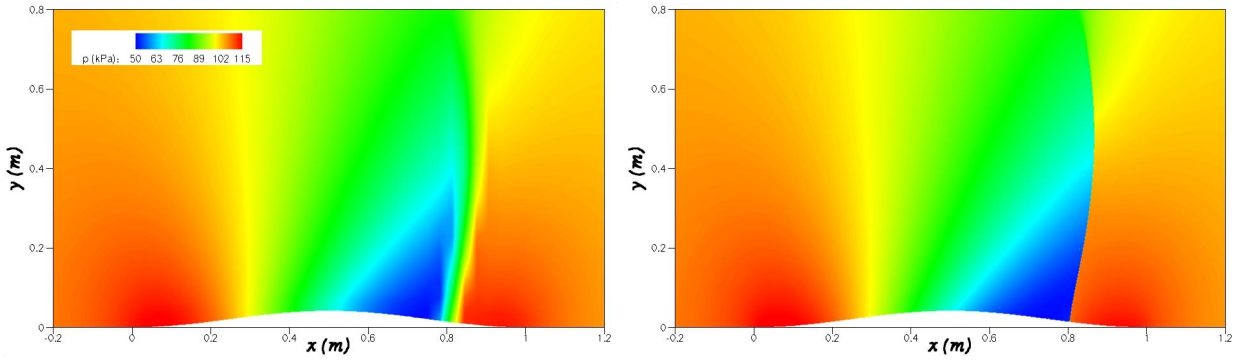


Figure 3.15: *Pressure contours for the inviscid transonic flow over a bump on the initial mesh (left panel) and after seven mesh refinements (right panel).*

can be seen that the solution blocks have been clustered on the incident shock, the Mach stem, the reflected shock, and the slip-stream. Seven levels of refinement were used resulting in 896 blocks and 175,616 cells, $(N_x, N_y) = (14, 14)$.

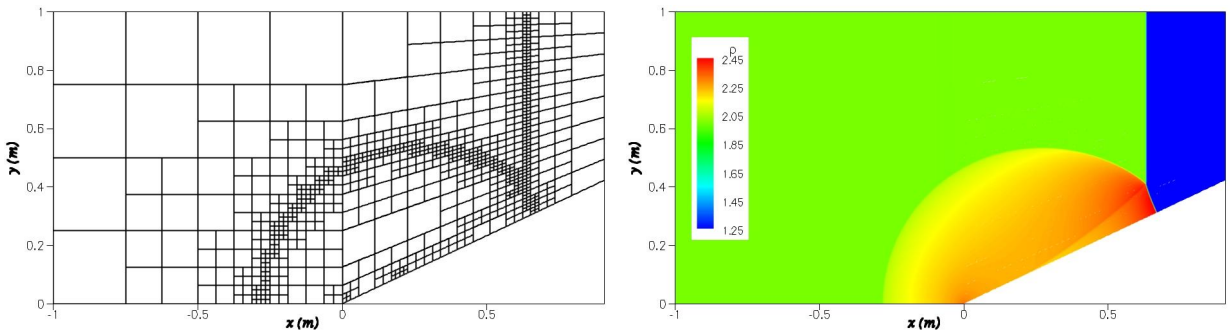


Figure 3.16: *Density contours and adapted mesh block structure for a shock reflection on a wedge at 2.44 ms with seven levels of refinement (896 blocks, 175,616 cells, $\eta = 0.891$). The incident shock Mach number and wedge angle are 1.36 and* 25° *corresponding to Mach reflection.*

3.7 Parallel Implementation

Although the block-based AMR approach described above is somewhat less flexible and incurs some inefficiencies in solution resolution as compared to cell-based approaches (i.e., for the same solution accuracy, generally more computational cells are introduced in the adapted grid), the block-based method offers many advantages over cell-based techniques when parallel implementation of the solution algorithm is considered and computational performance issues are taken into account. In particular, the multi-block quadrilateral mesh and quadtree data structure lends itself naturally to domain decomposition and thereby enables efficient and scalable implementations of the solution algorithm on distributed-memory multi-processor architectures.

A parallel implementation of the block-based AMR scheme has been developed using the C++ programming language and the MPI (message passing interface) library [66]. Use of these standards greatly enhances the portability of the computer code and has enabled very good parallel performance. Domain decomposition is carried out by merely farming the solution blocks out to the separate processors, with more than one block permitted on each processor. A simple stack is used to keep track of available (open) processors. For homogeneous architectures with multiple processors all of equal speed, an effective load balancing is achieved by exploiting the self-similar nature of the solution blocks and simply distributing the blocks equally among the processors. In doing so, all blocks are treated equally and, currently, no use is made of the hierarchical data

structure nor grid partitioning techniques to preferentially place neighbouring blocks on the same processors. With 10 blocks per processor, the maximum load imbalance attained by this simple block distribution procedure is less than 10% (near perfect load balancing is achieved if the number of block is a exact multiple of the number of available processors). For heterogeneous parallel machines, such as a network of workstations and computational grids, a weighted distribution of the blocks can be adopted to preferentially place more blocks on the faster processors and less blocks on slower processors.

In order to carry out mesh refinement and inter-block communication, a complete copy of the hierarchical quadtree data structure is stored on each processor. This is possible because, unlike cell-based unstructured meshing techniques, the block-based tree data structure is not overly large. The structure need only retain the connectivity between the solution blocks as opposed to a complete map of the cell connectivity required by general unstructured mesh procedures. Inter-processor communication is mainly associated with block interfaces and involves the exchange of ghost-cell solution values and conservative flux corrections at every stage of the multi-stage time integration procedure. Message passing of the ghost-cell values and flux corrections is performed in an asynchronous fashion with gathered wait states and message consolidation, and as such, typically amounts to less than 5-8% of the total processor time.

Implementation of the algorithm has been carried out on a Beowulf cluster of 26 4-way Hewlett-Packard Alpha SMP servers (14 ES40 and 12 ES45 servers) and 35 4-way Hewlett-Packard Integrity rx4640 servers with a total of 244 processors and 482 Gbytes of distributed memory. Each of the 56 ES40 processors is a 667 MHz Alpha EV67 processors with 8 Mbytes of L2 cache and the 48 ES45 processors are 1 GHz MHz Alpha EV68CB processors with 8 Mbytes of L2 cache. The 140 rx4640 processors are 1.5 GHz Itanium 2 processors with 6 Mbytes of L3 cache. A high-bandwidth low-latency Myrinet network is used to connect the servers in the cluster. The parallel performance and scalability of the numerical algorithm on this parallel architecture was assessed by conducting a series of numerical experiments using up to 80 of the Itanium 2 processors. The results of these tests are now discussed.

The parallel relative speed-up, S_p, given by $S_p = t_1/t_p$, and the parallel relative efficiency, E_p, given by $E_p = S_p/p$, are determined as a function of the number of processors, p, where t_p is the time required to solve a fixed-size problem using p processors and t_1 is the time required to solve the problem using a single processor. In a fixed-size problem, the total workload is constant and independent of the number of processors. The problem chosen here is the inviscid transonic bump flow investigated above. The domain is decomposed into 128 solution blocks and the relative speed-up test is performed with 1, 4, 8, 16, 32, and 64 processors such that perfect load-balancing

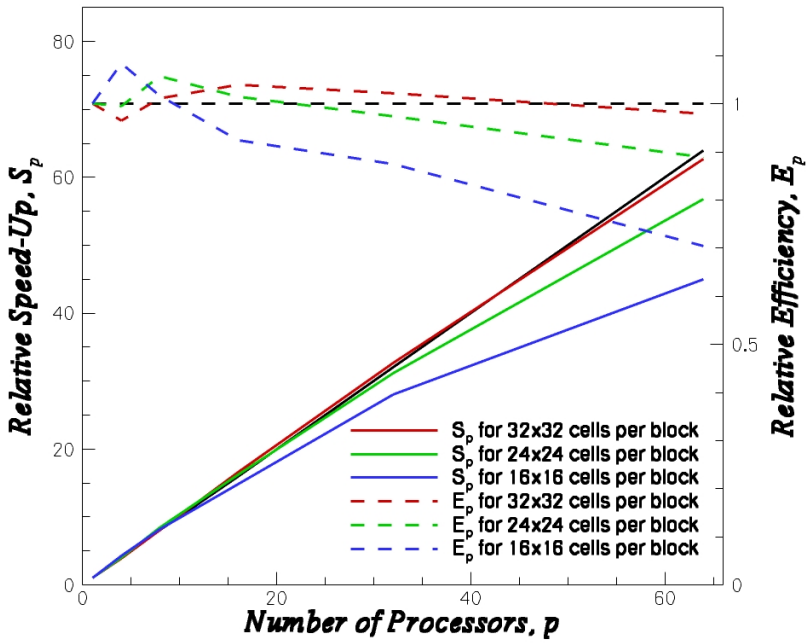


Figure 3.17: *Parallel relative speed-up, S_p, and parallel relative efficiency, E_p, for a fixed-size problem with perfect load-balancing using up to 64 processors.*

is achieved (each processor has the same amount of work). The results of this speed-up test are shown in Figure 3.17 for three different block sizes. It can be seen that with perfect load-balancing, block sizes of 32×32 cells per block provide the best scale-up with a nearly linear 62.68 parallel relative speed-up and 97.9% parallel relative efficiency for up to 64 processors. The results for the different block sizes also indicate that the number of cells used in each solution block will ultimately affect the trade-off between parallel efficiency and efficiency of the block-based mesh refinement procedure, with large block sizes improving the former but reducing the latter.

The effect of imperfect load-balancing on the parallel relative speed-up and efficiency is examined for a test case similar to the prediction of the unsteady shock-wedge interaction problem also described above in the previous section. Since the AMR algorithm keeps the mesh blocks clustered at the shocks as they propagate and reflect, the number of solution blocks is not constant. Therefore, it is difficult to distribute the solution blocks evenly among processors and imperfect load-balancing occurs. However, improved load balancing is expected as the number of blocks increases and improved load balancing is obtained. For the purpose of this test, the parallel relative speed-up and efficiency are measured on up to 80 processors from 1.2 ms to 1.6 ms of the flow solution. The number of solution blocks range from 350 to 596. Each of the solution blocks contain $(N_x, N_y) = (28, 28)$ cells. The relative parallel speed-up and relative parallel efficiency for this case is presented in Figure 3.18. It should be noted that for a high number of processors, the load balancing is

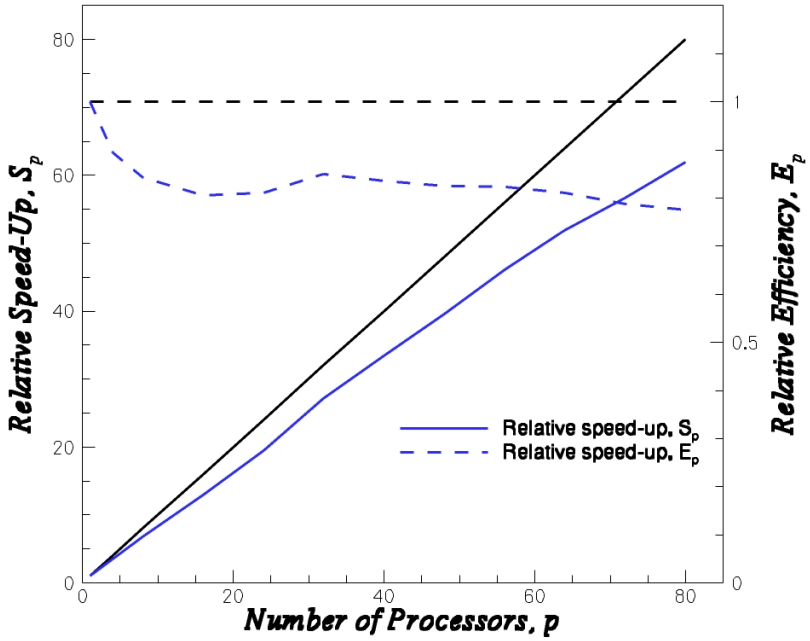
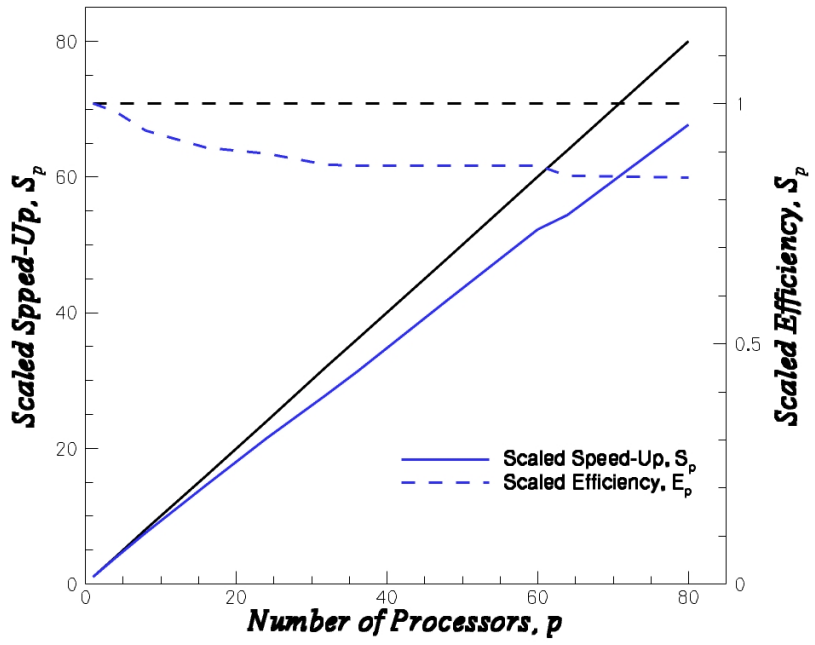


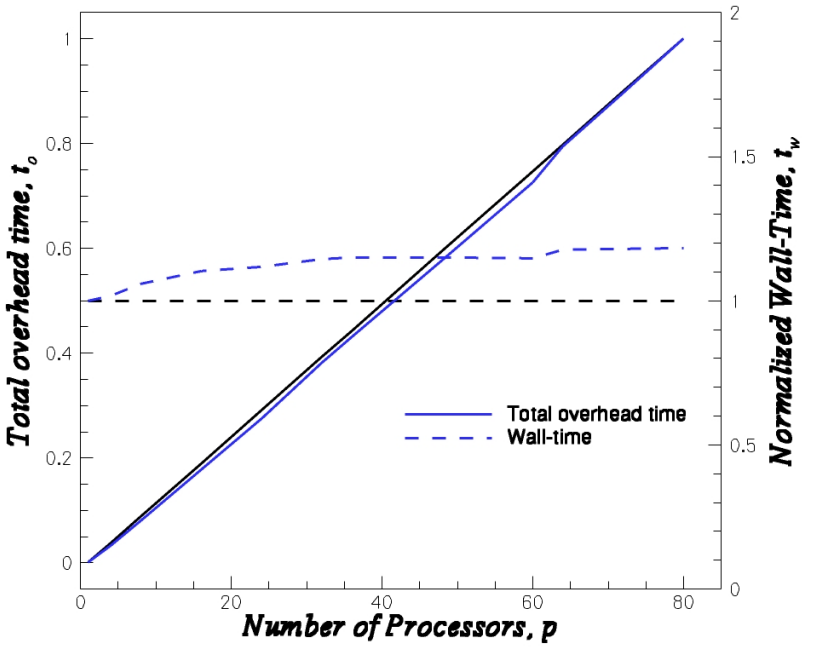
Figure 3.18: *Parallel relative speed-up, S_p, and parallel relative efficiency, E_p, for a fixed-size problem involving the block-based AMR scheme using up to 80 processors and 28×28 cells per solution block.*

poor for the first few refinement levels due to the low number of available solution blocks. The poor load-balancing leads to a less efficient parallel scaling, achieving 77% efficiency for up to 80 processors. Note that the parallel relative efficiency drops fairly rapidly initially but then flattens out as the number of processors increases.

An indication of how a parallel numerical algorithm can incorporate a fixed workload per processor can be determined by considering a scaled-size problem, in which the workload scales with the number of processors. Here, the important performance parameters are the parallel scaled speed-up, S_p, given by $S_p = (t_1/t_p)p$ and the parallel scaled efficiency, E_p, given by $E_p = S_p/p$. The scaled parallel performance parameters are determined using an inviscid test case involving unsteady shock-wave interaction in a square domain with reflecting boundaries. It can be seen in Figure 3.19(a) that when considering a scaled-size problem, the parallel scaled speed-up is nearly linear and maintains at 84.6% efficiency up to 80 processors. The over-head time, t_o, given by $t_o = pt_p - t_1$ and normalized by $\max(t_o)$ and the normalized wall-time, t_w, given by $t_w = t_p/t_1$ are presented in Figure 3.19(b) as a function of the number of processors. An algorithm is considered to be highly scalable if the iso-efficiency function, the relationship between the over-head time and the number of processors, is linear. It can be seen in Figure 3.19(b) that the iso-efficiency function is indeed linear and the normalized wall-time remains fairly constant for up to 80 processors.



(a)



(b)

Figure 3.19: *(a) Parallel scaled speed-up, S_p, and parallel scaled efficiency, E_p, and (b) parallel overhead time, t_o, and wall-time, t_w for a scaled-size problem using up to 80 processors.*

Chapter 4

Mesh Adjustment Scheme for Embedded Boundaries

In this Chapter, a scheme is described in which the body-fitted multi-block mesh used in the finite-volume scheme proposed in Chapter 3 is locally adjusted to arbitrarily embedded boundaries that are not necessarily aligned with the original mesh. This will allow for the modelling and treatment of the motion of the combustion interface of solid propellant rocket motors. Fluid flows involving moving boundaries are relevant to many other aerospace engineering applications as well, such as: aerodynamic control surfaces, helicopter rotor blades, compressor and turbine blades, and stage separation in launch vehicles. Numerical solution of these unsteady problems must account for the motion of the boundaries through the domain of interest. This can be accomplished by regenerating or adjusting the computational mesh according to the object's location [82, 12], by using overlapping meshes [13], or by representing the boundary as a force-term which influences the flow solution in a manner consistent with fluid dynamics [150, 151]. These approaches are now briefly summarized.

The Arbitrary Lagrangian-Eulerian (ALE) method pioneered by Hirt *et al.* [82] is a popular method for dealing with moving boundaries [215, 48, 58, 5]. This scheme involves the use of a grid, typically unstructured, that moves according to the motion of the boundaries. Conservation is strictly satisfied in ALE methods, however, the quality of the mesh can be severely degraded when large scale motions are involved requiring re-meshing and/or untangling procedures [197].

An alternative approach is the use of overlapping meshes, also known as overset mesh or Chimera method, where a separate body-fitted mesh is constructed around each object in the computational domain and is overlapped with each other and a simpler, in many cases Cartesian, mesh encompassing the computational domain of interest. This method was originated by Benek *et al.* [13] and a review of recent work on this method was given by Noack and Slotnick [132]. Interpolation is required between the meshes during the solution of the governing equations. These interpolation schemes are typically non-conservative and non-monotone [131], which may be problematic for many applications.

The immersed boundary method was devised by Peskin [150, 151] in order to predict fluid-structure interactions in biological fluid dynamics. In this scheme the boundary is represented as a force-term in the governing equations which influences the flow solution in a manner consistent with fluid dynamics. The immersed boundary method has been successfully applied to complex laminar and turbulent flows [86, 195, 143].

Although the methods described above have been shown to be effective for treating embedded boundaries, in this work a new approach is proposed that requires only local alterations to the mesh, is conservative, and can be readily coupled with the parallel AMR finite-volume scheme proposed in Chapter 3 for body-fitted multi-block mesh in a straight-forward manner. Not only does this scheme allow for rapid and robust mesh generation involving complex embedded boundaries, it also enables the solution of unsteady flow problems involving bodies and interfaces moving relative to the flow domain. The mesh adjustment algorithm described here has similarities with the Cartesian cut-cell techniques developed by De Zeeuw and Powell [38] and Aftosmis *et al.* [3] except that the underlying mesh is no longer restricted to a Cartesian mesh. The readjustment of the mesh for moving embedded boundaries follows the approach used in the Cartesian cut cell methods developed by Bayyuk *et al.* [12] and Murman *et al.* [126]. Unlike the Cartesian cut-cell method, the mesh adjustment algorithm presented here does not result in the generation of tiny cut-cells which can place severe restrictions on the time-step and do not permit the construction of consistent and accurate operators for viscous (elliptic) fluxes [32]. This scheme has been implemented within the block-based adaptive mesh refinement (AMR) numerical framework described in the previous Chapter.

4.1 Mesh Adjustment Algorithm

The motivation for developing a mesh adjustment scheme for arbitrarily embedded boundaries stems from the desire to numerically predict flow involving complex bodies which can be moving relative to the computational domain. The scheme proposed here requires only local alterations of the mesh allowing for its implementation within the context of the block-based AMR scheme described previously. Many of the techniques used by Cartesian cut-cell methods, such as bounding box filters and ray-casting [1], are used here to improve the efficiency of the mesh adjustment algorithm.

An illustration of the mesh adjustment algorithm is given in Figures 4.1–4.9. In Figure 4.1 the mesh of the initial domain is shown along with two embedded NACA0012 aerofoils. These objects are stored as n-sided polygons or n-element polylines with additional data stored at each vertex of the shape to specify local characteristics of the embedded interfaces (such as boundary condition

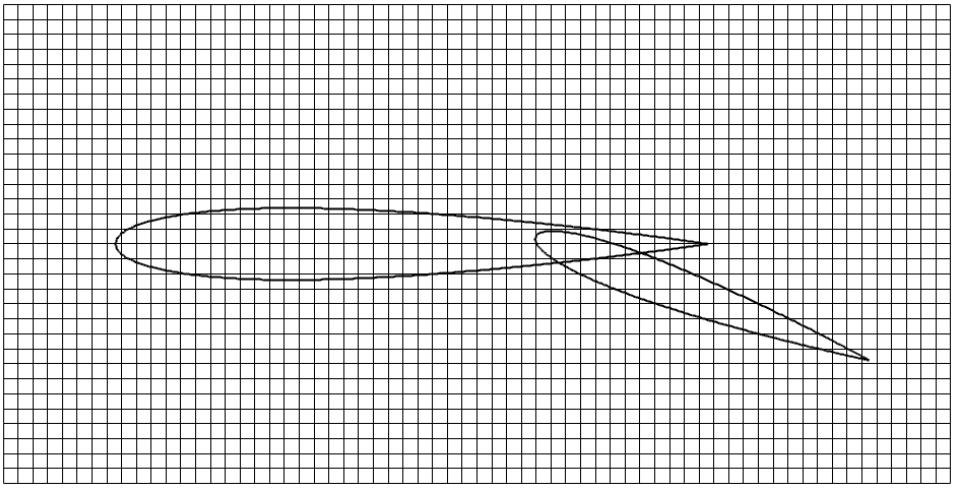


Figure 4.1: *Example discretized domain with 4,096 cells,* $(N_x, N_y) = (32, 128)$*, and two embedded boundaries for the illustration of the mesh adjustment algorithm.*

type, velocity, and temperature). Although not required for this scheme, for illustrative purpose, the domain is discretized by a uniform Cartesian mesh (the proposed algorithm is more generally applicable to any body-fitted multi-block mesh). As for Cartesian cut-cell methods, the union of the embedded boundary components is found before the mesh is adjusted to the boundary [1]. This allows for a more robust and efficient implementation of the algorithm. The Weiler-Atherton algorithm is an efficient and robust method for finding the union of two intersecting polygons [210] and is adopted here. A description of the Weiler-Atherton algorithm and other useful polygon

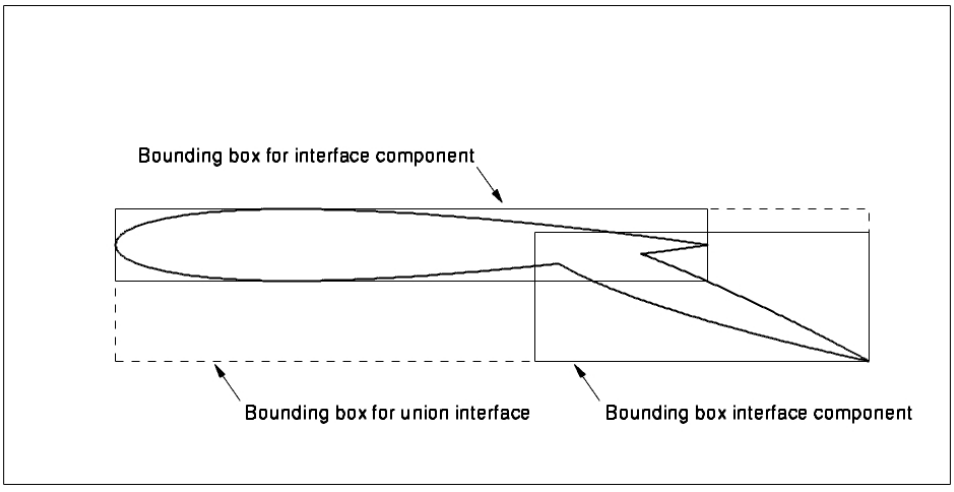


Figure 4.2: *Interface of union of the two embedded NACA0012 aerofoils and their associated bounding boxes.*

routines is provided in Appendix A. The union of the two embedded NACA0012 aerofoils is shown in Figure 4.2. Included in this figure are the bounding boxes for the individual interface components as well as the union interface. A bounding box is defined by the maximum and minimum Cartesian coordinates of the object of interest and, therefore, contains that object entirely. The bounding boxes of the interface components are used as filters to quickly determine which computational cells are potentially intersected by or are contained by the associated union interface.

The first stage of the mesh adjustment procedure is to tag/paint each cell as an *active* cell, *inactive* cell, or an *unknown* cell and every node of each cell as *known* or *unknown*. Note that all cells and nodes are initialized as active and known, respectively. The unknown nodes (of an unknown cell) are candidates for adjustment. Most of the active cells (and known nodes) are quickly identified by comparing each of the nodes of a cell with bounding boxes of each of the interface components. If all of the nodes are deemed outside the bounding box then the cell is deemed active. Otherwise, each edge of the cell with contrasting tags are tested for potential intersection points with each edge of the interface union polygons. If an intersection exists between the cell edge and an interface edge then the nodes of the cell edge and the cell itself are tagged as unknown. If no edge/interface intersections exist, ray-tracing is performed to determine if the cell is deemed active or inactive. The ray-tracing algorithm is performed by counting the number of intersections between the line composed of the cell centroid and a reference point within the embedded boundary (typically the centroid) and each edge of the embedded boundary (refer to Appendix A). An odd

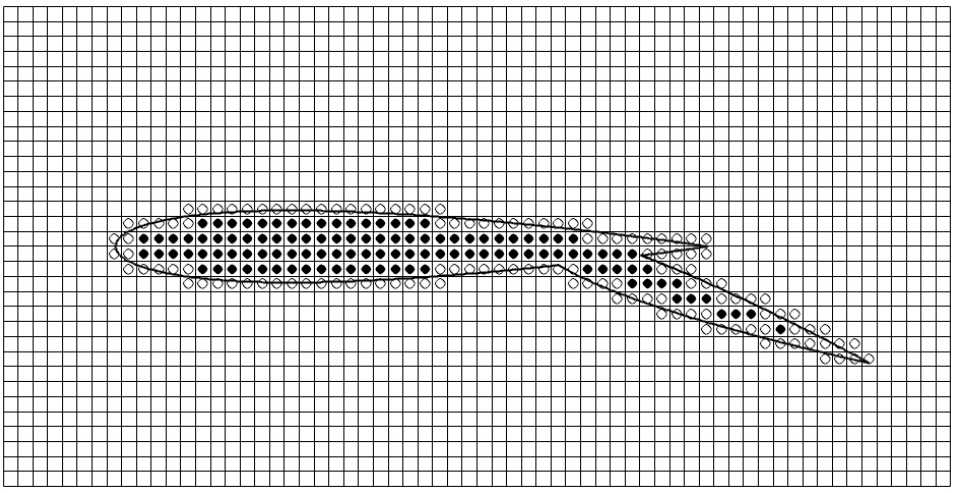


Figure 4.3: *Tagging of the computational cells: cells with filled circles are tagged as inactive (internal to the embedded boundary), cells with hollow circles are tagged as unknown and are candidates for adjustment, and all other cells are active (external to the embedded boundary).*

number of intersections indicates that the cell is outside the interface (active) and an even number of intersections indicates that the cell is inside the interface (inactive). Note that the actual point of intersection is not required during this initial tagging procedure and the point of intersection must be contained on both line segments. The result of this procedure is shown in Figure 4.3.

The NACA0012 aerofoil includes a sharp trailing edge. To provide an accurate representation of the embedded boundary, the next step of the adjustment algorithm is to identify the unknown cell that contains these sharp points and move the cell's node that is closest to the sharp point onto that point. Note that the points of union between the two NACA0012 aerofoils components are also defined as sharp points. The adjustment of the mesh to these four sharp points is high-lighted by the arrows in Figure 4.4. All mesh nodes that are adjusted to the embedded boundary are tagged as *aligned*.

The mesh adjustment is comprised of two main stages. The primary adjustment involves merely relocating the unknown mesh nodes that are closest to the intersection point between the spline defining the embedded boundary and the mesh lines. The resulting adjusted grid is shown in Figure 4.5. All potential adjustment directions (along north, south, east, or west mesh lines) and intersection points are tabulated before the actual adjustment occurs. This allows for the parsing/reduction of the adjustments near zones of interest such as near the trailing edge of an aerofoil. Here the nodes internal to the aerofoil are potentially an equal distance from the embedded body in opposite directions, e.g., north and south. These nodes are marked for adjustment in both directions. To ensure a symmetric adjustment, these choices are eliminated and the neighbour nodes

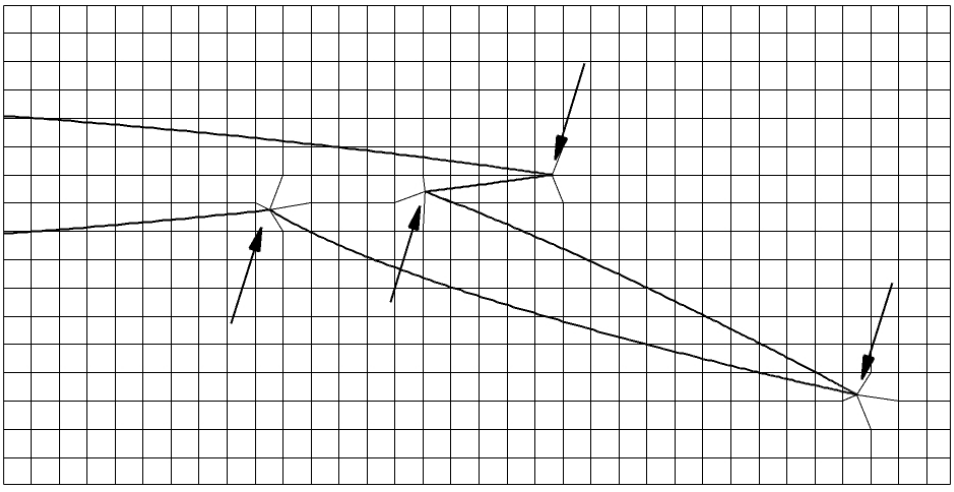


Figure 4.4: *Adjustment of the mesh to sharp corners defined in the embedded boundary: two at the trailing edge of the aerofoils and two at points of union between the two components.*

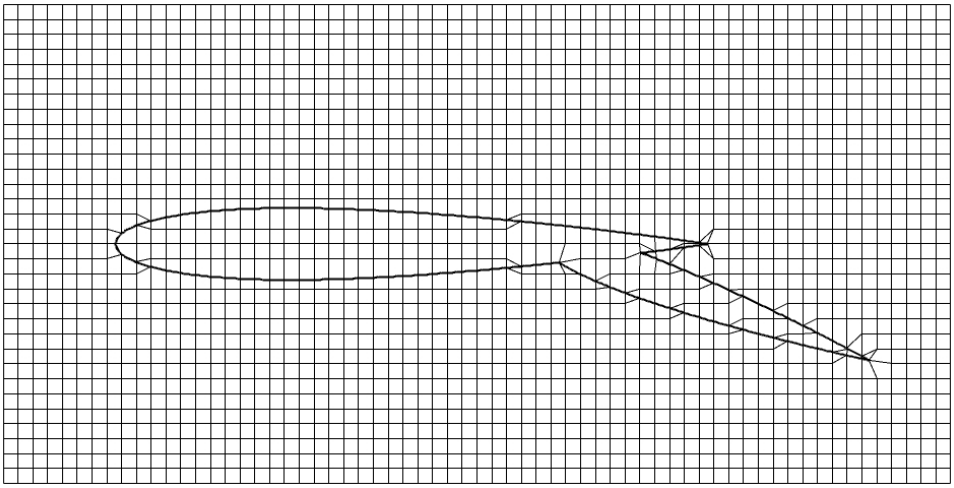


Figure 4.5: *Result of the primary mesh adjustment.*

are marked for adjustment instead, e.g., the $j+1$ and $j-1$ nodes to the north and south are marked for adjustment. As done previously, all mesh nodes that are adjusted to the embedded boundary are tagged as *aligned*. It was found that the robustness and accuracy of the adjustment procedure was greatly improved if this primary adjustment procedure was performed twice in succession. In particular, this is essential on coarse meshes in which an embedded boundary may pierce the edge of a cell more than once. Note that aligned nodes are not permitted to be readjusted.

It can be seen in Figure 4.5 that the primary mesh adjustment creates computational cells that are bisected diagonally by the embedded boundary. The second mesh adjustment stage is required to account for these cells. This is accomplished by choosing the closest not-aligned node

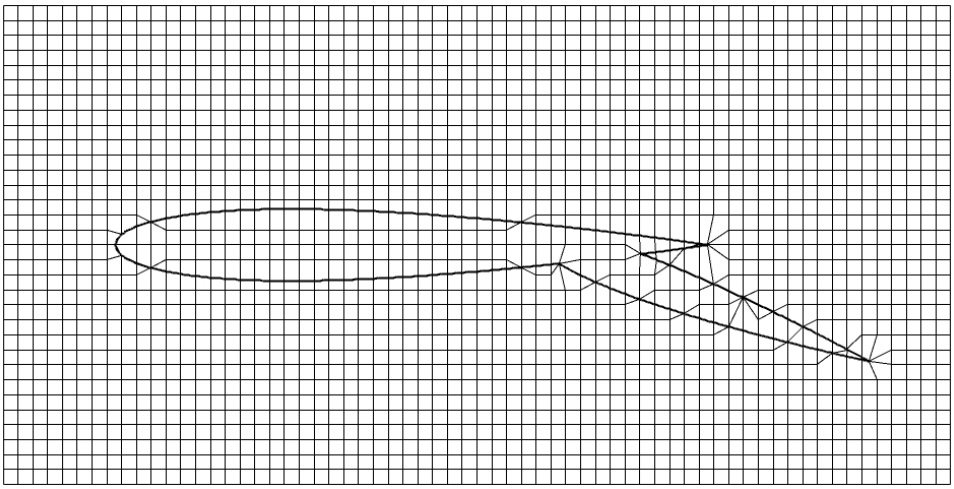


Figure 4.6: *Result of the second mesh adjustment step.*

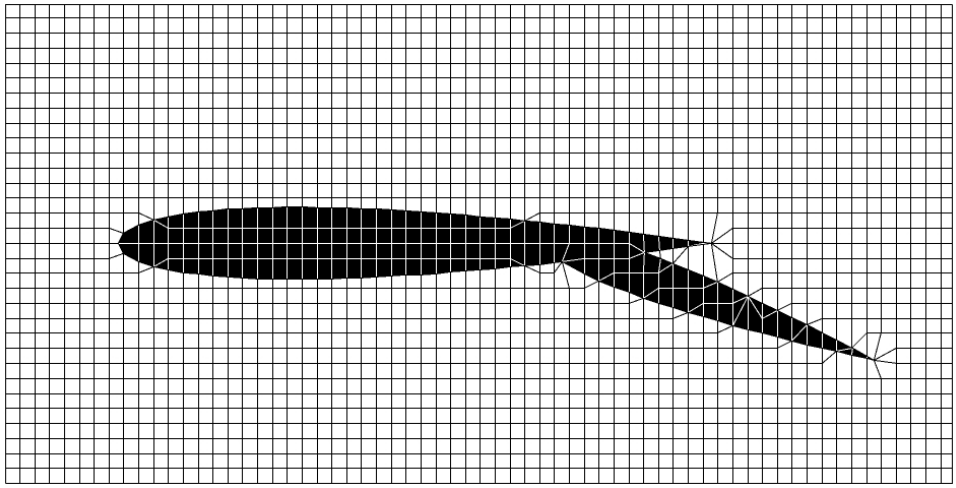


Figure 4.7: *Final tagging of the computational cells: shaded cells are tagged as inactive (internal to the embedded boundary) and all other cells are active (external to the embedded boundary).*

to the embedded boundary and relocating it to that boundary point, as shown in Figure 4.6. This adjustment can potentially create a cell with zero area – either the cell under consideration or one of its neighbours. If this occurs, the adjusted node is reset to its initial location and the other not-aligned node is adjusted. If no valid choice is available then the mesh is under-resolved and the mesh cannot be adjusted without refinement. Some triangular cells are generated as a result of this second adjustment step. These cells are treated as degenerate quadrilateral cells with two coincident nodes. Note that the degenerate edges are only generated on the embedded boundary.

The final stage of the mesh adjustment algorithm is to re-tag all of the cells previously marked as unknown. The ray-tracing algorithm discussed above is used to determine if the adjusted cell is inside (active) or outside (inactive) the embedded boundary. In situations with multiple embedded boundaries which could potentially have unique boundary conditions and motion characteristics, all inactive cells are tagged using the number of the interface of union that it is associated with (possible values range from one to the number of union interfaces). All union interfaces are stored in an array and, therefore, this tagging method allows for quick identification of the associated embedded boundary. All active cells are tagged as zero. The final adjusted mesh is given in Figure 4.7.

One of the primary advantages of the proposed mesh adjustment scheme is that it will generate a piece-wise linear representation of the embedded boundary while still maintaining the (i, j) data structure of the original body-fitted mesh, as shown by Figure 4.8 for a section of the embedded boundary. Note that the inactive nodes are retained to maintain the mesh data structure and may be reactivated in computations involving moving boundaries. The resulting mesh allows for

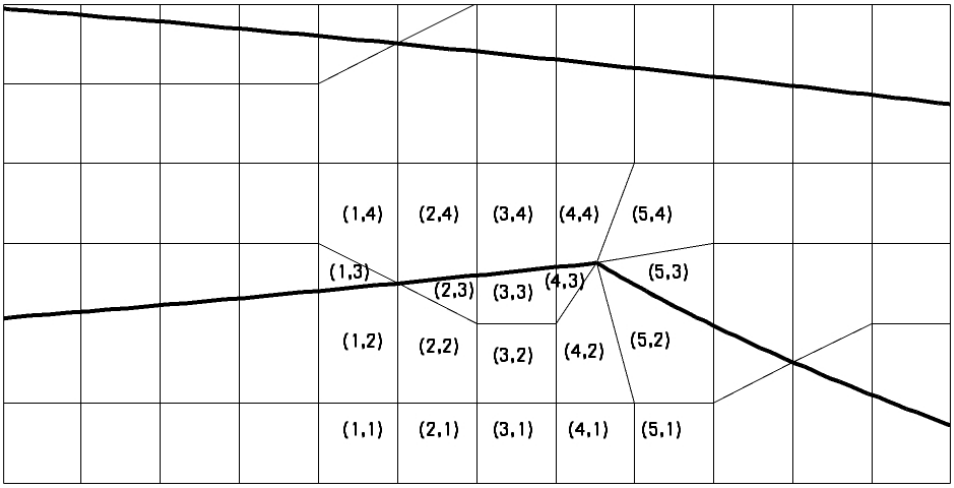


Figure 4.8: *Example of the (i, j)-indexing on an adjusted mesh.*

accurate calculation of cell areas and straightforward application of boundary conditions. Another advantage of this mesh adjustment algorithm is that very small cut-cells are not introduced and cell merging techniques are not required. The ratio of the smallest to largest neighbour cell areas is generally found to be not less than about 0.2–0.25. Since only local alterations are made to the mesh, the need for inter-solution-block communication is also not required and is, therefore, transparent to the parallel block-based AMR scheme. Moreover, application of the block-based AMR allows for a more detailed representation of the embedded boundary.

Use of the AMR scheme for the two embedded aerofoils is presented in Figure 4.9. The block structure of the initial adjusted mesh is shown in the left panel and includes 8 blocks and 2,048 cells. After four refinements, 497 blocks and 127,232 cells are used. To avoid excessive tangling/skewing of the mesh, the mesh is unadjusted (restored to its original form) before the mesh is refined and

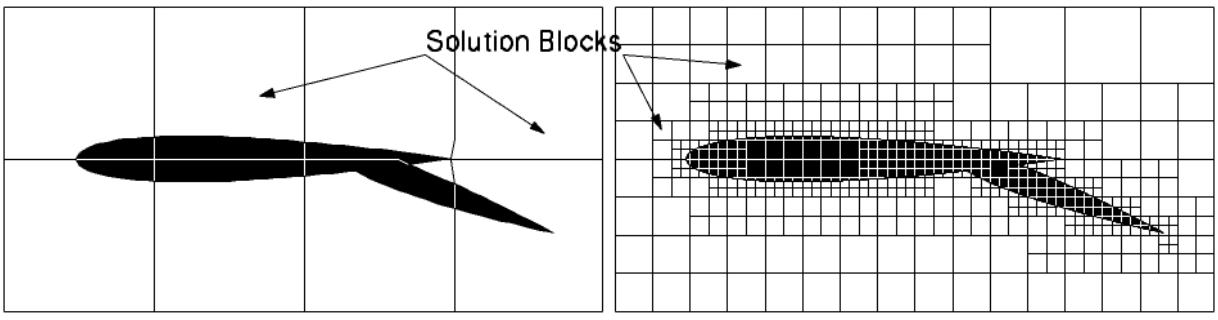
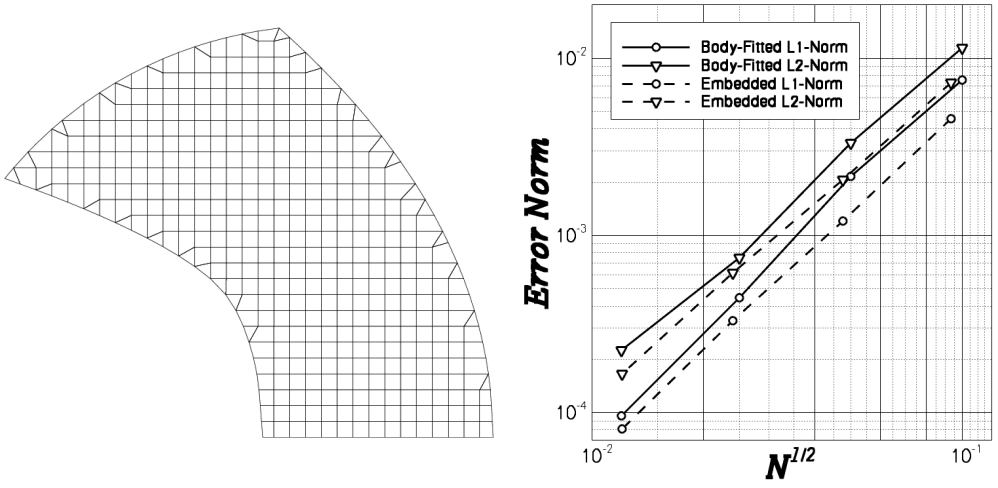


Figure 4.9: *Application of the mesh adjustment scheme with the block-based AMR procedure. The initial mesh (left panel) contains 8 blocks and 2,048 cells and after four refinements (right panel) with 497 blocks and 127,232 cells. The mesh block structure is shown (cells not shown).*

is readjusted after refinement. Note that to perform the restriction or prolongation of the solution information from the old mesh to the new mesh, the adjusted mesh before the refinement must be saved. Three copies of the grid are required: the unadjusted mesh, the adjusted mesh, and the previously adjusted mesh (before AMR is performed). Prolongation of the solution information from a parent solution block into the child solution block is performed by simple injection. Restriction of the solution information is determined using an area-weighted average of the fine cell solution information of the cells that intersect with the coarse cell. The Weiler-Atherton algorithm can also be used to determine the polygon of intersection between the two cells (triangular or quadrilateral polygons) [210].

4.1.1 Ringleb's Flow

The test problem used in Section 3.3.3 is re-considered to demonstrate the accuracy of the inviscid spatial discretization on a mesh adjusted to an embedded boundary. Here the embedded boundary represents the domain of the problem. A sample adjusted Cartesian mesh is shown in Figure 4.10(a) with 429 cells. The L_1- and L_2-norms of the solution error are plotted in Figure 4.10(b). The error-norms for the body-fitted mesh are included for comparison. For the adjusted Cartesian mesh, the slopes of the L_1- and L_2-norms are 1.93 and 1.81, respectively. However, considering only the two finest meshes, the L_1- and the and L_2-norms are 2.03 and 1.89, respectively. This indicates that the scheme is indeed second order accurate, even with the embedded boundary treatment.



(a) (b)

Figure 4.10: *Ringleb's flow: (a) Adjusted mesh with 429 cells and (b) Computed norms of the solution error.*

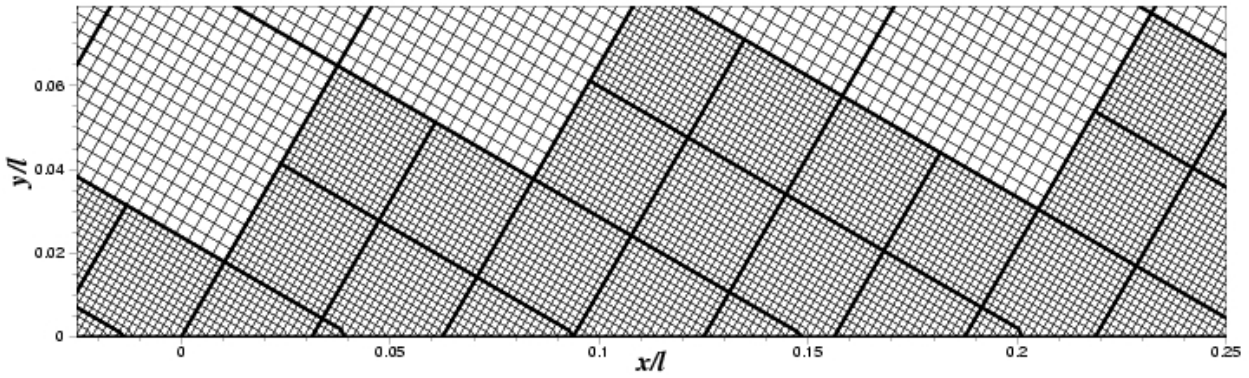


Figure 4.11: *Adjusted mesh at the leading edge of an embedded flat plate for a refined multi-block grid rotated at* 30° *(6 levels of refinement with 360 active blocks and 80,023 active cells). Block boundaries are indicated by the thick black lines. Each block contains 256 cells,* $(N_x, N_y) = (16, 16)$.

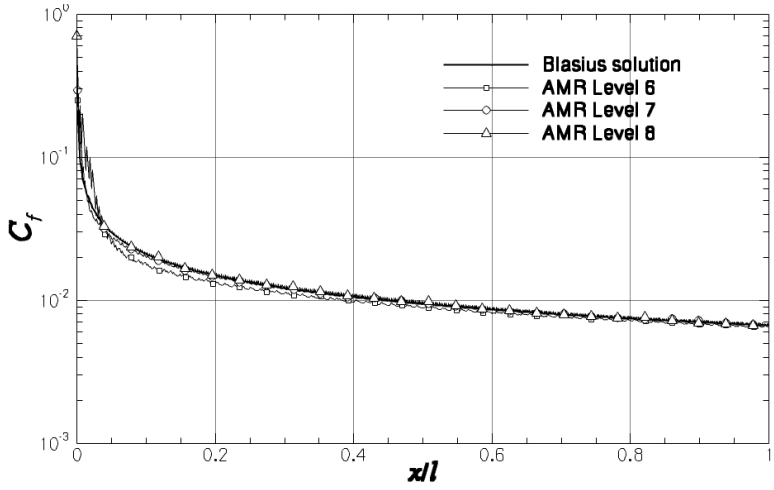
Moreover, it should be stated that for stationary (non-evolving) embedded boundaries, the spatial discretization scheme is strictly conservative and monotonicity of the solution is readily enforced via the slope limiting procedure, both of which were described in Chapter 3.

4.1.2 Laminar Flat Plate Boundary Layer

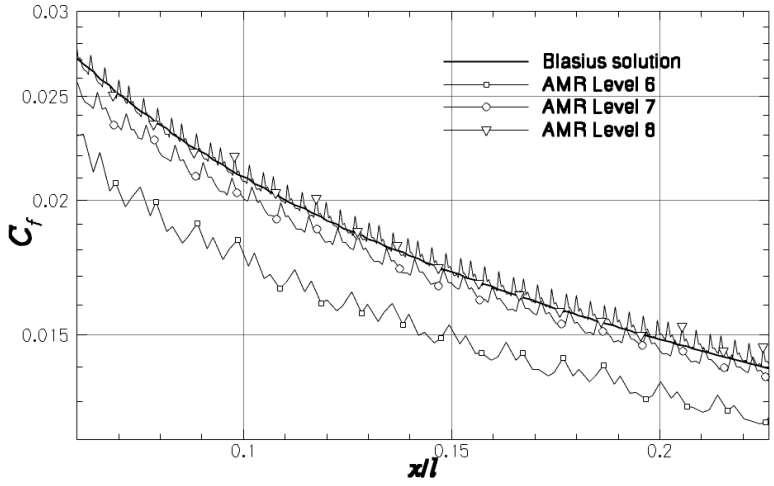
The computation of laminar flow over a flat plate is now considered to explore the accuracy and capability of the proposed scheme for predicting viscous flows. Coirier and Powell [32] considered this case to investigate the use of the Cartesian cut-cell approach for computing viscous flows and were able to show accurate prediction of the mean-flow quantities. However, the resulting skin-friction coefficient proved to be quite oscillatory. This is a direct result of the extremely small cut-cell generated by the Cartesian cut-cell approach which makes the creation of a consistent and accurate viscous discretization virtually impossible. It is useful to explore the improvement over the cut-cell method offered by the adjusted mesh algorithm proposed here.

In this work, a non-axis aligned flat plate is embedded at 30° to the mesh. The freestream Mach number is 0.2 and the Reynolds number (based on the length of the plate) is 10,000. The mesh in the vicinity of the leading edge of the plate is shown in Figure 4.11. The initial mesh consists of 360 active blocks and 80,023 active cells after six refinements. The predicted skin-friction coefficient for a Reynolds number of 10,000 is shown in Figure 4.12(a) for the initial mesh and after two additional refinements. The initial mesh does not provide the desired resolution in the boundary layer and the skin-friction coefficient is slightly under predicted. Improvement in the drag estimation is provided by the additional mesh refinements. More importantly, the oscillatory nature of the skin-friction predictions produced by the cut-cell method are experienced here to a much smaller degree.

The cut-cell method of Ref. [32] resulted in peak-to-peak changes in the skin-friction coefficient of approximately 0.0025 whereas the variations in the current predictions are at least an order of magnitude less. Moreover, Figure 4.13 indicates that, as the flow Reynolds number is decreased, less resolution is required to accurately predict the skin-friction coefficient in a monotonic fashion. This is because the boundary-layer is generally thicker and contains more computational cells. The preceding results would seem to confirm the applicability of meshes created by the adjustment scheme for viscous flows, at least for low-to-moderate Reynolds numbers.



(a)



(b)

Figure 4.12: *Estimation of the skin-friction coefficient for a non-axis aligned flat plate at* $\mathrm{M} = 0.2$ *and* $\mathrm{Re} = 10,000$ *for six, seven, and eight levels of refinement: (a) prediction for the entire plate and (b) a close-up of the prediction near the leading edge of the plate.*

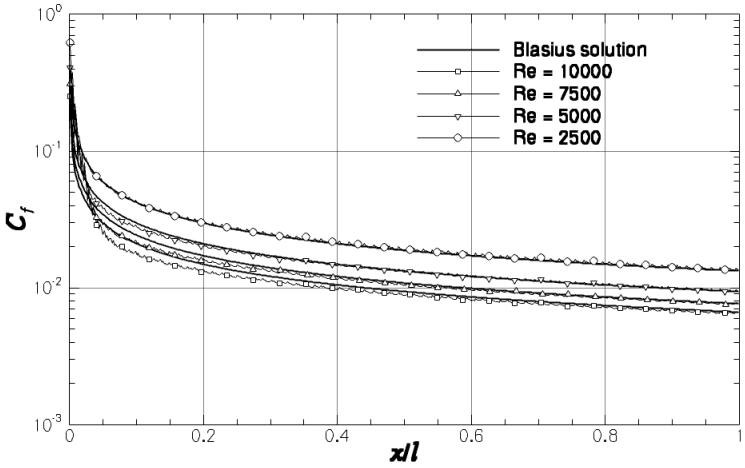


Figure 4.13: *Estimation of the skin-friction coefficient for a non-axis aligned flat plate at* $\mathrm{M} = 0.2$ *for Reynolds numbers of* $2,500$*,* $5,000$*,* $7,500$*, and* $10,000$ *on the mesh shown in Figure 4.11.*

4.1.3 Parallel Performance

It is useful to see how the parallel performance of the proposed finite-volume scheme is affected by the embedded boundary treatment. In order to do this, a fixed size problem is considered that involves the supersonic flow over a stationary ellipse. A similar problem is considered in Section 4.2.3 in which the flow over a translating ellipse is computed. For the current test case, a rectangular domain is decomposed into 640 solution blocks, each with $(N_x, N_y) = (32, 32)$ cells, of which 120 of the blocks were adjusted. These blocks were partially rendered inactive. An additional 160 blocks were completely inactivated by the mesh adjustment scheme. Though these inactive cells are maintained in the data structure, they are not used in the solution calculation. As a result, the mesh adjustment scheme alters the work-load of some of the solution blocks. Note that the percentage of inactivity of the solution blocks due to the adjustment of the mesh to an arbitrary interface is currently not considered when distributing the solution blocks to available processors and can, therefore, lead to an uneven load-balance among the available processors. The parallel performance for this test case is summarized in Figure 4.14. It can be seen that the efficiency of the parallel implementation suffers due to the load-imbalance resulting from the mesh adjustment scheme. However, only a 25% loss in efficiency was experienced over 80 processors and it is felt that the parallel performance is still very good considering the complexity of the problem. Redistribution of the solution blocks based on the percentage of inactivity would likely improve the efficiency of the implementation, however, this was not pursued since this distribution of the solution blocks could quite quickly become non-ideal when considering moving embedded boundaries and dynamic redistribution of solution blocks would be required.

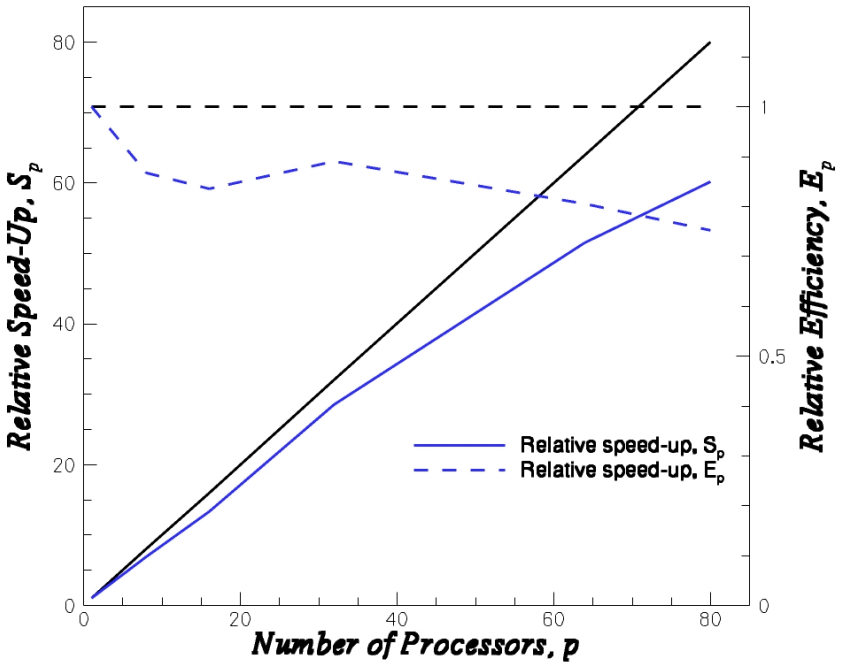


Figure 4.14: *Parallel relative speed-up, S_p, and parallel relative efficiency, E_p, for a fixed-size problem involving the mesh adjustment scheme using up to 80 processors and 32×32 cells per solution block.*

4.2 Dynamic Embedded Boundaries

By performing only local alterations to the mesh, motion of embedded boundaries can be readily accounted for by first unadjusting the mesh and then re-applying the mesh adjustment algorithm according to the new location of the embedded boundaries. This is the approach used by the Cartesian cut-cell community [12, 126, 85]. In this work, embedded boundaries undergoing a prescribed rigid body motion or evolving motion due to a physical process are considered. The latter is directly relevant to the modelling of the core flows of solid propellant rocket motors. Evolving embedded boundaries are modelled using a parallel block-based AMR level set method which is described in Section 4.3 to follow.

The algorithm for moving embedded boundaries is performed in the following sequence of steps:

1. Determine the velocity at all spline points of each interface component.
2. Determine the union of the embedded boundaries.
3. Perform mesh adjustment algorithm.
4. Perform the solution of equation (3.6). The velocity at the Gauss point of cell edges that are aligned with the embedded boundaries are determined from the corresponding interface

component.

5. Determine the new location of the embedded boundary components and recompute the velocity at all spline points if necessary.
6. Determine the union of the embedded boundaries.
7. Unadjust the mesh (store previous adjustment).
8. Perform mesh adjustment and redistribute solution information as required.
9. Loop through steps 4-8 until the computation is finished.

Unadjusting the mesh before performing the mesh adjustment at the new locations for the embedded boundaries avoids excessive tangling of the mesh, however, other complications can arise. Due to this step, some of the computational cells that were formerly active may now be inactive. Conversely, some inactive cells may now be active. A re-averaging of the solution data is required to ensure conservation of the solution quantities. The solution content of any cells that have been newly deactivated is area-averaged into neighbouring active cells. The solution content of all of the active cells involved in the mesh adjustment procedure are then determined by taking the area-average of all of the previously active cells that intersect with newly active cells. This is similar to the restriction process required for AMR and has the same grid requirements. Here, the previously adjusted mesh corresponds to that of the previous time-step.

4.2.1 Piston Problem

A one-dimensional piston problem is now considered to assess the conservation properties of the proposed scheme for moving boundaries [126]. Although the moving piston problem is a rather simple inviscid problem, analytic expressions can be determined for the resulting shock wave that forms ahead of the piston and the rarefaction wave that is generated behind the piston and it provides a good test of the scheme's conservation properties. If the method accurately conserves mass, momentum, and energy at the moving interface, the predicted shock strength and speed must match the analytic expression. Poor agreement would indicate solution content may be lost at the moving surface.

The predicted non-dimensional pressure and density fields are shown in Figure 4.15 for a piston moving at Mach 2 into a quiescent gas. Comparison with the analytical results reveals that the proposed numerical scheme accurately predicts the shock position and strength on the compression-side of the piston, as well as the rarefaction wave solution behind the piston, indicating that the conservation properties of the finite-volume formulation are maintained when a moving interface is introduced.

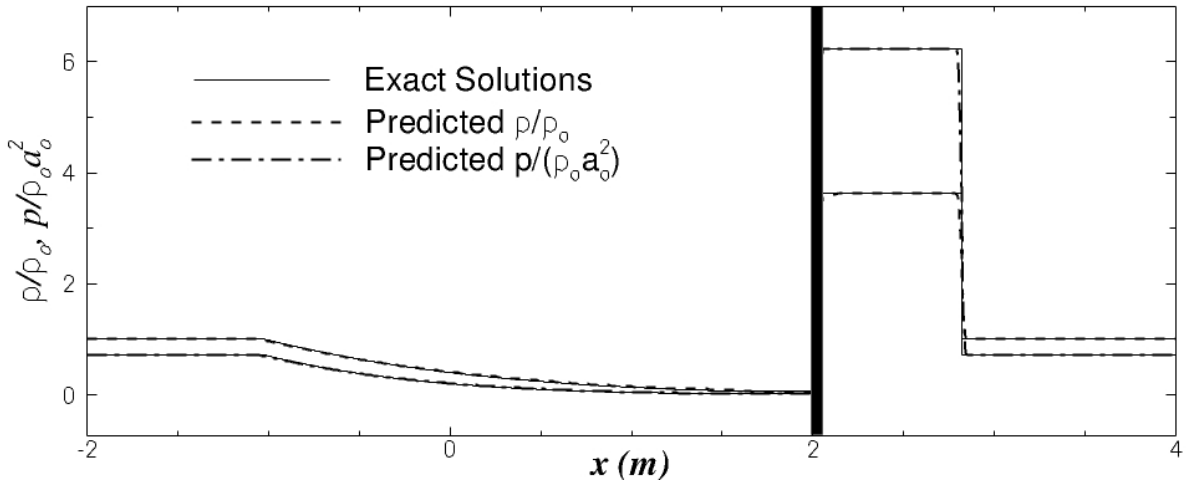


Figure 4.15: *Non-dimensional pressure and density ratios for the 1D moving piston at time 3 ms for a piston moving at Mach 2.*

4.2.2 Oscillating Aerofoil

Prediction of the fluid flow over an oscillating aerofoil is considered next and compared to the experimental measurements obtained by Landon [103] to provide further validation of the numerical method for moving embedded boundaries. The original experimental work was conducted to examine flow conditions experienced by helicopter blades. In particular, a NACA0012 aerofoil is undergoing an oscillatory motion about the quarter chord defined by $\alpha(t) = \alpha_0 + \alpha_m \sin(2\pi f t)$ where α is the angle of attack. The parameters $\alpha_0 = 0.016°$, $\alpha_m = 2.51°$, and $f = 62.5$ are the initial angle of attack, the amplitude of the oscillation, and the frequency of the oscillation. The freestream Mach number was $\mathrm{M} = 0.755$ and the flow Reynolds number based on the chord length was $\mathrm{Re} = 5.5 \times 10^6$. This particular set of experimental results have been used in previous studies to validate numerical algorithms for inviscid (e.g., Dubuc *et al.* [43] and Murman *et al.* [126]) and turbulent (e.g., Chassaing *et al.* [25]) flow.

In this work, inviscid flow over the oscillating NACA0012 aerofoil is computed where the aerofoil is embedded on a cylindrical grid. The outer radius of the grid was located 32 chord lengths from the quarter chord of the aerofoil. The initial mesh consisted of two solution blocks with 384 cells each, $(N_x, N_y) = (16, 24)$. To achieve an accurate representation of the aerofoil, five mesh refinements were performed in an area surrounding the embedded boundary. No further mesh refinement was used during the computation. The final mesh consisted of 248 blocks and 95,232 cells. A close-up of the mesh surrounding the embedded boundary at the initial angle of attack is shown in the top panel of Figure 4.16. The block boundaries are indicated by the thicker black lines. Note that a steady-state solution is achieved at each mesh level before refinement and before the pitching of the

aerofoil is started. The adjusted meshes at angles of attack corresponding to the extreme points in the oscillation are shown in the bottom two panels of Figure 4.16.

The pressure contours for the steady-state solution before the oscillation begins are shown in the top right panel of Figure 4.17. The contour levels range from 70-120 kPa. Shocks have formed on the upper and lower surfaces of the aerofoil. The initial angle of attack corresponds to a slight

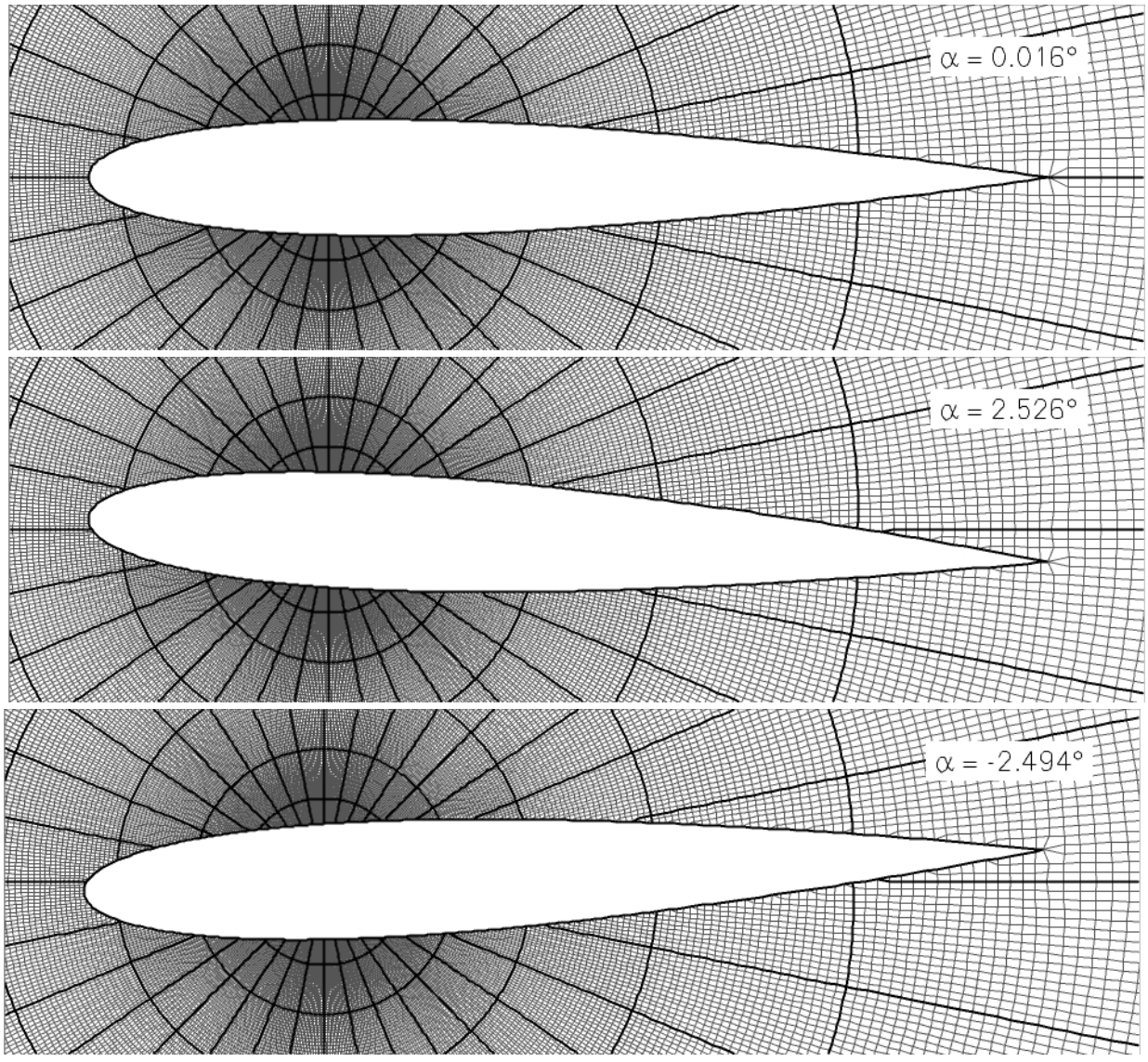


Figure 4.16: *Close-up of the adjusted mesh at the initial angle of attack (top frame),* $\alpha = 0.016°$*, the extreme pitch-up angle of attack (middle frame),* $\alpha = 2.526°$*, and the extreme pitch-down angle of attack (bottom frame),* $\alpha = -2.494°$*. Block boundaries are indicated by the thick black lines. Each block contains 384 cells,* $(N_x, N_y) = (16, 24)$*.*

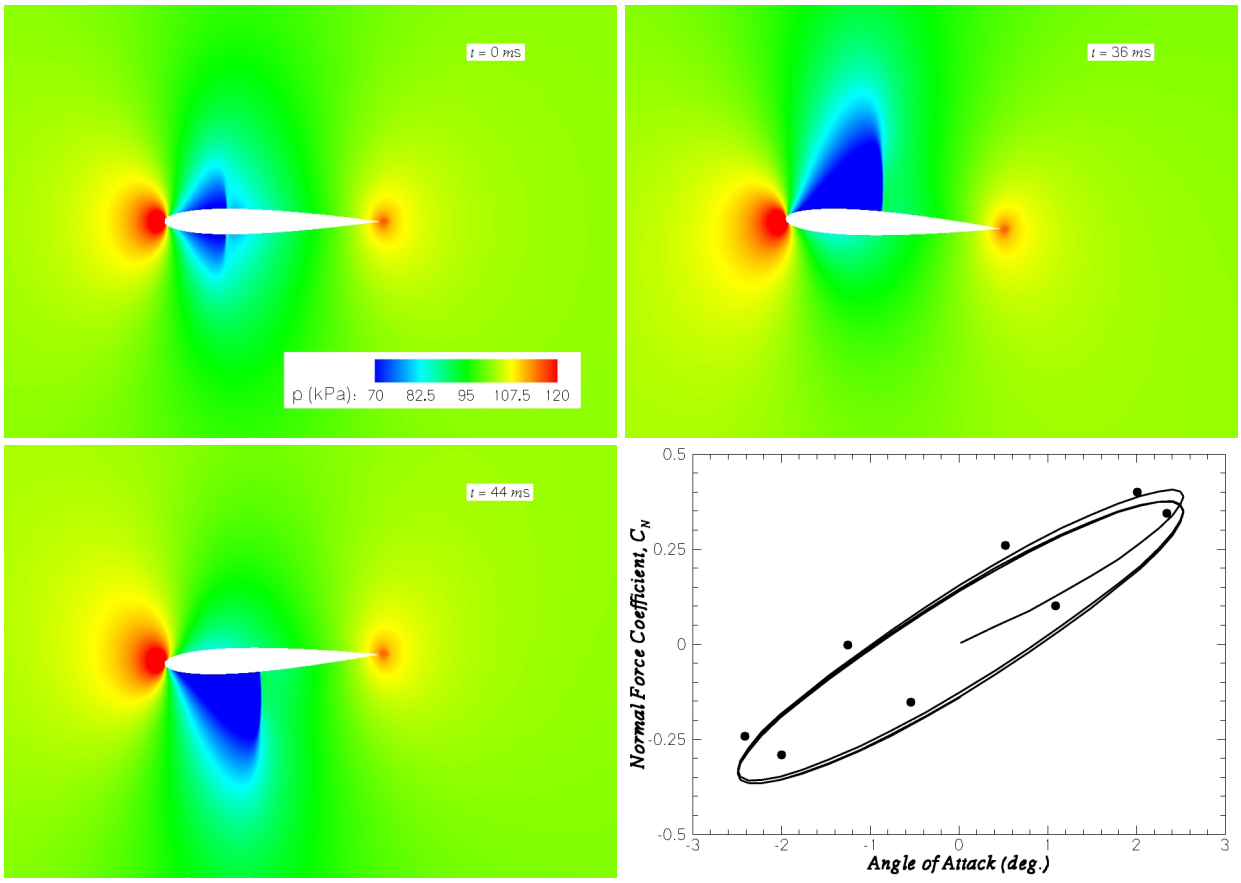


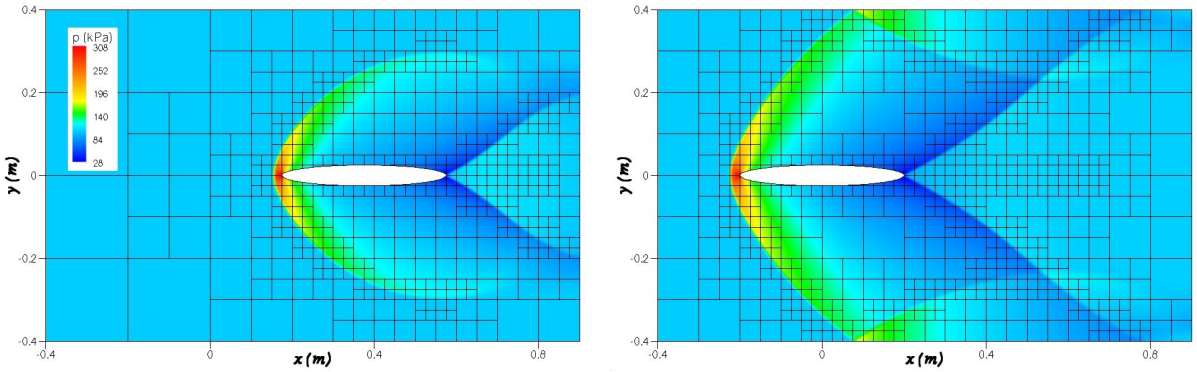
Figure 4.17: *Top two and bottom left panels: pressure contours, 70-120 kPa, for the steady-state solution before the oscillation and at two times during the third period of oscillation, 36 and 44 ms. Bottom right panel: comparison of the computed normal force coefficient for varying angle of attack with the experimental results of Landon [103].*

pitch-up and, therefore, the shock on the upper surface is slightly stronger. The oscillating motion of the aerofoil was started after the steady-state solution was achieved. Temporal discretization was achieved using MacCormack's predictor-corrector method with a CFL-number of 0.25. Approximately 300,000 time-steps were used for each period of the oscillation and a total time of 48 ms was simulated which corresponds to three periods of the aerofoil motion. As the aerofoil pitches up the shock on the upper surface strengthens and moves towards the trailing edge of the aerofoil. The shock on the lower surface diminishes in strength. At the peak of the pitch-up the shock on the lower surface has disappeared as can be seen in the upper right panel of Figure 4.17. Here, the pressure contours have been plotted at a time corresponding to 36 ms into the unsteady solution (first quarter of the third period). The converse of this situation occurs as the aerofoil pitches down. The shock on the upper surface begins to diminish and moves towards the leading edge

of the aerofoil, whereas a shock on the lower surfaces emerges from the leading edge, strengthens, and moves towards the trailing edge. At the extreme pitch-down point, the pressure contours are opposite to that of the peak pitch-up point as shown in the lower left panel of Figure 4.17. The pressure contours are plotted 44 ms into the unsteady solution (third quarter of the third period). It should be noted that during this process, the initial shock strengths are not re-established. For example, the shock on the upper surface is stronger than that on the lower surface at the half period (initial angle of attack, pitching down from maximum pitch-up). The reverse occurs at the end of each period. This leads to the hysteresis in the normal force coefficient as indicated in the lower right panel of Figure 4.17. In the figure, the normal force coefficient for varying angle of attack is plotted for the first three periods of the oscillation. After over-coming the initial response to the unsteady motion, the computed results are comparable to other numerical results [43, 126] and are in good agreement with Landon's experimental results. The predictions provide additional support for the implementation and use of the mesh adjustment algorithm for dynamic embedded boundaries.

4.2.3 Translating Ellipse

The computed inviscid flow produced by the prescribed motion of an ellipse undergoing impulsively initiated linear translations in the negative x-direction at Mach 1.5 in a channel with reflecting upper and lower boundaries is used as a final example to illustrate the effective combination of the moving boundary and AMR algorithms. Refinement was directed using the gradient of the density refinement criteria with division and coarsening thresholds of 0.15 and 0.05, respectively. As done



(a) (b)

Figure 4.18: *An ellipse translating at Mach 1.5 at (a) t = 0.735 ms (555 blocks, 55,500 cells, η = 0.783), and (b) t = 1.470 ms (802 blocks, 80,200 cells, η = 0.687). Pressure contours and mesh block boundaries are shown.*

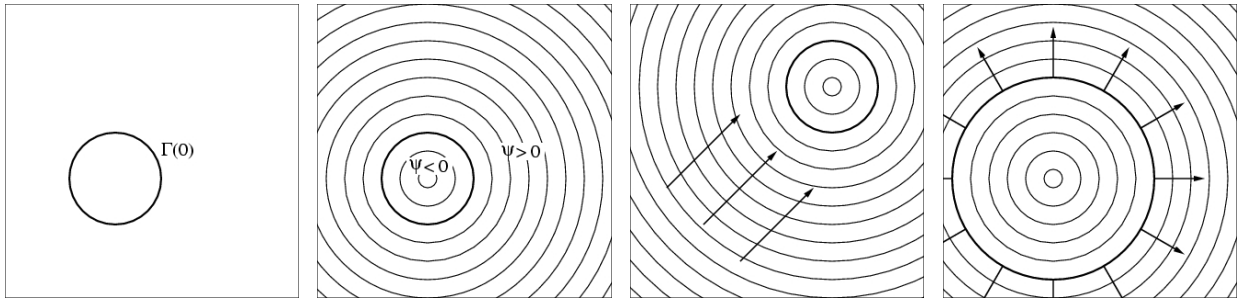
in the previous case, temporal discretization was achieved using MacCormack's predictor-corrector method with a CFL-number of 0.25. The computed pressure contours are shown in Figures 4.18(a) and 4.18(b) at two different times during the simulation. It is evident from the figures that the AMR scheme has successfully detected the bow shock which forms at the leading edge of the ellipse and the reflected and wake shock structures that form at solid boundaries and behind the translating ellipse.

4.3 Level Set Method for Evolving Boundaries

The level set method is an Eulerian front tracking method in which the location of the interface, $\Gamma(t)$, is implicitly defined by the zero contour of a function $\psi(\vec{x}(t), t)$ and is evolved by the solution of a Hamilton-Jacobi equation. The signed-distance function is used as the implicit function. Refer to Figures 4.19(a) and 4.19(b). This method was developed by Osher and Sethian [140]. Thorough explanations of the scheme are available in the textbooks by Sethian [175] and Osher and Fedkiw [138]. A brief description of the level set approach used herein is provided in the following subsections. In addition, the application of the block-based adaptive refinement scheme in conjunction with the level set method is outlined.

4.3.1 Level Set Equation

Evolution of the level set function is determined through the use of a scalar equation which can include motion due to a passive advection field, motion defined in the normal direction, and motion driven by the curvature of the interface. Curvature induced motion is not considered in this



(a) (b) (c) (d)

Figure 4.19: *Initial location of the interface is given in figure (a). Figures (b)–(d) show the initial signed-distance field, $\psi(x, y)$, passive advection of the scalar field in the indicated direction, and normal motion of the interface, respectively.*

work. The level set equation can be derived by simply taking the time derivative of the function $\psi(\vec{x}(t), t) = 0$ and applying the chain rule to arrive at

$$\frac{\mathrm{d}}{\mathrm{d}t}\psi(\vec{x}(t), t) = \frac{\partial \psi}{\partial t} + \vec{\nabla}\psi \cdot \frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = 0.$$

If the velocity is determined by a passive advection field, such as the one shown in Figure 4.19(c), then the velocity can be written as $\mathrm{d}\vec{x}/\mathrm{d}t = \vec{u}$. If, however, the motion of the front is known to be in the normal direction with a magnitude F, see Figure 4.19(d), then $\mathrm{d}\vec{x}/\mathrm{d}t \cdot \hat{n}_\psi$ where $\hat{n}_\psi = \vec{\nabla}\psi/|\vec{\nabla}\psi|$. Considering both types of motion, the level set equation is given by

$$\frac{\partial \psi}{\partial t} + F\,|\vec{\nabla}\psi| + \vec{u} \cdot \vec{\nabla}\psi = 0. \tag{4.1}$$

It was observed by Osher and Sethian that this equation is a Hamilton-Jacobi type equation [140] which can be written in a general form as

$$\frac{\partial \psi}{\partial t} + H(\vec{\nabla}\psi) = 0, \tag{4.2}$$

where the Hamiltonian, H, may be a function of both space and time. This is significant since in one spatial dimension the Hamilton-Jacobi equation can be related to a hyperbolic conservation law where $u = \partial\psi/\partial x$ is the conserved variable. This connection allowed Osher and Sethian to develop higher-order accurate numerical methods for solving Hamilton-Jacobi equations [140] of the form given in equation (4.2) in one space dimension. Dimension by dimension discretization is used for higher-dimensional problems. In this study, the fifth-order weighted essentially non-oscillatory (WENO) scheme developed by Liu *et al.* [114], Jiang and Shu [91], and Jiang and Peng [90] is used to determine the discrete approximations to the gradients for the solution of the level set equation and the Eikonal and scalar extension equations discussed next. These WENO finite-difference schemes are described in Appendix B. Temporal discretization is achieved using MacCormack's predictor-corrector or the fourth-order Runge-Kutta time-marching methods.

It should be noted that the level set equation, equation (4.1), is not in conservation form and is not a conservation statement for any relevant physical quantity. Therefore, the numerical solution of this equation in combination with the mesh adjustment scheme for evolving embedded boundaries is not strictly conservative. Although some attempts have been made by researchers to address the non-conservative nature of the level set equation [184, 186], these variants were not pursued as part of this work. Instead, the fifth-order WENO scheme as well as the application of the block-based AMR algorithm, refer to Section 4.3.4, are relied on to address this issue by allowing the errors associated with conservation violation to be reduced and controlled to acceptably small levels.

4.3.2 Eikonal Equation

Evolution of the level set equation, equation (4.1), will often cause the implicit function, ψ, to develop features, such as steep gradients, such that is no longer a valid signed-distance function. Note that it is very advantageous to maintain a signed-distance function since the spatial discretization schemes used are more accurate for smooth functions [138]. Re-distancing of the level set function can be accomplished by iteratively solving the Eikonal equation, $F|\vec{\nabla}\psi| = 1$, with a unit speed function, $F = 1$. In this work, the Eikonal equation is reformulated into a Hamilton-Jacobi equation, given by

$$\frac{\partial \psi}{\partial \tau} + S(\psi)(|\vec{\nabla}\psi| - 1) = 0, \tag{4.3}$$

as proposed by Sussman *et al.* [187] such that the discretization methods discussed above can be directly applied. The function is re-distanced when $\tau \to \infty$. The function, $S(\psi)$, is a sign function that allows for both the negative and positive sides of the distance function to be re-distanced at once. In this work, the sign function is approximated by

$$S(\psi) = \frac{\psi}{\sqrt{\psi^2 + |\vec{\nabla}\psi|^2(\triangle x)^2}}, \tag{4.4}$$

as proposed by Peng *et al.* [149]. Although more expensive to formulate, it was found that this provided a better estimate of the sign function than the original smeared version, $\psi_0/\sqrt{\psi_0^2 + (\triangle x)^2}$, used by Sussman *et al.* [187].

An initial estimate of the signed-distance function can be determined geometrically. Then the iterative solution of the Eikonal equation described above can be used to improve the initial signed-distance function.

4.3.3 Scalar Extension Equation

The speed of the interface in the normal direction is typically defined along the interface itself. However, before the level set equation, equation 4.1, is solved this speed function must be extended throughout the computational domain. Since the speed function must always point in the normal direction of the level set function, it follows that the speed function must satisfy

$$\vec{\nabla}\psi \cdot \hat{n}_\psi = 0, \tag{4.5}$$

which states that the angle between the gradient of the speed function and the normal direction of the level set function must be zero. As for the Eikonal equation, the scalar extension equation

can be reformulated as a Hamilton-Jacobi equation (or in this case it is simply a scalar hyperbolic equation) given by

$$\frac{\partial F}{\partial \tau} + S(\psi)\vec{\nabla}\psi \cdot \vec{\nabla}F = 0, \tag{4.6}$$

which can again be solved using the same discretization methods. This approach is used herein. It was proposed by Zhao *et al.* [214] and first implemented by Peng *et al.* [149]. Equation (4.5) is satisfied in the limit of $\tau \to \infty$.

4.3.4 Level Set Method with Adaptive Mesh Refinement

The use of AMR can significantly improve solutions provided by the level set method, and Milne [123] (see also Sethian [175]) has applied cell-based AMR methods [16] to improve mesh resolution in areas of high-curvature of the zero-level set. In this work, the block-based AMR scheme outlined in Chapter 3 is applied to the solution of the level set, Eikonal, and scalar extension equations. The curvature of the zero level set function is also used here as the refinement criteria. The curvature of the level set function, κ, is computed by

$$\epsilon \propto |\kappa| = \left| \vec{\nabla} \cdot \left(\frac{\vec{\nabla}\phi}{|\vec{\nabla}\phi|} \right) \right|. \tag{4.7}$$

User defined thresholds are used to determine the levels of high and low curvature in which block division and coarsening are to be pursued.

4.3.5 Zalesak's Disk

A popular test case for assessing the performance of front tracking algorithms is the evolution of a rigid slotted disk in a rotating flow, also known as Zalesak's disk [185, 47]. The problem configuration used here is the same as that studied by Sussman *et al.* [185]. The rotating two-dimensional velocity field is taken to be $(u, v) = (\pi/314)(50 - y, x - 50)$. The initial configuration of the disk is shown in the left-most panel of Figure 4.20 where the disk boundary is given by the zero contour shown by the thick black line. Initially, the mesh includes three levels of refinement with 250 blocks and 100,000 cells, $(N_x, N_y) = (20, 20)$. The subsequent three panels of Figure 4.20 show the computed position of the disk after an incremental rotation of 120° about the origin. These meshes contain 232, 229, and 250 blocks, respectively. The level set equation and the Eikonal equation are both solved using the fifth-order weighted essentially non-oscillatory spatial discretization procedure as outlined by Osher and Fedkiw [139] and a second-order predictor-corrector time-marching method. Solution of the Eikonal equation is required at every time-step to maintain an accurate

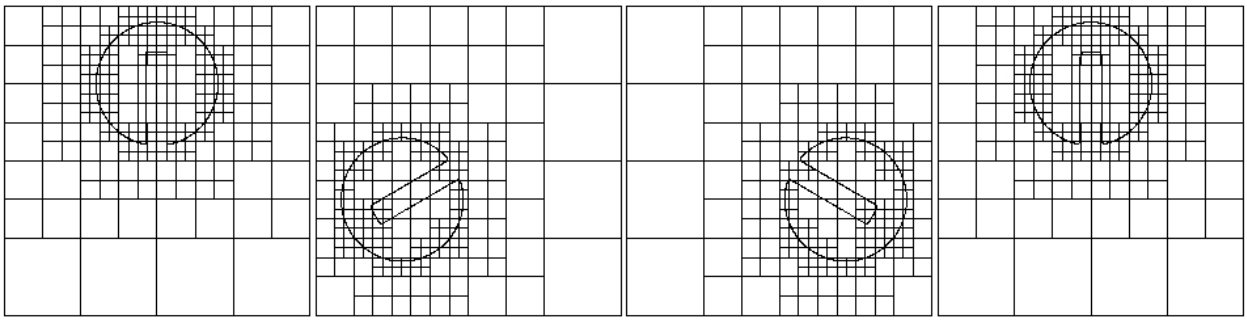


Figure 4.20: *Zalesak's disk, from left to right: initial configuration (250 blocks, 100,000 cells,* $(N_x, N_y) = (20, 20)$*,* $\eta = 0.756$*), after* 120° *rotation (232 blocks, 92,800 cells,* $\eta = 0.773$*), after* 240° *rotation (229 blocks, 91,600 cells,* $\eta = 0.776$*), and* 360° *rotation (250 blocks, 100,000 cells,* $\eta = 0.756$*). The zero contour is shown by the thick black line and the block boundaries are given by the this black lines.*

and valid signed-distance function. It can be seen that the level set method coupled with the block-based AMR scheme is capable of maintaining an accurate definition of the disk throughout the rotation. Additional refinement could be employed to further reduce the diffusion of the disk as it rotates.

4.3.6 Constant Volume Bomb

The combustion of a semicircular charge of solid propellant in a closed vessel is simulated to demonstrate the coordination of the finite volume scheme, the AMR method, the mesh adjustment scheme, and the level set method for predicting fluid flows with an embedded boundary evolving due to a physical process. The initial adjusted multi-block mesh with five levels of refinement (130 blocks, 33,280 cells, $\eta = 0.87$) in the left panel of Figure 4.21 (mesh lines not shown). The right panel of this figure shows the initial multi-block mesh used for the solution of the level set problem. The level set method is computed on a separate overlapping Cartesian mesh which is adapted independently from the body-fitted mesh. Four levels of refinement are used here (58 blocks, 14,848 cells, $\eta = 0.77$). However, the mesh adjustment scheme requires an explicit representation of the embedded boundary at its current location. Therefore, a contour tracing algorithm is required to provide an approximate representation of the zero level set. The piecewise linear contour tracing algorithm proposed by Dobkin *et al.* [42] is used for this purpose and is described in Appendix C of the thesis.

The burning rate of the propellant and the state quantities of the combustion products is determined using a pressure dependent empirical relation. Application of the burning surface

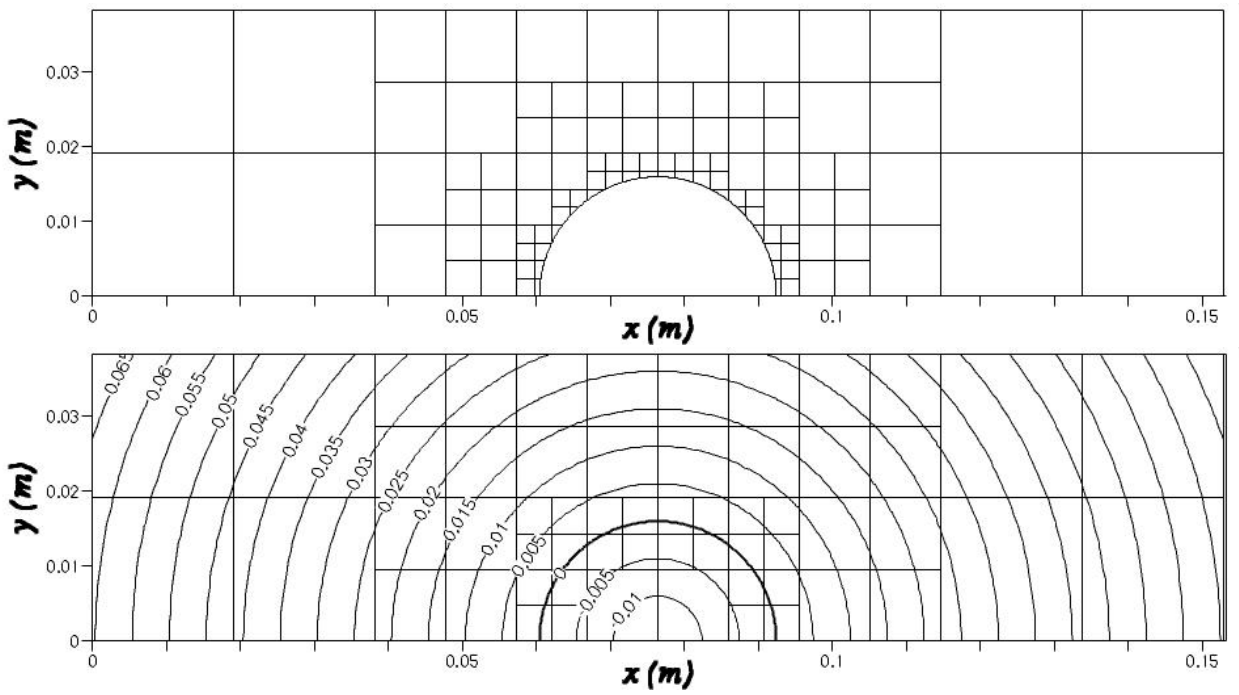


Figure 4.21: *Initial configuration for the combustion of a semicircular propellant charge in a closed vessel. The initial adjusted multi-block body-fitted mesh is shown in the top panel (5 levels of refinement, 130 blocks, 33,280 cells, $\eta = 0.87$). The initial multi-block Cartesian mesh for the level set method is shown in the bottom panel (4 levels of refinement, 130 blocks, 33,280 cells, $\eta = 0.87$). Level set contours are shown and the zero level set is highlighted by the thick black line.*

boundary condition is described in Appendix D. A composite propellant, AP-HTPB, is considered here. Refer to table 5.3 for the characteristics of the propellant in solid and gaseous states. The burning rate increases as the pressure in the chamber increases. It should be noted that the pressure is non-uniform over the surface and, therefore, the burning rate is also non-uniform. Consequently, the scalar extension equation, equation 4.6, is solved iteratively before the interface is evolved using the level set method. The predicted solution at 16 ms is shown in Figure 4.22 for a laminar flow ($\mathrm{Re} \approx 6,000$). The predictor-corrector time-marching method outlined in Section 3.2 was used in this simulation with a CFL number of 0.25. Initially, the chamber was set at a pressure of 1 MPa and the pressure in the chamber has reached over 2.3 MPa at 16 ms. The multi-block mesh (310 blocks, 79,360 cells, $\eta = 0.70$) and the predicted temperature contours are given in the left panel of Figure 4.22. The level set contours and the multi-block mesh (28 blocks, 7,168 cells, $\eta = 0.89$) are shown in the right frame. It can be seen that at 16 ms the circular grain has regressed significantly and the propellant is slightly more than half burnt.

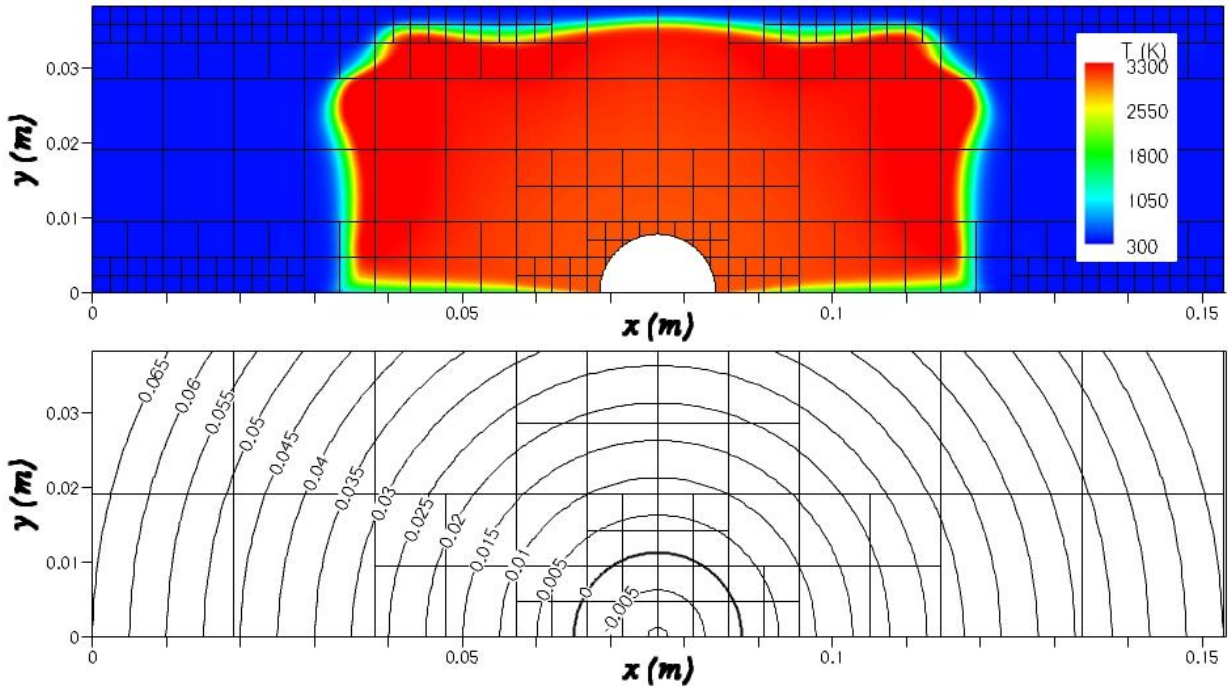


Figure 4.22: *Predicted temperature contours and the adjusted multi-block body-fitted mesh are shown in the top panel (5 levels of refinement, 310 blocks, 79,360 cells, $\eta = 0.70$) for the combustion of a semicircular propellant charge in a closed vessel 16 ms after ignition. Level set contours and multi-block Cartesian mesh are shown in the bottom panel (4 levels of refinement, 28 blocks, 7,168 cells, $\eta = 0.89$). The zero level set is highlighted by the thick black line.*

Chapter 5

Solid Propellant Rocket Motor Simulations

5.1 Rocket Motor Configurations

Predictions of solid propellant rocket motor operation for configurations typical of a CRV-7 rocket system are now described to demonstrate the capability of the proposed numerical algorithm. The CRV-7 rocket system represents the base target case of primary interest for the long-term research objectives. The CRV-7 rocket system contains a non-aluminized composite propellant composed of 80% oxidizer (ammonium perchlorate, AP) and 20% fuel (hydroxyl terminated polybutadiene, HTPB). As is typical in some tactical rocketry, inert aluminium oxide particles account for 3% of the solid propellant by mass, $\alpha_s = 0.03$. Therefore, the modelling of burning particles, particle-particle collision processes (such as agglomeration), and the transport of smoke are not required for the present work. The burning of the solid propellant and production of the propellant gases at the surface of the propellant grain is specified through the solution of a Riemann problem which is described in Appendix D. The treatment of the burning propellant boundary is very similar in spirit to the methods proposed by Gottlieb and Groth [61] for imposing boundary data at a variety of flow boundaries based on the solution of Riemann problems.

Results for four different simulations are presented and discussed. An inviscid gas-particle flow in a cylindrical grain rocket motor is described in the first section. The multi-velocity formulation is used for the particle-phase. The second case involves a laminar gas-only flow with an evolving cylindrical grain rocket motor. Two turbulent rocket motor flow configurations are discussed in the last section. In the first of these two cases, turbulent flow in a nozzleless rocket motor is predicted and compared to the experiments performed by Traineau *et al.* [193]. Here, air is injected directly into the rocket chamber at a specified mass flow rate. There is no combusting nor burning interface for this cold-flow problem. The second turbulent flow case includes the combustion of solid propellant. Results are given for this hot-flow case with and without particles to study the influence of the particle phase on the intensity of the turbulence.

5.2 Inviscid Gas-Particle Flow with a Stationary Combustion Interface

Rocket motor predictions using the multi-velocity formulation are presented in Figures 5.1–5.4 for an inviscid gas-particle flow in a cylindrical grain rocket motor with a 400 mm chamber length, a 40 mm internal radius, a nozzle throat radius of 10 mm, and an internal port radius of 20 mm. The propellant and particle characteristics are given in Tables 5.1 and 5.2, respectively. The solution procedure involved time-stepping the conservation equations to steady-state then refining the mesh and computing again. This is repeated for four mesh refinements. The initial and final grid configurations are shown in Figure 5.1 in the upper and lower panel of the figure, respectively. Note that only the nozzle end of the rocket is shown. The burning of the solid propellant leads to a head end pressure in excess of 2.8 MPa and produces sonic flow conditions at the nozzle throat and supersonic outflow in the rocket nozzle with the Mach number approaching 3.35. The particle-phase concentration contours through the converging-diverging nozzle computed using the multi-velocity formulation are depicted in Figure 5.2 for the initial and final meshes in the upper and lower portions of the figures, respectively. A comparison of the propellant gas and inert particle axial velocity components is shown in Figure 5.3. In addition, comparison of the computed centreline axial profiles for the density, pressure, gas and particle phase axial velocity component, and the gas and particle phase temperatures with the results of a one-dimensional analysis [64, 160] is given in Figure 5.4. The density and pressure profiles are in excellent agreement, both qualitatively and quantitatively, with the one-dimensional results. Much of the differences between the predictions

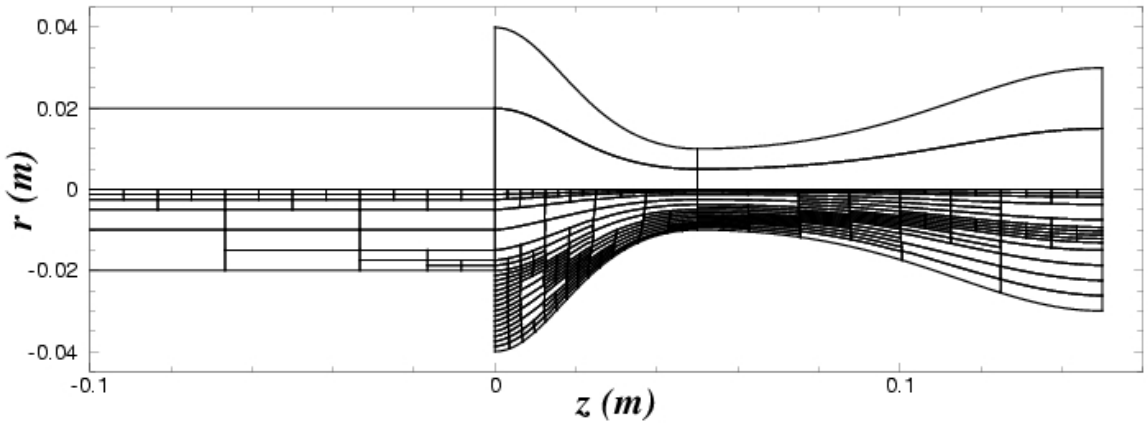


Figure 5.1: *Multi-block grid structure for a cylindrical grain rocket motor. The initial grid (upper panel) contains 7 blocks and 2,688 cells (24×16) and the final grid (lower panel) after four refinements contains 556 blocks and 213,504 cells. Entire rocket not shown.*

Table 5.1: *Propellant characteristics for AP-HTPB.*

Propellant density (ρ_s)	1740.0 kg/m^3
Propellant specific heat (c_s)	1510 J/kg K
Propellant flame temperature (T_f)	3060 K
Propellant surface temperature (T_s)	1130 K
Propellant burning rate ($r_{bs} = \beta p^n$)	$0.5[p(\text{kPa})]^{0.33}$ mm/s
Gas specific heat (c_p)	1845 J/kg K
Specific gas constant (R)	318 J/kg K
Gas thermal conductivity (κ)	0.184 W/m K
Gas absolute viscosity (μ)	8.19 $\times 10^{-5}$ kg/m s
Gas ratio of specific heats (γ)	1.21

can be attributed to the two-dimensional behaviour of the particle-phase, a discussion of which now follows.

It is clear from Figures 5.3 and 5.4(c) that there is a particle-phase velocity lag relative to the gas-phase velocity after the rapid acceleration through the nozzle. The particles are unable to expand in the nozzle due to their relative high mass, resulting in a region of zero particle concentration in the diverging section of the nozzle as shown in Figure 5.2. Before the nozzle throat, the particles are unable to negotiate the curved path of the converging section of the nozzle and impact the

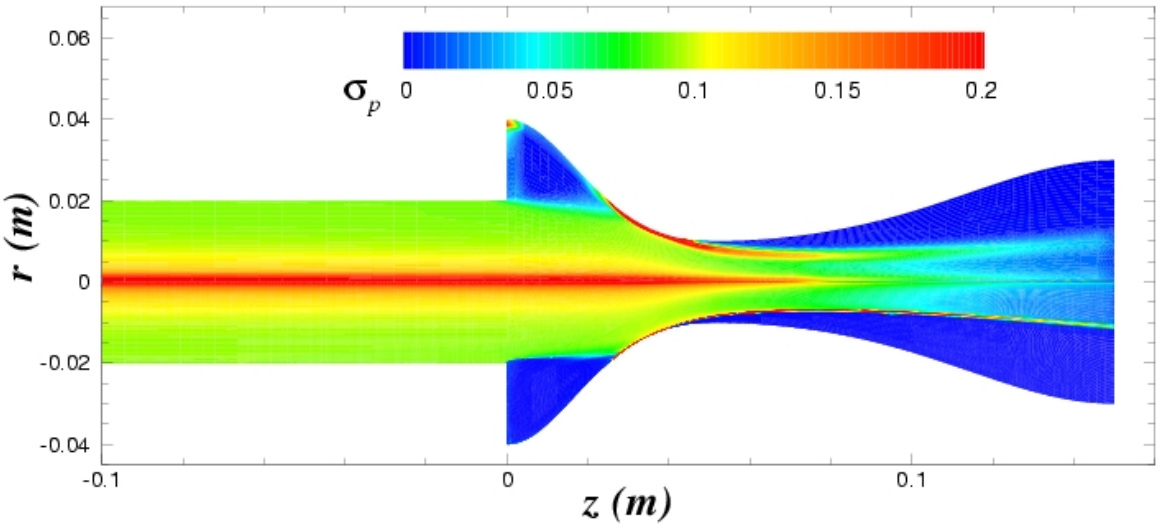


Figure 5.2: *Predicted particle-phase concentration contours for a cylindrical grain rocket motor calculated on the initial grid (upper panel) and the grid after four mesh refinements (lower panel).*

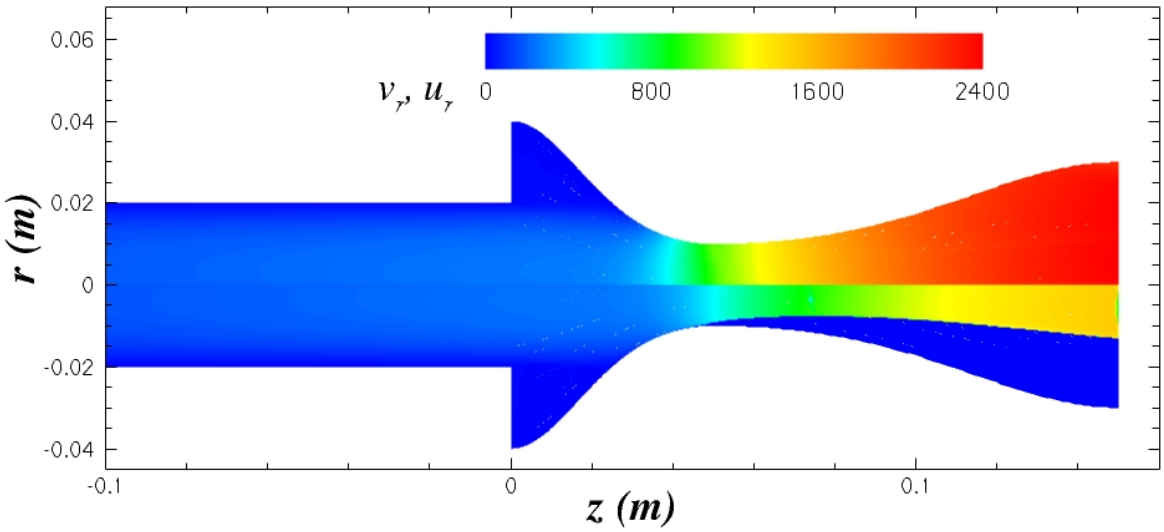


Figure 5.3: *Axial velocity component for the propellant gas-phase (upper panel) and inert particle-phase (lower panel) for a cylindrical grain rocket motor.*

nozzle at a high velocity and can lead to erosion of the nozzle material. The particles become entrained with the gas and are pulled through the nozzle throat into the diverging section where a dense stream of particles can be seen. Similar results are found when using a single-velocity formulation for the particle-phase, however, with artificially higher peak concentrations predicted at the nozzle and the axis of symmetry due to issues associated with the degeneracy of the Eulerian formulation. The computations of Vuillot *et al.* [207] and York *et al.* [213] show similar patterns in the particle concentration contours. Figure 5.4(d) also indicates a temperature slip between the two phases in the diverging section of the nozzle. The gas-phase rapidly cools down as it expands and lowers the particle-phase temperature through the transfer of heat. Finally, it can be seen in Figure 5.2 that the application of the block-based mesh refinement algorithm has successfully clustered computational blocks at areas of interest. In particular, accurate resolution of the interface between zero and non-zero particle concentrations has been achieved.

Table 5.2: *Inert* Al_2O_3 *particle characteristics.*

Average particle diameter (d_p)	10 μm
Particle solid density (ρ_p)	2700 kg/m^3
Particle specific heat (c_m)	900 J/kg K

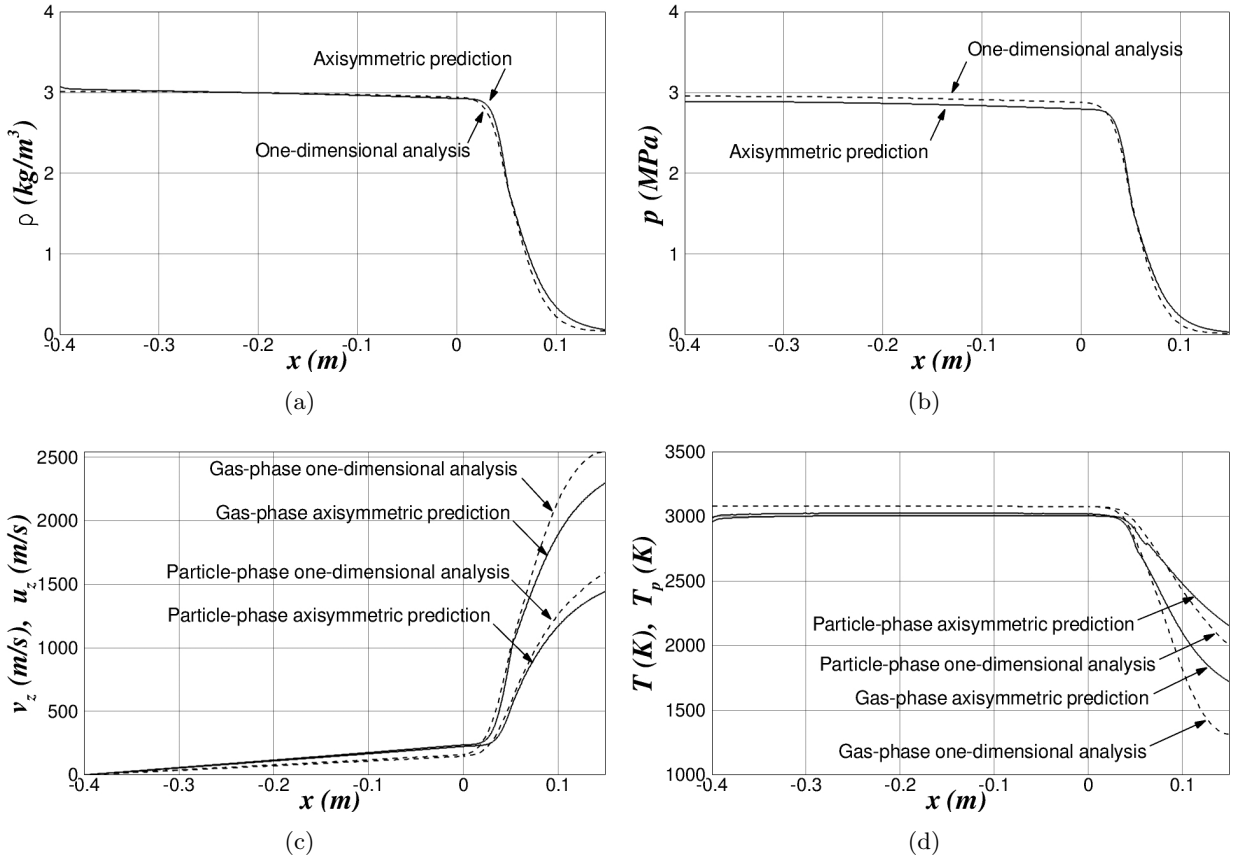


Figure 5.4: *Comparison of computed centreline axial profiles with a one-dimensional analysis for (a) density, (b) pressure, (c) axial velocity component of the gas and particle phases, and (d) temperature of the gas and particle phases.*

5.3 Laminar Flow with an Evolving Combustion Interface

Prediction of an internal solid propellant rocket motor flow is now described to demonstrate the viability and capability of the proposed scheme for computing two-dimensional axisymmetric laminar flow with an evolving embedded boundary. Here, a faster burning propellant was used than the previous case. The characteristics of the solid propellant and the combustion products are summarized in Table 5.3.

As for the constant volume bomb example, Section 4.3.6, the level set method is used to determine the evolution of the propellant grain based on the pressure-dependent burning rate. It is computed on a separate overlapping Cartesian mesh which is adapted independently from the body-fitted mesh and the piecewise linear contour tracing algorithm of Dobkin *et al.* [42] is used to provide an explicit representation of the embedded boundary for the mesh adjustment algorithm.

Table 5.3: *Propellant characteristics for a fast burning AP-HTPB.*

Propellant density (ρ_s)	17.4 kg/m^3
Propellant specific heat (c_s)	1510 J/kg K
Propellant flame temperature (T_f)	3060 K
Propellant surface temperature (T_s)	1130 K
Propellant burning rate ($r_{bs} = \beta p^n$)	$16.18[p(\text{kPa})]^{0.50}$ mm/s
Gas specific heat (c_p)	1845 J/kg K
Specific gas constant (R)	318 J/kg K
Gas thermal conductivity (κ)	0.184 W/m K
Gas absolute viscosity (μ)	8.19 $\times 10^{-5}$ kg/m s
Gas ratio of specific heats (γ)	1.21

The cost of evolving the location of the combustion interface, tracing the location of the zero level set, and subsequently performing the mesh adjustment algorithm according to the new location of the embedded boundary was found to require approximately one-and-a-half to two times the computational work associated with the application of one step of the explicit time-marching scheme for this case.

The predicted results are presented in Figures 5.5–5.7 for a cylindrical grain rocket motor with a 208.75 mm chamber length, 31.75 mm internal radius, a nozzle throat radius of 10.15 mm, an initial internal port radius of 20 mm, and an initial distance from the throat to the propellant grain of 65 mm. The initial geometry of the rocket motor can be seen in the top half of Figure 5.5. Note that the entire length of the combustion chamber is not shown. The overlapping Cartesian domain used for the level set problem is shown in the lower half of Figure 5.5. The location of combustion interface is given by the zero level set contour. Five levels of adaptive mesh refinement are used on the initial level set solution (56 blocks, 16,800 cells, $(N_x, N_y) = (30, 10)$, and $\eta = 0.985$). The steady state solution on the initial geometry is computed before the propellant grain is allowed to evolve. Six levels of adaptive mesh refinement are used during the steady state solution (168 blocks, 64,512 cells, $(N_x, N_y) = (24, 16)$, and $\eta = 0.945$). The steady state pressure and Mach number contours are shown in the upper and lower panels of Figure 5.6, respectively. The burning of the solid propellant leads to a head end pressure in excess of 1.4 MPa and produces sonic flow conditions at the nozzle throat and supersonic outflows in the rocket nozzle with Mach numbers over 2.5. The streaklines are also depicted in the lower half of Figure 5.6. The inset contains the velocity vectors which clearly indicate a low speed recirculation zone near the beginning of the converging section of the nozzle.

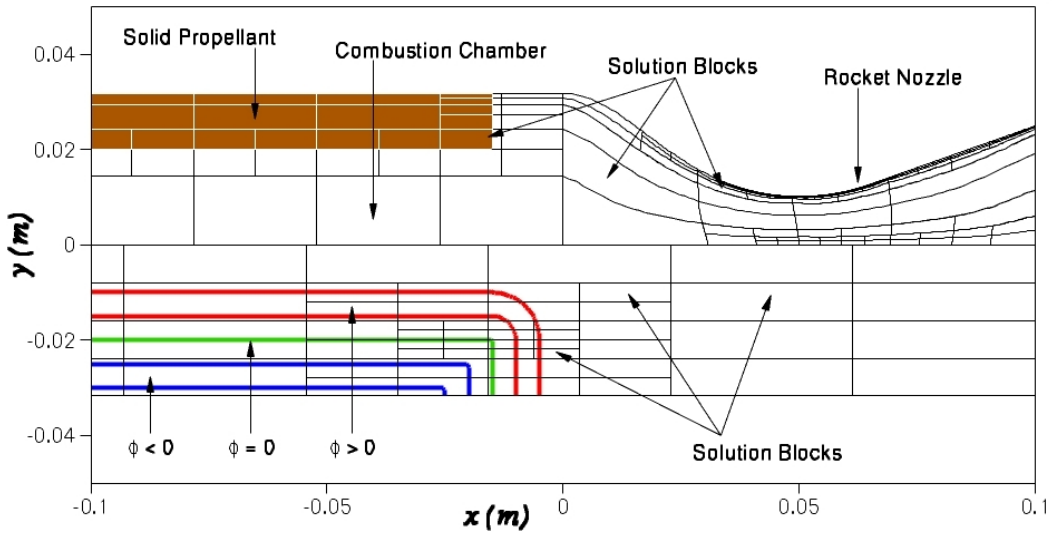


Figure 5.5: *Block structure and configuration for the prediction of an internal rocket motor flow for the fluid (top panel) and level set (lower panel) domains before the evolution of the combustion interface. Five levels of adaptive mesh refinement are used on the initial level set solution (56 blocks, 16,800 cells,* $(N_x, N_y) = (30, 10)$*, and* $\eta = 0.985$*). Six levels of adaptive mesh refinement is used during the steady state solution (168 blocks, 64,512 cells,* $(N_x, N_y) = (24, 16)$*, and* $\eta = 0.945$*).*

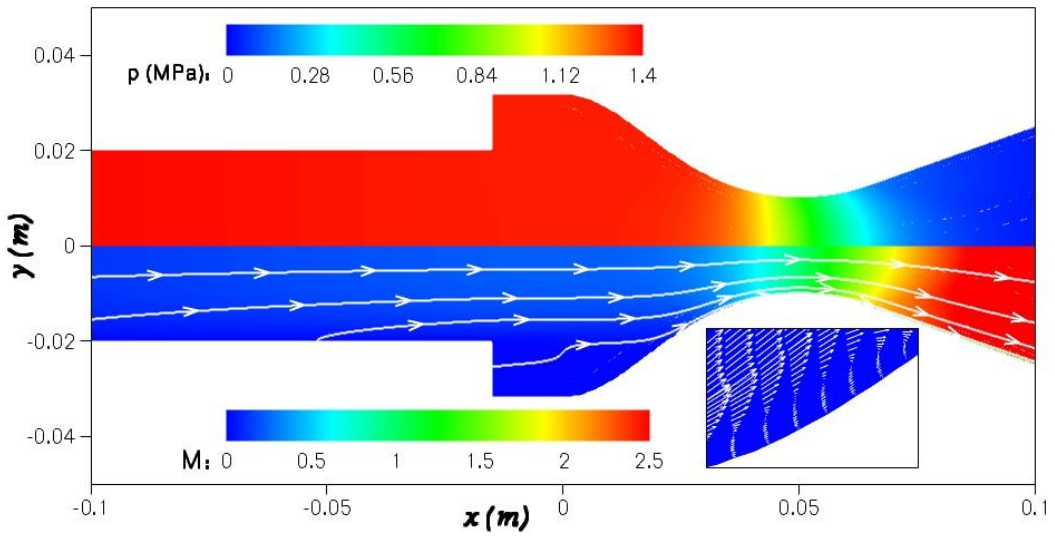


Figure 5.6: *Predicted steady-state pressure (top panel) and Mach number (bottom panel) distribution for the initial configuration of the cylindrical grain rocket motor. The streaklines are also shown in the bottom panel. The inset contains the velocity vectors which indicate a low speed recirculation zone near the beginning of the converging section of the nozzle.*

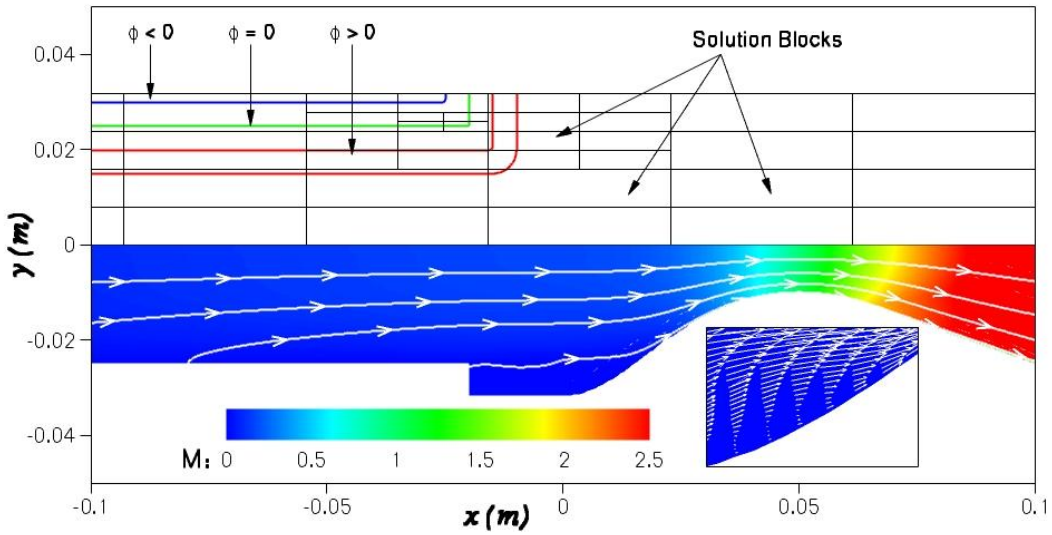


Figure 5.7: *Predicted cylindrical grain rocket motor solution 8 ms after the combustion interface is allowed to evolve. The level set solution and block structure (41 blocks, 12,300 cells, and* $\eta = 0.92$*) are shown in the top panel. The Mach number distribution and streaklines are shown in the bottom panel. The adjusted body-fitted mesh includes 189 blocks and 72,576 cells (*$\eta = 0.938$*). The inset contains the velocity vectors which indicates that the low speed recirculation zone near the beginning of the converging section of the nozzle has decreased in size.*

The predicted solutions 8 ms after the combustion interface is allowed to evolve is shown in Figure 5.7. Temporal discretization was achieved using the fourth-order Runge-Kutta time-marching method with a CFL-number of 0.95. Nearly four-million time-steps were required to evolve the combustion interface for 8 ms. Unlike the earlier simulations with moving embedded boundaries in which the location of the boundary and the mesh was readjusted every time-step, the very slow motion of the combustion interface as compared with the acoustic wave propagation speeds requires a much less frequent update of the geometry. For this case the location of the combustion interface was only updated every 2500 time-steps. The level set solution and block structure are given in the upper panel of this Figure. At this time, the Cartesian mesh for the level set problem contains 41 blocks and 12,300 cells. The Mach number distribution and streaklines are given in the lower panel. The adjusted body-fitted mesh contains 189 blocks and 72,576 cells (the block structure is not shown in the figure). It can be seen that just over 40% of the propellant has been burnt at this time. The low-speed recirculation region shown in the inset has decreased in size and strength since the sudden expansion after the propellant grain has decreased in height. As a result, the flow from the main cavity is able to interact more directly with the converging section of the nozzle.

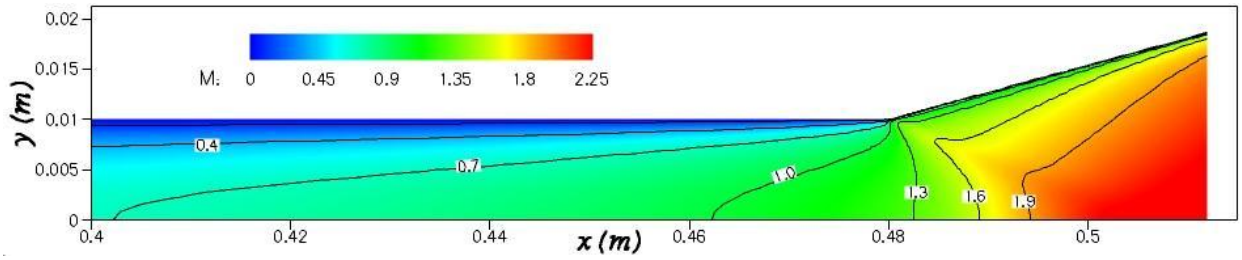


Figure 5.8: *Predicted Mach number contours for the nozzleless rocket motor. Entire rocket motor not shown.*

5.4 Turbulent Rocket Motor Core Flows

5.4.1 Nozzleless Rocket Motor

The experimental measurements taken by Traineau *et al.* [193] for a turbulent flow in a nozzleless rocket motor can be used to validate the proposed algorithm for computing turbulent rocket motor core flows. In their experiment, air is injected from a reservoir directly into a planar rocket chamber at a constant mass flow rate. The length and height of the chamber are 48 cm and 2 cm, respectively. A diverging expansion section was included at the open end of the duct and has a length of 3.2 cm and a divergence angle of 15°. The total temperature and pressure in the reservoir are 260 K and 3 atm, respectively. The injection rate was controlled to be uniform along the entire chamber wall and the mass flow rate was designed to be 13 kg/m^{2}s. Traineau *et al.* [193] reported that a high level of surface generated turbulence was experienced at the injection wall which had a porosity of 50μm. Accordingly, the surface roughness parameter, σ_v, and the turbulence length-scale, l_w, at the mass injection boundary were set to 1.0 and 25μm, respectively, for this calculation. Furthermore, the turbulent kinetic energy and specific dissipation rate were integrated directly to the wall at this boundary with surface values specified by equation 2.72.

The predicted results are presented in Figures 5.8–5.10 and were computed on a uniformly refined mesh with 32 solution blocks and 24,576 cells, $(N_x, N_y) = (24, 32)$. The mesh was stretched towards the solid and injection boundaries such that the largest cell aspect ratio was 21,000. The predicted Mach contours for the nozzleless rocket motor are shown in Figure 5.8 for a region of flow that includes the expansion area. It can be seen that an exit Mach number over 2.25 is obtained and that the sonic speed contour occurs well within the duct. Similar Mach number distributions were obtained in the inviscid flow computations of Traineau *et al.* [193] and the turbulent flow predictions of Sabnis *et al.* [167]. The predicted static pressure profile along the centreline of the rocket and radial profiles of the axial velocity component at various stations in the rocket chamber are plotted

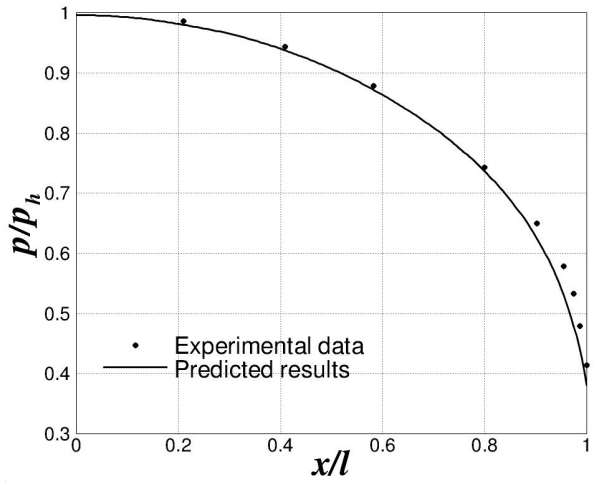


Figure 5.9: *Comparison of the predicted axial pressure along the centreline of the rocket with the experimental results of Traineau et al. [193].*

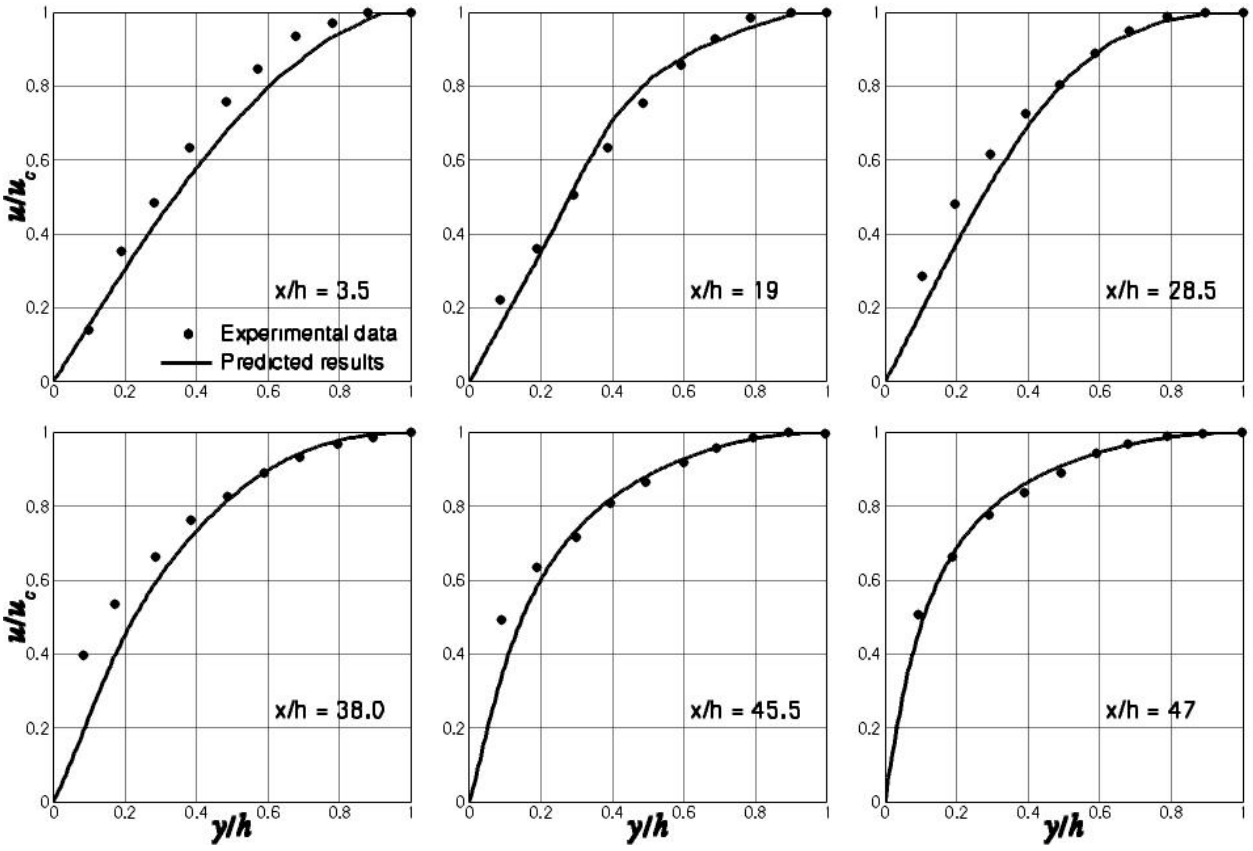


Figure 5.10: *Comparison of the predicted normalized axial velocity profiles at various stations in the rocket chamber with the experimental results of Traineau et al. [193].*

against the the experimental data of Traineau *et al.* [193] in Figures 5.9 and 5.10, respectively. The pressure and axial velocity profiles are in excellent agreement with the experimental results. These predictions provide additional support for the implementation and use of the k-ω turbulence model for the prediction of turbulent rocket motor core flows.

5.4.2 Turbulent Gas-Particle Flow

A final set of numerical solutions are now described to demonstrate the viability and capability of the proposed scheme for computing two-dimensional axisymmetric turbulent gas-particle SRM core flows. The rocket used here has chamber length of 208.75 mm, a chamber radius of 31.75 mm, and a throat radius of 10.15 mm. The propellant and particle characteristics are given in Tables 5.1 and 5.2, respectively.

No-slip and isothermal boundary (T = 298 K) conditions were applied at the solid walls at the head-end and along the nozzle. The turbulence boundary condition (wall functions or direct integration) are applied according to an automated switching process as dictated by the available cell resolution [55]. The combustion products of the solid propellant are injected into the flow directly from the chamber wall. As discussed in Chapter 2, the kinetic energy and specific dissipation rate are directly integrated to the burning surface where their values where the surface roughness parameter and the length scale of the turbulence are specified by $\sigma_v = 0.035$ and $l_{bs} = 200$ μm. These are the same values used by Cai *et al.* [23].

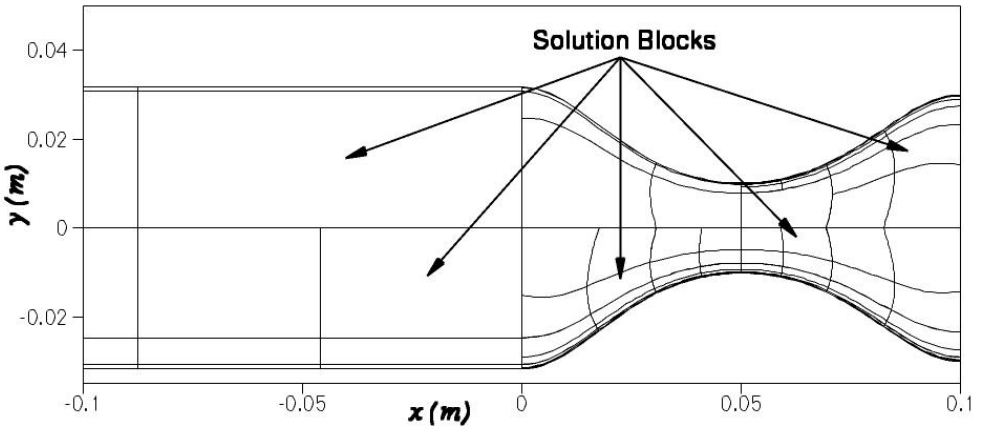


Figure 5.11: *Multi-block grid structure for a cylindrical grain rocket motor after four levels of mesh refinement. The top panel contains the multi-block mesh for the gas-only flow (40 blocks, 11,520 cells, $\eta = 0.844$). The bottom panel contains the multi-block mesh for the gas-particle flow (82 blocks, 23,616 cells, $\eta = 0.680$).*

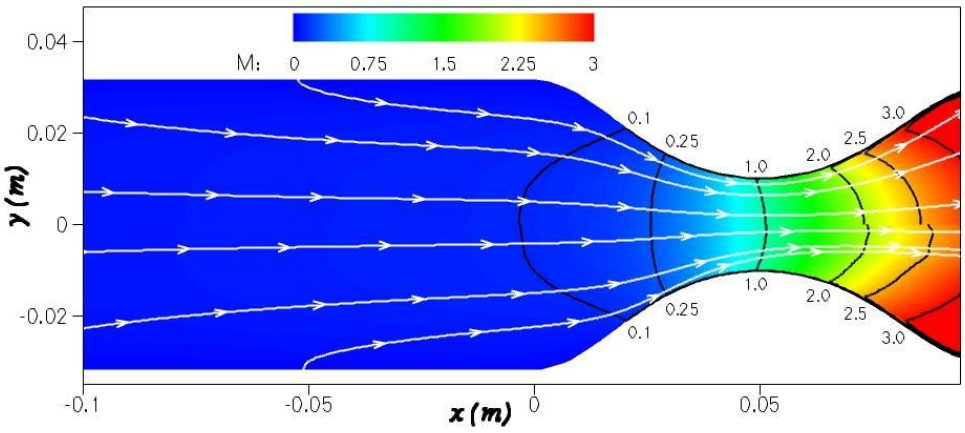


Figure 5.12: *Predicted Mach number contours for a gas-only flow (upper panel) and a gas-particle flow (lower panel) for a cylindrical grain rocket motor. Gas streamlines are shown in the upper panel for the gas-only flow and particle-phase streamlines are given in the lower panel for the gas-particle flow.*

The numerical results for this turbulent case are presented in Figures 5.11–5.14 for both a single-phase gas-only flow and a gas-particle flow. The results for the gas-only and gas-particle flows are shown in the upper and lower portions of the figures, respectively. The particle concentration contours are shown in Figure 5.13 for the two-phase flow case. The adapted multi-block mesh is shown in Figure 5.11. Note that the entire length of the rocket is not shown. Four levels of mesh refinement were used in both cases. The initial mesh included four mesh blocks, each with 288 cells, $(N_x, N_y) = (12, 24)$. The final mesh for the gas-only flow, included 40 blocks, 11,520 cells, and an efficiency of 0.844 as seen in the upper panel of Figure 5.11. The gas-particle flow included the gradient of the particle concentration as an additional refinement criteria resulting in a denser final mesh with 82 blocks, 23,616 cells, and an efficiency of 0.680. The burning of the solid propellant leads to a head end pressure in excess of 2.1 MPa for the gas-only flow and 1.95 MPa for the gas-particle flow. In both cases, sonic flow conditions are produced at the nozzle throat and supersonic outflow in the rocket nozzle with the Mach number exceeding 3. The Mach number contours are shown in Figure 5.12. Ahead of the nozzle throat, the Mach contours are very similar for both cases. However, after the nozzle throat, it can be seen that the heavy particles impede the acceleration of the gas and lower exit velocities are achieved. Also shown in this Figure are the streamlines for the gas in the upper panel from the gas-only flow and the particle-phase in the lower panel for the gas-particle flow. It is evident from the particle-phase streamlines that the high concentration of particles near the centre-line of the rocket are affecting the behaviour of the gas phase. As seen in Section 5.2, the particle concentration contours shown in Figure 5.13

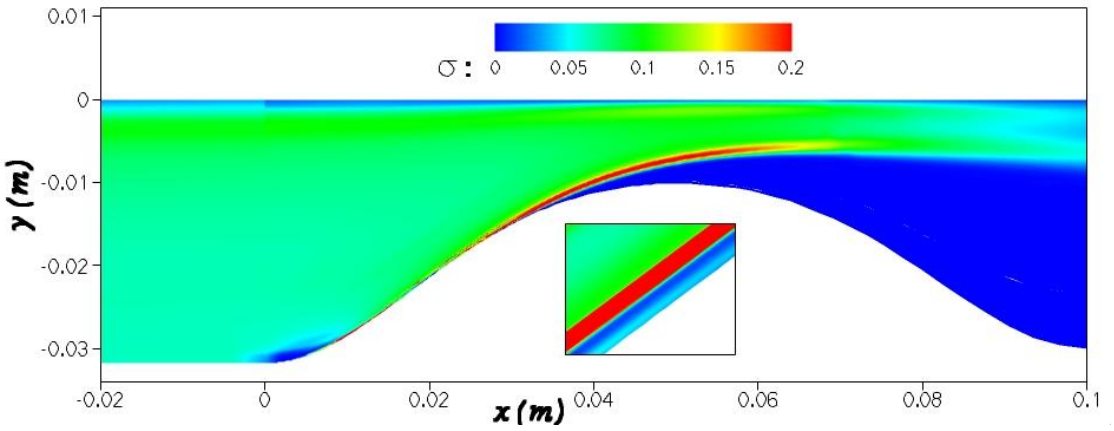


Figure 5.13: *Predicted particle-phase concentration contours for a cylindrical grain rocket motor.*

indicate that the particles before the nozzle throat are unable to negotiate the curved path of the converging section of the nozzle and impact the nozzle at a high velocity. The particles become entrained with the gas and are pulled through the nozzle throat into the diverging section where a dense stream of particles can be seen. The inset of this figure shows the particle concentration contours at the nozzle wall. The high particle concentration zone is actually off of the wall. This is a result of the reflection of the particles from the wall. The multi-velocity formulation for the particle-phase enabled this detailed prediction of the particle dynamics previously unattainable by the single-velocity formulation.

The contours of the turbulence intensity are shown in Figure 5.14. The turbulence intensity is determined as

$$I = \frac{\sqrt{k}}{U_0}, \tag{5.1}$$

where the reference speed, U_0, is taken to be the approximate speed of the gas at the transonic contour at the nozzle throat, $U_0 \approx 1000$ m/s. It can be seen that the maximum turbulence intensity occurs at the walls of the nozzle and at the centre-line of the rocket in the area of the nozzle throat. The turbulence intensity at the centre-line of the rocket has achieved a maximum of approximately 0.03 and 0.025 for the gas-only and the gas-particle flows, respectively. The heavy particles have, therefore, effectively reduced the intensity of the turbulence by 16.7%.

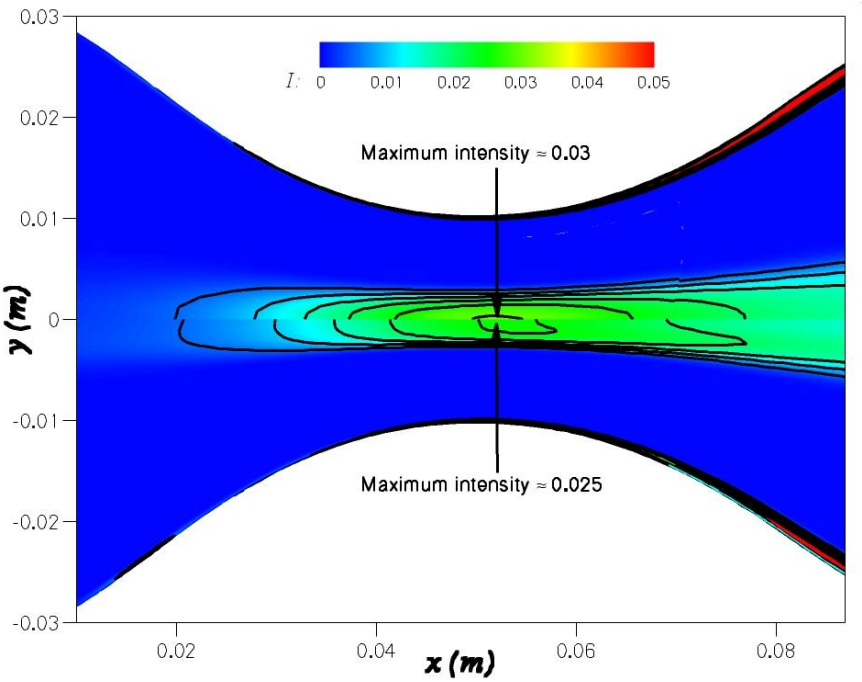


Figure 5.14: *Predicted turbulence intensity contours for a gas-only flow (upper panel) and a gas-particle flow (lower panel) for a cylindrical grain rocket motor.*

Chapter 6

Conclusions and Recommendations

6.1 Summary and Conclusions

A new computational framework for predicting two-dimensional axisymmetric turbulent multi-phase solid propellant rocket motor core flows has been proposed, developed, and demonstrated via application to be effective. The cornerstone of this numerical scheme is the use of a parallel block-based adaptive mesh refinement algorithm. AMR methods are effective in resolving the multiple solution scales typical of such complex fluid flow while minimizing computational expense. The use of body-fitted mesh blocks makes the application of the block-based AMR more amenable to flows with thin boundary layers and permits anisotropic refinement as dictated by initial mesh stretching. In addition, the block-based data structure lends itself naturally to domain decomposition and thereby enables efficient and scalable implementations of the algorithm on distributed-memory multi-processor architectures. To add even greater flexibility, a new mesh adjustment scheme has also been devised in which the body-fitted multi-block mesh is locally adjusted to arbitrarily embedded boundaries that are not necessarily aligned with the mesh. Not only does this scheme allow for rapid and robust mesh generation involving complex embedded boundaries, it also enables the solution of unsteady flow problems involving bodies and interfaces moving relative to the flow domain, such as the propellant grain of SRMs. Furthermore, the level set method has been adapted to successfully treat evolving embedded interfaces and boundaries. The cost of evolving the location of the combustion interface, tracing the location of the zero level set, and subsequently performing the mesh adjustment algorithm according to the new location of the embedded boundary was found to be only about one-and-a-half to two times the computational work associated with the application of one step of the explicit time-marching scheme, making the scheme quite efficient for unsteady computations. The mesh adjustment scheme proposed in this dissertation permits the treatment of stationary and dynamic embedded boundaries and is competitive and in many respects superior to

other techniques. The main advantages of this scheme are (1) only local alterations to the mesh are required, (2) the creation of small cut-cells is avoided, (3) the (i, j) data structure of the underlying body-fitted mesh is maintained, (4) conservation of the flow quantities is maintained, and (5) the scheme allows for effective parallel implementation.

An Eulerian formulation was used to describe the behaviour of the coupled gas-particle flow. Eigensystem and dispersion analyses of the one-dimensional form of the equations were conducted to gain the a better understanding of the wave structure and dynamic behaviour of the system equations as well as the degenerate nature of the inert, disperse, and dilute particle-phase equations. A Godunov-type finite-volume scheme and explicit time-marching schemes have been developed and implemented to accurately predict the interaction between the gas and solid particles injected into the rocket chamber at the combustion interface. The mathematical characteristics of the Eulerian formulation for the particle phase required a new solution method to be developed for the particle-phase that allows for a more physically realistic solution than was previously attainable. Single-velocity formulations are capable of predicting regions of zero particle concentration but are problematic with crossing particle trajectories or compression waves. The multi-velocity formulation described herein can account for crossing particle trajectories by splitting the particle-phase into distinct velocity families which are transported separately in the flow and thereby alleviates many of the difficulties associated with the degeneracy. Although this method requires some additional computing resources in terms of both memory (twelve additional variables) and approximately one-and-a-half times the computing time of the single-velocity formulation, it enables for the first time the treatment of crossing particle trajectories within an Eulerian framework. Stronger particle compression waves are easily reproduced and, therefore, crossing trajectories and reflection boundary conditions can be realistically handled. Significantly improved two-phase flow solutions can thus be obtained. A distinct advantage of using an Eulerian formulation over a Lagrangian formulation for the particle phase is that efficient parallel implementation is easily accomplished.

A considerable amount of attention has been given to the validation of each component of the computational framework described in this dissertation. This was accomplished by comparing predicted solutions to a variety of available analytic solutions and experimental data. The accuracy of the inviscid gas-phase discretization scheme was verified by comparing with Ringleb's hodograph solution for an isentropic, irrotational flow contained between two streamlines. In addition, the Mach reflection of a shock wave from a wedge was qualitatively compared to the shadowgraphs presented by Van Dyke [45]. The viscous gas-phase flux evaluation procedure was validated by comparing predicted solution of a laminar flat plate boundary layer with Blasius' analytical solution. Although not reported in this document, laminar channel and pipe flows were also used to validate

the viscous flux discretization procedure [170]. The experimental results taken by Laufer [107] for a turbulent pipe flow were used to validate the implementation of the k-ω turbulence model. The analytic solution to the one-dimensional piston problem was used to demonstrate the conservation properties of the moving boundary scheme. Landon's experimental measurements of the normal force coefficient for an oscillating NACA0012 aerofoil [103] was also used to validate the scheme for dynamic embedded boundaries. Predicted rocket motor results have also been validated by quantitatively comparing the two-dimensional predictions with the results of a one-dimensional analysis [64, 160] and the nozzleless rocket motor experimental results of Traineau *et al.* [193].

Coupling of the multi-velocity formulation and embedded mesh algorithm within the parallel AMR framework has resulted in a unique and powerful tool for the treatment of core flows in solid rocket motors. The viability of this computational tool for predicting complex internal rocket motor flows has also been clearly demonstrated. The proposed methodology will allow for future study of the structure and characteristics of two-dimensional axisymmetric SRM core flows, including the fluid and solid particle transport through the core flow and the effective thrust and choking of the rocket motor due to particle concentration. Other potential research includes the effects of structural oscillations [62], ignition dynamics [65, 4], and more sophisticated burning models [94, 159, 63, 22, 183]. Beyond modelling the regression of the combustion interface, the capability of simulating moving boundaries can allow for the study of the influence of flexing inhibitors on the fluid flow as well as propellant slumping effects at joint slots between propellant segments [54]. Figure 6.1 shows a preliminary computation of a solid propellant rocket motor core flow where two

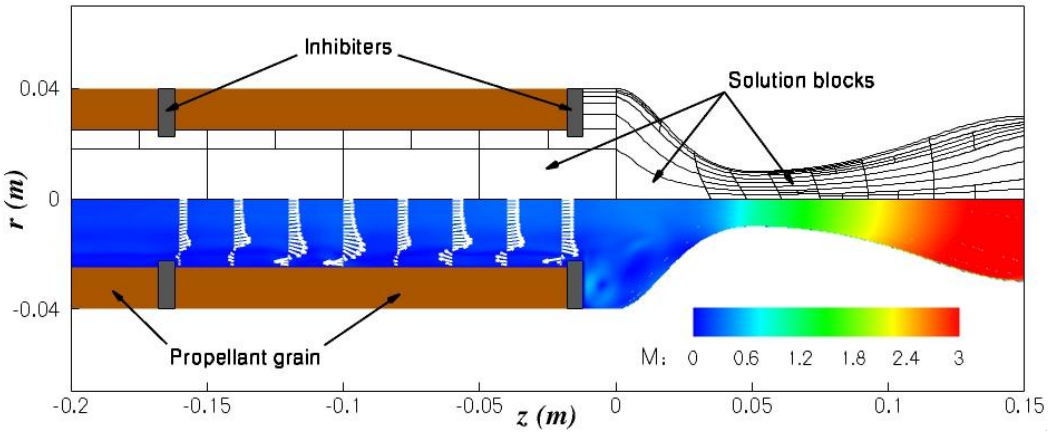


Figure 6.1: *Predicted cylindrical grain rocket motor results with two embedded inhibiters. The block structure is shown in the top panel after five levels of mesh refinement (195 blocks and 99,480 cells). The Mach number contours are shown in the lower panel as well as the velocity vectors in the chamber between the the two inhibiters.*

inhibiters have been embedded in the domain along with the propellant grain. The multi-block grid structure is shown in the top panel of the figure with five levels of mesh refinements (195 blocks and 99,480 cells). The predicted Mach number contours are shown in the bottom panel for a laminar flow and the velocity vectors shown in the rocket chamber indicate that vorticies have been generated by the leading inhibiter.

6.2 Recommendations

Although the computational framework outlined in this dissertation provides an excellent tool for studying the characteristics of rocket motor core flows, there are a number of areas in which additional capability could be added and the performance of the algorithm could be improved, which were deemed beyond the scope and time frame of this thesis. These recommendations for future research are summarized below.

The rocket motor results described in Chapter 5 make evident the capability of the block-based AMR scheme to provide additional resolution where it is required. In particular, the AMR scheme in conjunction with the use of body-fitted mesh blocks, allows for accurate prediction of the boundary-layer as it changes in thickness along the converging-diverging nozzle for laminar and turbulent flows. However, the current implementation of the AMR scheme only allows for the division of a block into four children blocks in the two-dimensional case. An alternative approach would be to allow blocks to divide into two children blocks with mesh refinement only taking place in one of the two space dimensions. This type of block division could enhance the cell stretching whereas the current method simply maintains it. Note that the block connectivity would now be maintained using a binary tree instead of the quadtree. This type of refinement was used by Ham *et al.* [72] on Cartesian meshes and by Ferm and Lötstedt [49] on body-fitted meshes. In addition, Ham *et al.* and Ferm and Lötstedt use error estimates to provide refinement criteria based on local truncation errors. Refinement criteria based on error estimates could lead to more efficient meshes. For unsteady flow, the current algorithm requires a user-specified frequency for the mesh refinement algorithm. Although effective, this process can be automated through the use of an error projection method [24].

Solid particles are often added to the propellant to enhance combustion and burning stability. In tactical rocketry, inert particles, such as aluminum oxide, Al_2O_3, are often used resulting in reduced smoke and thermal signatures. However, reactive particles, typically aluminum, are more commonly used as they also act as a fuel, injecting hot gas into the combustion chamber as they burn. Difficulties in the simulation of burning particles using Eulerian formulations (tracking

varying particle sizes) has forced some researchers to turn to Lagrangian formulations for the particle phase [44, 129, 168]. However, recent work by Laurent and co-workers on Eulerian modelling for polydisperse dense liquid sprays has provided a potential method for overcoming this difficulty [108, 109, 110]. For many applications, Eulerian methods are preferred over Lagrangian purposes due to computational efficiency and parallel scalability. Use of this scheme for simulating burning aluminum particles should be investigated as it could allow for a much broader range of rocket motor core flows to be studied.

The stencil for the linear least squares reconstruction of the gradients of the primitive solution quantities at cell (i, j) includes all eight of its neighbour cells. Ghost cells are used at all block boundaries and are filled appropriately so that the reconstruction stencil is consistent everywhere. However, at embedded boundaries the stencil is reduced by neglecting the inactive neighbour cells since it is difficult to fill these cells with reliable information. This can result in one-sided reconstruction stencils which can lead to poor approximations to the gradients. The reconstruction at embedded boundaries can be improved by imposing constraints on the least squares procedure as provided directly by the boundary condition data [135].

Due to the resolution required within boundary layers, only low-to-moderate Reynolds number flows involving embedded boundaries are tractable. However, this can be alleviated by using a hybrid mesh approach where a body-fitted mesh is attached to the embedded boundary. The mesh adjustment algorithm is then performed at the interface between the mesh fitted to the embedded boundary and the domain mesh. This approach has previously been used with the Cartesian cut-cell method [39]. Not only will the use of a hybrid mesh approach allow the computation of higher-Reynolds number flows but the oscillations that appear in the skin friction coefficient due to the non-smooth mesh will be eliminated. In addition, the block-based AMR scheme could be applied independently to the background and body-fitted meshes.

The explicit time-marching scheme with multigrid convergence acceleration, while adequate for the present study, is thought not to be optimal and further improvement to the time-marching scheme is possible. Parallel implicit methods for predicting steady-state inviscid and laminar combusting flows have been shown to be computationally efficient by Groth and Northrup [68] and Northrup and Groth [133] when coupled with the finite-volume scheme described in Chapter 3. This approach should be extended to turbulent multi-phase flow to provide an improvement over the performance of the time-marching scheme described in this thesis for steady-state calculations.

The calculation of unsteady flows can be time-prohibitive when using an explicit time-marching scheme due to stability constraints. This fact is aggravated when the unsteady flow involves dynamic embedded boundaries as the magnitude of the time-step is also restricted by the speed of the

boundary movement. The mesh must be readjusted as the embedded boundary advances through the domain and some of the solution content may have to be redistributed. If the boundary is allowed to move through more than one computational cell during a single time-step then the solution content in those cells would be lost. This same constraint exists for moving boundary problems using the Cartesian cut-cell method [12, 126]. In an attempt to circumvent this issue, Murman *et al.* [126] derived a correction term to account for the missing wall flux contribution if the boundary were to skip cells. This algorithm, implemented with the Cartesian cut-cell method, requires that the motion of the moving boundary can be predetermined. They showed encouraging results using a dual time-stepping scheme in which implicit Euler was used for the physical time-step. Research into this area should be continued to make such unsteady computations more tractable.

The current implementation of the mesh adjustment algorithm will only allow flow computation in the active cells. Extension of the numerical framework to not only allow for computation on both sides but also allow for different system of equations to be computed on either side would be straight-forward. In terms of rocket motors, such an extension would allow for a concurrent heat transfer and structural analysis inside the propellant grain and in any structural members inside the rocket (*e.g.*, inhibitors). The potential to simulate the erosion of the nozzle due to impact of solid particles also exists. Other applications include the large-eddy simulation of premixed turbulent flames, gas-liquid flows, and multi-phase flows.

The treatment for applying boundary conditions for the turbulence variables with mass injection should be investigated further. In addition, the incorporation of erosive [94, 159, 63] and transient burning effects [22, 183] should be incorporated into the propellant combustion model. Erosive burning rates can be of the same order of magnitude as the pressure dependent burning rate, particularly away from the head end of the rocket motor where the combustion products are moving more rapidly and during the initial phase of a rocket firing when the port-to-throat area is small [63].

One of the objectives of this dissertation was to provide insight into the requirements for developing a high-fidelity, efficient, and robust three-dimensional tool. Even though the block-based AMR scheme was essential in achieving the required mesh resolution with minimal cells in two-dimensions, the savings it will provide in terms of memory and computational time is even more critical in three space dimensions. With or without the changes recommended above, implementation of the AMR scheme to three-dimensional body-fitted mesh blocks would be straightforward. The multi-velocity formulation for the particle-phase will also extend simply to three dimensions. Here, there are eight velocity families to track and store. Although conceptually straightforward

and certainly possible, the mesh adjustment scheme may become fairly computationally intensive in three dimensions. Moreover, a large number of cells may then be required to provide an accurate representation and ample resolution of an arbitrarily shaped boundary, especially when predicting viscous flows. Nevertheless, despite the additional complexity in the logic that would be required, it is felt that this scheme would provide an efficient method for adjusting three-dimensional meshes to arbitrarily embedded boundaries. To help alleviate the number of cells required to accurately resolve geometrically complex embedded boundaries it is recommended that the hybrid mesh approach discussed above is adopted so that the mesh adjustment algorithm is only required to adjust to ellipsoids. Moreover, the interface between the meshes where the adjustment occurs should be located in regions of flow where viscous effects are negligible and mesh resolution requirements are reduced.

References

[1] M. J. Aftosmis, *Solution adaptive Cartesian grid methods for aerodynamics flows with complex geometries.* von Kármán Institute for Fluid Dynamics Lecture Series, 1997.

[2] M. J. Aftosmis, M. J. Berger, and G. Adomavicius, *A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries*, Paper 2000-0808, AIAA, January 2000.

[3] M. J. Aftosmis, M. J. Berger, and J. E. Melton, *Robust and efficient Cartesian mesh generation for component-base geometry*, AIAA Journal, 36 (1998), pp. 952–960.

[4] P. Alavilli, J. Buckmaster, T. L. Jackson, and M. Short, *Ignition-transient modeling for solid propellant rocket motors*, Paper 00–3567, AIAA, July 2000.

[5] R. W. Anderson, N. S. Elliot, and R. B. Pember, *An arbitrary Lagrangian-Eulerian method with adaptive mesh refinement for the solution of the Euler equations*, Journal of Computational Physics, 199 (2004), pp. 598–617.

[6] S. Apte and V. Yang, *Effect of acoustic oscillation on flow development in a simulated nozzleless rocket motor*, in Solid Propellant Chemistry, Combustion, and Motor Interior Ballistics, V. Yang, T. B. Brill, and W.-Z. Ren, eds., vol. 185 of Progress in Astronautics and Aeronautics, Reston, VA, 2000, AIAA, pp. 791–822.

[7] ——, *Unsteady flow evolution in porous chamber with surface mass injection, part 1: free oscillation*, AIAA Journal, 39 (2001), pp. 1577–1586.

[8] S. Balachandar, J. P. Ferry, and P. Bagchi, *Fundamental two-phase flow modeling efforts at CSAR*, Paper 00–3569, AIAA, July 2000.

[9] T. J. Barth, *Recent developments in high order k-exact reconstruction on unstructured meshes*, Paper 93-0668, AIAA, January 1993.

[10] T. J. Barth and P. O. Fredrickson, *Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction*, Paper 90-0013, AIAA, January 1990.

[11] T. J. Barth and D. C. Jespersen, *The design and application of upwind schemes on unstructured meshes*, Paper 89-0366, AIAA, January 1989.

[12] S. A. Bayyuk, K. G. Powell, and B. van Leer, *A simulation technique for 2-D unsteady inviscid flows around arbitrarily moving and deforming bodies of arbitrary geometry*, Paper 93–3391, AIAA, January 1993.

[13] J. A. Benek, J. Steger, and F. Dougherty, *A flexible grid embedding technique with applications to the Euler equations*, Paper 1983-1944, AIAA, 1983.

[14] M. J. Berger, *Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations*, PhD thesis, Stanford University, January 1982.

[15] ——, *Adaptive mesh refinement for hyperbolic partial differential equations*, Journal of Computational Physics, 53 (1984), pp. 484–512.

[16] M. J. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, Journal of Computational Physics, 82 (1989), pp. 67–84.

[17] M. J. Berger and R. J. LeVeque, *An adaptive Cartesian mesh algorithm for the Euler equations in arbitrary geometries*, Paper 89-1930, AIAA, June 1989.

[18] M. J. Berger and J. S. Saltzman, *AMR on the CM-2*, Applied Numerical Mathematics, 14 (1994), pp. 239–253.

[19] M. Berglund and C. Fureby, *Large eddy simulation of the flow in a solid rocket motor*, Paper 2001–0895, AIAA, January 2001.

[20] J. Blazek, *Flow simulation in solid rocket motors using advanced CFD*, Paper 03–5111, AIAA, July 2003.

[21] F. Bramkamp, J. Ballmann, and S. Müller, *Development of a flow solver employing local adaptation based on multiscale analysis on b-spline grids*, in Proceedings of the Eighth Annual Conference of the CFD Society of Canada, Montreal, Canada, June 11–13, 2000, vol. 1, CFD Society of Canada, 2000, pp. 111–118.

[22] M. Q. Brewster, *Solid propellant combustion response: Quasi-steady (QSHOD) theory development and validation*, in Solid Propellant Chemistry, Combustion, and Motor Interior Ballistics, V. Yang, T. B. Brill, and W.-Z. Ren, eds., vol. 185 of Progress in Astronautics and Aeronautics, Reston, VA, 2000, AIAA, pp. 607–638.

[23] W. Cai, F. Ma, and V. Yang, *Two-phase vorticoacoustic flow interactions in solid-propellant rocket motors*, Journal of Propulsion and Power, 19 (2003), pp. 385–396.

[24] P. A. Cavallo, S. Arunajatesan, N. Sinha, and T. J. Baker, *Transient mesh adaptation using an error wake and projection method*, Paper 2006-1149, AIAA, January 2006.

[25] J. C. Chassaing, G. A. Gerolymos, and I. Vallet, *Reynolds-stress model dual-time-stepping computation of unsteady three-dimensional flows*, AIAA Journal, 41 (2003), pp. 1882–1894.

[26] H. C. Chen and V. C. Patel, *Near-wall turbulence models for complex flows including separation*, AIAA Journal, 26 (1984), pp. 641–648.

[27] C. C. Chieng and B. E. Launder, *On the calculation of turbulent heat transport downstream from an abrupt pipe expansion*, Numerical Heat Transfer, 3 (1981), pp. 189–207.

[28] A. Ciucci and G. Iaccarino, *Numerical analysis of turbulent flow and alumina particle trajectories in solid rocket motors*, Paper 97–2860, AIAA, July 1997.

[29] A. Ciucci, G. Iaccarino, and M. Amato, *Numerical investigation of 3D two-phase turbulent flows in solid rocket motors*, Paper 98–3966, AIAA, July 1998.

[30] W. J. Coirier, *An Adaptively-Refined, Cartesian, Cell-Based Scheme for the Euler and Navier-Stokes Equations*, PhD thesis, University of Michigan, 1994.

[31] W. J. Coirier and K. G. Powell, *An accuracy assessment of Cartesian-mesh approaches for the Euler equations*, Journal of Computational Physics, 117 (1995), pp. 121–131.

[32] ——, *Solution-adaptive Cartesian cell approach for viscous and inviscid flows*, AIAA Journal, 34 (1996), pp. 938–945.

[33] P. Colella and P. R. Woodward, *The piecewise parabolic method (PPM) for gas-dynamical simulations*, Journal of Computational Physics, 54 (1984), pp. 174–210.

[34] F. E. C. Culick, *Combustion instabilities in solid propellant rocket motors*, in Internal Aerodynamics in Solid Rocket Propulsion, RTO Educational Notes EN-023, North Atlantic Treaty Organization, 2004, ch. 11.

[35] F. E. C. Culick and V. Yang, *Prediction of stability of unsteady motions in solid-propellant rocket motors*, in Nonsteady Burning and Combustion Stability of Solid Propellants, L. D. Luca, E. W. Price, and M. Summerfield, eds., vol. 143 of Progress in Astronautics and Aeronautics, Washington, D. C., 1992, AIAA, pp. 719–799.

[36] E. Daniel, T. Basset, and J. C. Loraud, *Eulerian approach for unsteady two-phase reactive solid rocket motor flows loaded with aluminum particles*, Paper 98–3697, AIAA, July 1999.

[37] R. L. Davis and J. F. Dannenhoffer, *Decomposition and parallelization strategies for adaptive grid-embedding techniques*, International Journal of Computational Fluid Dynamics, 1 (1993), pp. 79–93.

[38] D. De Zeeuw and K. G. Powell, *An adaptively refined Cartesian mesh solver for the Euler equations*, Journal of Computational Physics, 104 (1993), pp. 56–68.

[39] M. Delanaye, M. J. Aftosmis, M. J. Berger, Y. Liu, and T. H. Pulliam, *Automatic hybrid-Cartesian grid generation for high-Reynolds number flows around complex geometries*, Paper 99-0777, AIAA, 1999.

[40] W. A. Dick and M. T. Heath, *Whole system simulation of solid propellant rockets*, Paper 02–4345, AIAA, July 2002.

[41] W. A. Dick, M. T. Heath, and R. A. Fiedler, *Integrated 3-D simulation of solid propellant rockets*, Paper 01–3949, AIAA, July 2001.

[42] D. P. Dobkin, S. V. F. Levy, W. P. Thurston, and A. R. Wilks, *Contour tracing by piecewise linear approximations*, ACM Transactions on Graphics, 9 (1990), pp. 389–423.

[43] L. Dubuc, F. Cantariti, M. Woodgate, B. Gribben, K. J. Badcock, and B. E. Richards, *Solution of the unsteady Euler equations using an implicit dual-time method*, AIAA Journal, 36 (1998), pp. 1417–1424.

[44] J. Dupays, S. Wey, and Y. Fabignon, *Steady and unsteady reactive two-phase computations in solid rocket motors with Eulerian and Lagrangian approaches*, Paper 2001–3871, AIAA, 2001.

[45] M. V. Dyke, *An Album of Fluid Mechanics*, The Parabolic Press, Stanford, CA, 9th ed., 1982.

[46] B. Einfeldt, *On Godunov-type methods for gas dynamics*, SIAM Journal on Numerical Analysis, 25 (1988), pp. 294–318.

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

[48] C. Farhat, P. Geuzaine, and C. Grandmont, *The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids*, Journal of Computational Physics, 174 (2001), pp. 669–694.

[49] L. Ferm and P. Lötstedt, *Anisotropic grid adaptation for Navier-Stokes' equations*, Journal of Computational Physics, 190 (2003), pp. 22–41.

[50] J. Ferry and S. Balachandar, *A fast Eulerian method for disperse two-phase flow*, International Journal of Multiphase Flow, 27 (2001), pp. 1199–1226.

[51] ——, *Equilibrium expansion for the Eulerian velocity of small particles*, Powder Technology, 125 (2002), pp. 131–139.

[52] J. Ferry, S. L. Rani, and S. Balachandar, *A locally implicit improvement of the equilibrium Eulerian method*, International Journal of Multiphase Flow, 29 (2003), pp. 869–8891.

[53] R. Fiedler, X. Jiao, A. Namazifard, A. Hasselbacher, F. Najjar, and I. D. Parsons, *Coupled fluid-structure 3-D solid rocket motor simulations*, Paper 01–3954, AIAA, July 2001.

[54] R. A. Fiedler, M. S. Breitenfeld, X. Jiao, A. Hasselbacher, P. Geubelle, D. Guoy, and M. Brandyberry, *Simulations of slumping propellant and flexing inhibitors in solid rocket motors*, Paper 02–4341, AIAA, July 2002.

[55] X. Gao and C. P. T. Groth, *A parallel adaptive mesh refinement algorithm for predicting turbulent non-premixed combusting flows*, International Journal of Computational Fluid Dynamics, 20 (2006), pp. 349–357.

[56] P. Gerlinger and D. Brüggemann, *An implicit multigrid scheme for the compressible Navier-Stokes equations with low-Reynolds-number turbulence closure*, Journal of Fluids Engineering, 120 (1998), pp. 257–262.

[57] P. Geuzaine, *An Implicit Upwind Finite-Volume Method for Compressible Turbulent Flows on Unstructured Meshes*, PhD thesis, Université de Liège, 1999.

[58] P. Geuzaine, C. Grandmont, and C. Farhat, *Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations*, Journal of Computational Physics, 191 (2003), pp. 206–227.

[59] S. K. Godunov, *Finite-difference method for numerical computations of discontinuous solutions of the equations of fluid dynamics*, Matematicheskii Sbornik, 47 (1959), pp. 271–306.

[60] J. J. Gottlieb and C. P. T. Groth, *Assessment of Riemann solvers for unsteady one-dimensional inviscid flows of perfect gases*, Journal of Computational Physics, 78 (1988), pp. 437–458.

[61] ——, *Collection of boundary conditions for one- and some multi-dimensional unsteady flows of polytropic gases*, Canadian Aeronautics and Space Journal, 45 (1999), pp. 161–182.

[62] D. R. Greatrix, *Combined structural oscillation effects on solid rocket internal ballistics*, Paper 1999-2509, AIAA, June 1999.

[63] D. R. Greatrix and J. J. Gottlieb, *Erosive burning model for composite-propellant rocket motors with large length-to-diameter ratios*, Canadian Aeronautics and Space Journal, 33 (1987), pp. 133–142.

[64] D. R. Greatrix, J. J. Gottlieb, and T. Constantinou, *Quasi-steady analysis of the internal ballistics of solid-propellant rocket motors*, Canadian Aeronautics and Space Journal, 33 (1987), pp. 61–70.

[65] ——, *Numerical model for pellet-dispersion igniter systems*, Journal of Propulsion and Power, 4 (1988), pp. 412–420.

[66] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI*, MIT Press, Cambridge, Massachusetts, 1999.

[67] C. P. T. Groth, D. L. De Zeeuw, T. I. Gombosi, and K. G. Powell, *Global three-dimensional MHD simulation of a space weather event: CME formation, interplanetary propagation, and and interaction with the magnetosphere*, Journal of Geophysical Research, 105 (2000), pp. 25,053–25,078.

[68] C. P. T. Groth and S. A. Northrup, *Parallel implicit adaptive mesh refinement scheme for body-fitted multi-block mesh*, Paper 2005-5333, AIAA, 2005.

[69] C. P. T. Groth, P. L. Roe, T. I. Gombosi, and S. L. Brown, *On the nonstationary wave structure of a 35-moment closure for rarefied gas dynamics*, Paper 95-2312, AIAA, June 1995.

[70] C. P. T. Groth, D. L. D. Zeeuw, K. G. Powell, T. I. Gombosi, and Q. F. Stout, *A parallel solution-adaptive scheme for ideal magnetohydrodynamics*, Paper 99-3273, AIAA, June 1999.

[71] A. Haider and O. Levenspiel, *Drag coefficient and terminal velocity of spherical and nonspherical particles*, Powder Technology, 58 (1989), pp. 63–70.

[72] F. E. Ham, F. S. Lien, and A. B. Strong, *A Cartesian grid method with transient anisotropic adaptation*, Journal of Computational Physics, 179 (2002), pp. 469–494.

[73] A. Harten, *High resolution schemes for hyperbolic conservation laws*, Journal of Computational Physics, 49 (1983), pp. 357–393.

[74] ——, *On a class of high resolution total-variation-stable finite-difference schemes*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 1–23.

[75] A. Harten, B. Enquist, S. Osher, and S. R. Chakravarthy, *Uniformly high order accurate essentially non-oscillatory schemes, III*, Journal of Computational Physics, 71 (1987), pp. 231–303.

[76] A. Harten, P. D. Lax, and B. van Leer, *On upstream differencing and Godunov-type schemes for hyperbolic conservation laws*, SIAM Review, 25 (1983), pp. 35–61.

[77] R. Hartmann and P. Houston, *Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws*, SIAM Journal on Scientific Computing, 24 (2002), pp. 979–1004.

[78] A. Haselbacher and J. Blazek, *Accurate and efficient discretization of Navier-Stokes equations on mixed grids*, AIAA Journal, 38 (2000), pp. 2094–2102.

[79] M. T. Heath, R. A. Fiedler, and W. A. Dick, *Simulating solid propellant rockets at CSAR*, Paper 00–3455, AIAA, July 2000.

[80] A. Hegab, T. L. Jackson, J. Buckmaster, and D. S. Stewart, *The burning of periodic sandwich propellants*, Paper 00–3459, AIAA, July 2000.

[81] C. Hirsch, *Numerical Computation of Internal and External Flows, Volume 2, Computational Methods for Inviscid and Viscous Flows*, John Wiley & Sons, Toronto, 1990.

[82] C. W. Hirt, A. A. Amsden, and J. L. Cook, *An Arbitrary Lagrangian–Eulerian computing method for all flow speeds*, Journal of Computational Physics, 14 (1974), pp. 227–253. Reprinted in Vol. 135, 1997, pages 227–253.

[83] D. Holmes and S. D. Connell, *Solution of the 2D Navier-Stokes equations on unstructured adaptive grids*, Paper 89–1932, AIAA, June 1989.

[84] R. W. Humble, G. N. Henry, and W. J. Larson, eds., *Space Propulsion Analysis and Design*, Space Technology Series, McGraw-Hill, Toronto, 1995.

[85] J. D. Hunt, *An adaptive 3D Cartesian approach for the parallel computation of inviscid flow about static and dynamic configurations*, PhD thesis, University of Michigan, 2005.

[86] G. Iaccarino and R. Verzicco, *Immersed boundary technique for turbulent flow simulations*, Applied Mechanics Review, 56 (2003), pp. 331–347.

[87] O. Igra, G. Hu, J. Falcovitz, and B. Y. Wang, *Shock wave reflection from a wedge in a dusty gas*, International Journal of Multiphase Flow, 30 (2004), pp. 1139–1169.

[88] A. JAMESON, *Solution of the Euler equations by a multigrid method*, Applied Mathematics and Computation, 13 (1983), pp. 327–356.

[89] A. JAMESON AND S. YOON, *Multigrid solution of the Euler equations using implicit schemes*, AIAA Journal, 24 (1986), pp. 1737–1743.

[90] G.-S. JIANG AND D. PENG, *Weighted ENO schemes for Hamilton-Jacobi equations*, SIAM Journal on Scientific Computing, 21 (2000), pp. 2126–2143.

[91] G.-S. JIANG AND C.-W. SHU, *Efficient implementation of weighted ENO schemes*, Journal of Computational Physics, 126 (1996), pp. 202–228.

[92] W. P. JONES AND B. E. LAUNDER, *The prediction of laminarization with a two-equation model of turbulence*, International Journal of Heat and Mass Transfer, 15 (1972), pp. 301–314.

[93] S. M. H. KARIMIAN AND A. AMOLI, *Two-dimensional simulation of SRM internal ballistics on a moving grid*, Paper 99–2801, AIAA, June 1999.

[94] M. K. KING, *Erosive burning of composite solid propellants: experimental and modeling studies*, Journal of Spacecraft and Rockets, 16 (1979), pp. 154–162.

[95] B. KLEB, B. VAN LEER, AND B. WOOD, *Matching multi-stage schemes to viscous flow*, Paper 2005-4708, AIAA, June 2005.

[96] G. M. KNOTT AND M. Q. BREWSTER, *A two-dimensional model of composite propellant flame structure and burning rate*, Paper 00–3460, AIAA, July 2000.

[97] S. KOCHEVETS, J. BUCKMASTER, AND T. L. JACKSON, *Random propellant packs and the flames they support*, Paper 00–3461, AIAA, July 2000.

[98] W. R. KOLK, *Modern Flight Dynamics*, Prentice-Hall, 1961.

[99] P. KUENTZMANN, *Introduction to solid rocket propulsion*, in Internal Aerodynamics in Solid Rocket Propulsion, RTO Educational Notes EN-023, North Atlantic Treaty Organization, 2004, ch. 1.

[100] K. K. KUO AND M. SUMMERFIELD, eds., *Fundamentals of Solid Propellant Combustion*, vol. 90 of Progress in Astronautics and Aeronautics, AIAA, October 1986.

[101] C. K. G. LAM AND K. BREMHORST, *A modified form of the $k-\epsilon$ model for predicting wall turbulence*, Transactions of the ASME, 103 (1981), pp. 456–460.

[102] L. D. LANDAU AND E. M. LIFSHITZ, *Fluid Mechanics*, Pergamon Press, Oxford, England, 2nd ed., 1987.

[103] R. H. LANDON, *Compendium of unsteady aerodynamic measurements*, Advisory Report 702, NATO AGARD, August 1982.

[104] C. B. LANEY, *Computational Gasdynamics*, Cambridge University Press, Cambridge, 1998.

[105] J. V. LASSALINE AND D. W. ZINGG, *Development of an agglomeration multigrid algorithm with directional coarsening*, Paper 1999-3338, AIAA, 1999.

[106] ——, *An investigation of directional coarsening and line-implicit smoothing applied to agglomeration multigrid*, Paper 2003-3435, AIAA, 2003.

[107] J. Laufer, *The structure of turbulence in fully developed pipe flow*, Report 1174, NACA, 1954.

[108] F. Laurent and M. Massot, *Multi-fluid modelling of laminar polydisperse spray flames: origin, assumptions and comparison of sectional and sampling methods*, Combustion Theory and Modelling, 5 (2001), pp. 537–572.

[109] F. Laurent, M. Massot, and P. Villedieu, *Eulerian multi-fluid modelling for the numerical simulation of coalescence in polydisperse dense liquid sprays*, Journal of Computational Physics, 194 (2004), pp. 505–543.

[110] F. Laurent, V. Santoro, M. Noskov, M. D. Smooke, A. Gomez, and M. Massot, *Accurate treatment of size distribution effects in polydisperse spray diffusion flames: multi-fluid modelling, computations and experiments*, Combustion Theory and Modelling, 8 (2004), pp. 385–412.

[111] P. D. Lax, *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, vol. 11 of CMBS–NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, 1973.

[112] E. Li, *A parallel multigrid method for predicting compressible flow in turbomachinery*, Master's thesis, University of Toronto, 2004.

[113] T. Linde, *A practical, general-purpose, two-state HLL Riemann solver for hyperbolic conservation laws*, International Journal for Numerical Methods in Fluids, 40 (2002), pp. 391–402.

[114] X.-D. Liu, S. Osher, and T. Chan, *Weighted essentially non-oscillatory schemes*, Journal of Computational Physics, 115 (1994), pp. 200–212.

[115] H. Lomax, T. H. Pulliam, and D. W. Zingg, *Fundamentals of Computational Fluid Dynamics*, Scientific Computation, Springer, Berlin, 2001.

[116] J. F. Lynn, *Multigrid Solution of the Euler Equations with Local Preconditioning*, PhD thesis, University of Michigan, 1995.

[117] J. F. Lynn and B. van Leer, *Multi-stage schemes for the Euler and Navier-Stokes equations with optimal smoothing*, Paper 93-3355-CP, AIAA, July 1993.

[118] F. E. Marble, *Dynamics of dusty gases*, Annual Review of Fluid Mechanics, 2 (1970), pp. 397–446.

[119] S. R. Mathur and J. Y. Murthy, *A pressure-based method for unstructured meshes*, Numerical Heat Transfer Part B, 31 (1997), pp. 195–215.

[120] D. J. Mavriplis, *Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes*, Journal of Computational Physics, 145 (1998), pp. 141–165.

[121] M. R. MAXEY AND J. J. RILEY, *Equation of motion for a small rigid sphere in a uniform flow*, Physics of Fluids, 26 (1983), pp. 883–889.

[122] R. MEI, R. J. ADRIAN, AND T. J. HANRATTY, *Particle dispersion in isotropic turbulence under Stokes drag and Basset force with gravitational settling*, Journal of Fluid Mechanics, 225 (1991), pp. 481–495.

[123] R. B. MILNE, *An adaptive level set method*, PhD thesis, University of California - Berkeley, 1995.

[124] W. A. MULDER, *Multigrid relaxation for the Euler equations*, Journal of Computational Physics, 60 (1985), pp. 235–252.

[125] C. D. MUNZ, *On the numerical dissipation of high resolution schemes for hyperbolic conservation laws*, Journal of Computational Physics, 77 (1988), pp. 18–39.

[126] S. M. MURMAN, M. J. AFTOSMIS, AND M. J. BERGER, *Implicit approaches for moving boundaries in a 3-D Cartesian method*, Paper 03–1119, AIAA, January 2003.

[127] F. M. NAJJAR, S. BALACHANDAR, P. V. S. ALAVILLI, AND J. FERRY, *Computations of two-phase flow in aluminized solid propellant rockets*, Paper 00–3568, AIAA, July 2000.

[128] F. M. NAJJAR, J. FERRY, B. WASISTHO, AND S. BALACHANDAR, *Full-physics large-scale multiphase large eddy simulations of flow inside solid rocket motors*, Paper 02–4343, AIAA, July 2002.

[129] F. M. NAJJAR, A. HASELBACHER, J. FERRY, B. WASISTHO, S. BALACHANDAR, AND R. D. MOSER, *Large-scale multiphase large-eddy simulation of flows inside solid-rocket motors*, Paper 03–3700, AIAA, June 2003.

[130] A. NAMAZIFARD, I. D. PARSONS, A. ACHARYA, E. TACIROGLU, AND J. HALES, *Parallel structural analysis of solid rocket motors*, Paper 00-3457, AIAA, July 2000.

[131] R. W. NOACK, *DiRTlib: A library to add overset capability to your flow solver*, Paper 2006–5116, AIAA, 2006.

[132] R. W. NOACK AND J. P. SLOTNICK, *A summary of the 2004 overset symposium on composite grids and solution technology*, Paper 2004–0921, AIAA, 2004.

[133] S. A. NORTHRUP AND C. P. T. GROTH, *Solution of laminar combusting flows using a parallel implicit adaptive mesh refinement algorithm*, in Proceedings of the 4th International Conference on Computational Fluid Dynamics, Ghent, Belgium, July 10–14, 2006, 2006.

[134] G. C. OATES, *Aerothermodynamics of Gas Turbine and Rocket Propulsion*, Education Series, American Institute of Aeronautics and Astronautics, Reston, Virginia, 3rd ed., 1997.

[135] C. OLLIVIER-GOOCH AND M. V. ALTENA, *A high-order-accurate unstructured mesh finite-volume scheme for the advection-diffusion equation*, Journal of Computational Physics, 181 (2002), pp. 729–752.

[136] E. Orbekk, *Internal flow analysis of a technology demonstrator rocket motor with new CFD code*, Paper 98–3967, AIAA, July 1998.

[137] S. Osher and S. R. Chakravarthy, *Very high order accurate TVD schemes*, Report 84-44, ICASE, September 1984.

[138] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, vol. 153 of Applied Mathematical Sciences, Springer, 2003.

[139] S. Osher and R. P. Fedkiw, *Level set methods: an overview and some recent results*, Journal of Computational Physics, 169 (2001), pp. 463–502.

[140] S. Osher and J. A. Sethian, *Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations*, Journal of Computational Physics, 79 (1988), pp. 12–49.

[141] S. Osher and F. Solomon, *Upwind difference schemes for hyperbolic systems of conservation laws*, Mathematics of Computation, 38 (1982), pp. 339–374.

[142] H. Paillère, K. G. Powell, and D. L. De Zeeuw, *A wave-model-based refinement criterion for adaptive-grid computation of compressible flows*, Paper 92-0322, AIAA, January 1992.

[143] P. D. Palma, M. D. de Tullio, G. Pascazio, and M. Napolitano, *An immersed-boundary method for 3D compressible viscous flows*, Paper 2005–4715, AIAA, June 2005.

[144] S. H. Park and J. H. Kwon, *Implementation of k-ω turbulence models in an implicit multigrid method*, AIAA Journal, 42 (2004), pp. 1348–1357.

[145] I. D. Parsons, P. Alavilli, A. Namazifard, A. Acharya, X. Jiao, and R. Fiedler, *Coupled simulations of solid rocket motors*, Paper 00–3456, AIAA, July 2000.

[146] N. A. Patankar and D. D. Joseph, *Lagrangian numerical simulation of particulate flows*, International Journal of Multiphase Flow, 27 (2001), pp. 1685–1706.

[147] ——, *Modeling and numerical simulation of particulate flows by the Eulerian-Lagrangian approach*, International Journal of Multiphase Flow, 27 (2001), pp. 1659–1684.

[148] E. Peirano and B. Leckner, *Fundamentals of turbulent gas-solid flows applied to circulating fluidized bed combustion*, Progress in Energy and Combustion Science, 24 (1998), pp. 259–296.

[149] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang, *A PDE-based fast local level set method*, Journal of Computational Physics, 155 (1999), pp. 410–438.

[150] C. S. Peskin, *Numerical analysis of blood flow in the heart*, Journal of Computational Physics, 25 (1977), pp. 220–252.

[151] ——, *The immersed boundary method*, Acta Numerica, (2002), pp. 1–39.

[152] K. G. Powell, P. L. Roe, T. J. Linde, T. I. Gombosi, and D. L. De Zeeuw, *A solution-adaptive upwind scheme for ideal magnetohydrodynamics*, Journal of Computational Physics, 154 (1999), pp. 284–309.

[153] K. G. Powell, P. L. Roe, and J. Quirk, *Adaptive-mesh algorithms for computational fluid dynamics*, in Algorithmic Trends in Computational Fluid Dynamics, M. Y. Hussaini, A. Kumar, and M. D. Salas, eds., Springer-Verlag, New York, 1993, pp. 303–337.

[154] E. W. Price, *L^* instability*, in Nonsteady Burning and Combustion Stability of Solid Propellants, L. D. Luca, E. W. Price, and M. Summerfield, eds., vol. 143 of Progress in Astronautics and Aeronautics, Washington, D. C., 1992, AIAA, pp. 325–361.

[155] J. J. Quirk, *An Adaptive Grid Algorithm for Computational Shock Hydrodynamics*, PhD thesis, Cranfield Institute of Technology, January 1991.

[156] J. J. Quirk and U. R. Hanebutte, *A parallel adaptive mesh refinement algorithm*, Report 93-63, ICASE, August 1993.

[157] S. L. Rani and S. P. Vanka, *Direct numerical simulation of two-way coupling effects in a particle-laden turbulent pipe flow*, Paper 00–3570, AIAA, July 2000.

[158] P. C. Reist, *Aerosol Science and Technology*, McGraw-Hill Inc, New York, 1993.

[159] J. P. Renie and J. R. Osborn, *Erosive burning*, AIAA Journal, 21 (1983), pp. 1681–1689.

[160] W. Richardson-Little, *A one-dimensional model for predicting quasi-steady core flows in solid propellant rocket motors*, internal report, University of Toronto Institute for Aerospace Studies, 2000.

[161] W. Rodi, *Experience with two-layer models combining the k-e model with a one-equation model near the wall*, Paper 1991–0216, AIAA, 1991.

[162] P. L. Roe, *Approximate Riemann solvers, parameter vectors, and difference schemes*, Journal of Computational Physics, 43 (1981), pp. 357–372.

[163] ——, *Generalized formulation of TVD Lax-Wendroff schemes*, Report 84-53, ICASE, January 1984.

[164] P. L. Roe and J. Pike, *Efficient construction and utilisation of approximate Riemann solutions*, in Computing Methods in Applied Science and Engineering, R. Glowinski and J. L. Lions, eds., vol. VI, Amsterdam, 1984, North-Holland, pp. 499–518.

[165] E. Rouy and A. Tourin, *A viscosity solution approach to shape-from-shading*, SIAM Journal on Numerical Analysis, 29 (1992), pp. 867–884.

[166] G. Rudinger, *Fundamentals of Gas-Particle Flow*, Elsevier Scientific Publishing Company, Amsterdam, 1980.

[167] J. Sabnis, R. Madabhushi, H. Gibeling, and H. McDonald, *On the use of k-e turbulence model for computation of solid rocket internal flows*, Paper 89–2558, AIAA, 1989.

[168] J. S. Sabnis, *Numerical simulations of distributed combustion in solid rocket motors with metalized propellant*, Journal of Propulsion and Power, 19 (2003), pp. 48–55.

[169] J. S. Sachdev, *A review of dispersion modelling and particle trajectories in atmospheric flows*, Master's thesis, University of Toronto, 2000.

[170] J. S. Sachdev, C. P. T. Groth, and J. J. Gottlieb, *Parallel AMR scheme for turbulent multi-phase rocket motor core flows*, Paper 2005-5035, AIAA, June 2005.

[171] T. Saito, M. Marumoto, and K. Takayama, *Numerical investigations of shock waves in gas-particle mixtures*, Shock Waves, 13 (2003), pp. 299–322.

[172] H. Sauerwein and F. E. Fendell, *Method of characteristics in two-phase flow*, Physics of Fluids, 8 (1965), pp. 1564–1565. Research Note.

[173] R. Saurel, E. Daniel, and J. C. Loraud, *Two-phase flows: Second-order schemes and boundary conditions*, AIAA Journal, 32 (1994), pp. 1214–1221.

[174] H. Schlichting, *Boundary-Layer Theory*, McGraw-Hill, Toronto, 7th ed., 1979.

[175] J. A. Sethian, *Level Set Methods and Fast Marching Methods*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2nd ed., 1999.

[176] C.-W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, Journal of Computational Physics, 77 (1988), pp. 439–471.

[177] ——, *Efficient implementation of essentially non-oscillatory shock-capturing schemes: II*, Journal of Computational Physics, 83 (1989), pp. 32–78.

[178] T. Siikonen, *An application of Roe's flux-difference splitting for $k - \epsilon$ turbulence model*, International Journal for Numerical Methods in Fluids, 21 (1995), pp. 1017–1039.

[179] S. A. Slater and J. B. Young, *The calculation of inertial particle transport in dilute gas-particle flows*, International Journal of Multiphase Flow, 27 (2001), pp. 61–87.

[180] J. Steelant, E. Dick, and S. Pattijn, *Analysis of robust multigrid methods for steady viscous low Mach number flows*, Journal of Computational Physics, 136 (1997), pp. 603–628.

[181] T. N. Stevenson, *A law of the wall for turbulent boundary layers with suction or injection*, Aero. Rept. 166, College of Aeronautics, Cranfield, July 1963.

[182] M. Sun and K. Takayama, *Conservative smoothing on an adaptive quadrilateral grid*, Journal of Computational Physics, 150 (1999), pp. 143–180.

[183] S. T. Surzhikov, J. J. Murphy, and H. Krier, *2D model for unsteady burning heterogeneous AP/binder solid propellants*, Paper 00–3573, AIAA, July 2000.

[184] M. Sussman and E. Fatemi, *An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow*, SIAM Journal on Scientific Computing, 20 (1999), pp. 1165–1191.

[185] M. Sussman, E. Fatemi, P. Smereka, and S. Osher, *An improved level set method for incompressible two-phase flow*, Computers & Fluids, 27 (1998), pp. 663–680.

[186] M. Sussman and E. G. Puckett, *A coupled level set and volume-of-fluid method for computing 3D and axisymetric incompressible two-phase flows*, Journal of Computational Physics, 162 (2000), pp. 301–337.

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

[188] I. E. Sutherland and G. W. Hodgman, *Reentrant polygon clipping*, Communications of the ACM, 17 (1974), pp. 32–42.

[189] P. K. Sweby, *High resolution schemes using flux limiters for hyperbolic conservation laws*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 995–1011.

[190] P. D. Thomas and C. K. Lombard, *Geometric conservation law and its application to flow computations on moving grids*, AIAA Journal, 17 (1979), pp. 1030–1037.

[191] E. Toro, *Riemann Solvers and numerical methods for fluid dynamics*, Springer-Verlag, Berlin, 1999.

[192] E. F. Toro, M. Spruce, and W. Speares, *Restoration of the contact surface in the HLL-Riemann solver*, Shock Waves, 4 (1994), pp. 25–34.

[193] J.-C. Traineau, P. Hervat, and P. Kuentzmann, *Cold-flow simulation of a two-dimensional nozzleless solid rocket motor*, Paper 1986-1447, AIAA, June 1986.

[194] I. S. Tseng and V. Yang, *Computational of a double-base homogeneous propellant in a rocket motor*, Combustion and Flame, 96 (1994), pp. 325–342.

[195] Y.-H. Tseng and J. H. Ferziger, *A ghost-cell immersed boundary method for flow in complex geometry*, Journal of Computational Physics, 192 (2003), pp. 593–623.

[196] R. Turton and O. Levenspiel, *A short note on the drag correlation for spheres*, Powder Technology, 47 (1986), pp. 83–86.

[197] P. Vachal, R. V. Garimella, and M. J. Sashkov, *Untangling of 2D meshes in ALE simulations*, Journal of Computational Physics, 196 (2004), pp. 627–644.

[198] B. van Leer, *Towards the ultimate conservative difference scheme. I. The quest of monotonicity*, in Lecture Notes in Physics, vol. 18, New York, 1973, Springer-Verlag, pp. 163–168.

[199] ——, *Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second order scheme*, Journal of Computational Physics, 14 (1974), pp. 361–370.

[200] ——, *Towards the ultimate conservative difference scheme. III. Upstream-centered finite-difference schemes for ideal compressible flow*, Journal of Computational Physics, 23 (1977), pp. 263–275.

[201] ——, *Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection*, Journal of Computational Physics, 23 (1977), pp. 276–299.

[202] ——, *Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method*, Journal of Computational Physics, 32 (1979), pp. 101–136.

[203] B. van Leer, C. H. Tai, and K. G. Powell, *Design of optimally-smoothing multi-stage schemes for the Euler equations*, Paper 89-1933-CP, AIAA, June 1989.

[204] D. Vandromme and H. H. Minh, *About the coupling of turbulence closure models with averaged Navier-Stokes equations*, Journal of Computational Physics, 65 (1986), pp. 386–409.

[205] V. Venkatakrishnan, *Convergence to steady state solutions of the Euler equations on unstructured grids with limiters*, Journal of Computational Physics, 118 (1995), pp. 120–130.

[206] P. Venugopal, F. M. Najjar, and R. D. Moser, *DNS and LES computations of model solid rocket motors*, Paper 00–3571, AIAA, July 2000.

[207] F. Vuillot, T. Basset, J. Dupays, E. Daniel, and N. Lupoglazoff, *2D Navier-Stokes stability computations for solid rocket motors: rotational, combustion and two-phase flow effects*, Paper 97–3326, AIAA, January 1997.

[208] F. Vuillot and G. Casalis, *Motor flow instabilities – part 1*, in Internal Aerodynamics in Solid Rocket Propulsion, RTO Educational Notes EN-023, North Atlantic Treaty Organization, 2004, ch. 7.

[209] B. Wasistho, A. Haselbacher, F. M. Najjar, D. Tafti, S. Balachandar, and R. D. Moser, *Direct and large eddy simulations of compressible wall-injection flows in laminar, transitional, and turbulent regimes*, Paper 02–4344, AIAA, July 2002.

[210] K. Weiler and P. Atherton, *Hidden surface removal using polygon area sorting*, in Proceedings of SIGGRAPH '77, July 1977, pp. 214–222.

[211] D. C. Wilcox, *Turbulence Modeling for CFD*, DCW Industries, La Cañada, 1998.

[212] M. Wolfshtein, *The velocity and temperature distribution in one-dimensional flow with turbulence augmentation and pressure gradient*, International Journal of Heat and Mass Transfer, 12 (1969), pp. 301–318.

[213] B. J. York, R. A. Lee, N. Sinha, and S. M. Dash, *Progress in the simulation of particulate interactions in solid propellant rocket exhausts*, Paper 2001-3590, AIAA, July 2001.

[214] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher, *A variational level set approach to multiphase motion*, Journal of Computational Physics, 127 (1996), pp. 179–195.

[215] P. J. Zwart, G. D. Raithby, and M. J. Raw, *The integrated space-time finite volume method and its application to moving boundary problems*, Journal of Computational Physics, 154 (1999), pp. 497–519.

APPENDICES

Appendix A

Polygon and Line Intersection Routines

The mesh adjustment scheme outlined in Chapter 4 uses n-sided polygons to represent the embedded boundaries. Polygons are closed plane shapes that are bounded by three or more line segments. Various algorithms are described in this Appendix involving polygons, including: area calculation, centroid determination, a point-in-polygon routine, and the Weiler-Atherton algorithm [210] for finding the intersection or union of two polygons. It is assumed that the polygons are simple (not self-intersecting) and do not contain any holes. Concave and convex polygons are allowed. In addition, the vertexes of all polygons are listed in counter-clockwise order. Both the Weiler-Atherton and the point-in-polygon routines make extensive use of a line intersection routine. An efficient intersection method based on the solution of the equations resulting from the parameterization of the two line segments is also presented in this Appendix.

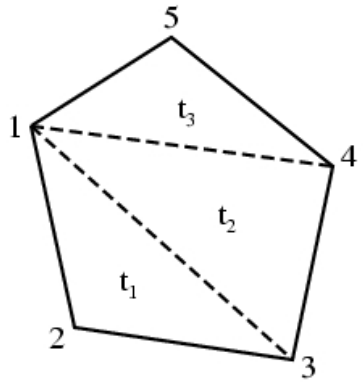


Figure A.1: *A 5-sided polygon split into three triangles. The first node is common to all of the triangles and the triangles do not intersect.*

Area of a Polygon

All n-sided polygons can be split into $n-2$ triangles. Figure A.1 shows a 5-sided polygon that is sectioned into three non-intersecting triangles. The first node is common in all of the triangles. The area of the n-sided polygon can be found by simply summing the area of each of the $n-2$ triangles, as given by

$$A = \frac{1}{2} \sum_{i=2}^{i=n-1} \left[(\vec{x}_i - \vec{x}_1) \times (\vec{x}_{i+1} - \vec{x}_1) \right] \cdot \hat{k} \,. \tag{A.1}$$

Note that for a convex polygon, a negative area results from one of the triangles since the vertexes of this triangle are listed in clock-wise order. This negative area removes duplicated and additional areas included by the other triangles and, therefore, resulting in the correct area.

Centroid of a Polygon

Determination of the centroid of an n-sided polygon is similar to the area calculation. The centroid of a triangle is the average of its three vertexes. The centroid of any polygon can be found by taking the area-weighted average of each of the sub-triangles,

$$\vec{x}A = \sum_{i=1}^{i=n-2} \vec{x}_{t_i} \left[(\vec{x}_i - \vec{x}_1) \times (\vec{x}_{i+1} - \vec{x}_1) \right] \cdot \hat{k} \,, \tag{A.2}$$

where $\vec{x}_{t_i}$ is the centroid of triangle i. For convex polygons, the negative area resulting from the triangles whose vertices are listed in clock-wise order correctly accounts for duplicated areas.

Point-In-Polygon Test

An efficient method is required to determine whether or not a a point of interest is contained within a polygon of interest. For example, this test is used to determine if a cell is contained inside or outside an embedded boundary. Two methods, bounding-box filters and ray-casting, are used to achieve this task in a rapid manner. These algorithms are used frequently in the Cartesian cut-cell approach [1].

The bounding-box filter provides a quick and cheap test that narrows down candidate points that may be contained within a polygon. The bounding-box is defined by $\vec{x}_{min} = (x_{min}, y_{min})$ and $\vec{x}_{max} = (x_{max}, y_{max})$ where the components of these vectors correspond to the minimum and maximum x and y values of the polygon. An example of a bounding-box for a seven-sided polygon is shown in the left side of Figure A.2. The bounding-box is indicated by the dashed line. The

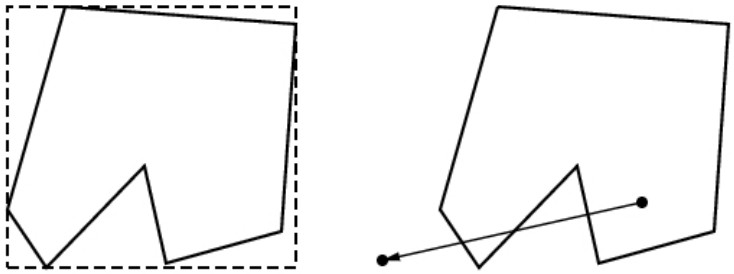


Figure A.2: *The bounding-box for a 7-sided polygon is shown in the left diagram where the bounding-box is given by the dashed line. An example of ray-casting for a point outside the polygon is shown on the right.*

point of interest, (x, y), may be inside the polygon if

$$x \geq x_{min}, \quad y \geq y_{min}, \quad x \leq x_{max}, \quad \text{and} \quad y \leq x_{max}. \tag{A.3}$$

If the point of interest is contained within the bounding-box then ray-casting is performed to make the final determination. The ray-tracing algorithm is performed by counting the number of intersections between the line composed of the point of interest and a reference point within the polygon and each edge of the polygon. An odd number of intersections indicates that the point is outside the polygon and an even number of intersections indicates that the point is inside. An example of ray-casting is shown in the right side of Figure A.2. Three intersection points exist and, therefore, the point is outside the polygon.

Weiler-Atherton Algorithm

The Weiler-Atherton algorithm is an efficient and robust method for finding the union or intersection of two overlapping polygons [210]. This algorithm is used in this work to determine the union of embedded boundary components and to determine the intersection of cells for performing a conservative restriction of the solution information from a coarse cell to a fine cell. An alternative method for finding the intersection of two polygons is the polygon clipping approach of Sutherland and Hodgman [188]. However, it was found that the Weiler-Atherton algorithm is more flexible and robust for general n- and m-sided polygons, whereas, the polygon clipping algorithm can be made extremely efficient if one of the polygons of interest is a non-rotated square. The latter method is often used in the Cartesian cut-cell method for this reason.

Consider the intersecting polygons in Figure A.3. The first polygon is a square with vertices 1, 2, 3, and 4. The second polygon is a triangle with vertices a, b, and c. One of the points of

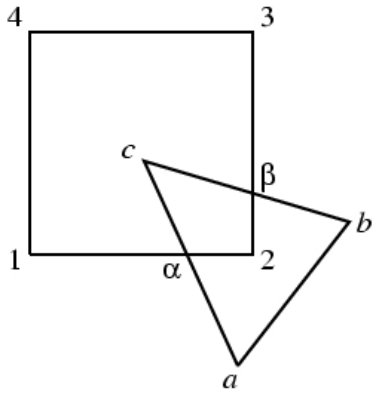


Figure A.3: *Two overlapping polygons, a square and a circle. The two points of intersection, α and β, are indicated.*

intersection, α, results from the intersection of edge $1-2$ of the square and edge $c-a$ of the triangle. The other point of intersection, β, results from the intersection of edge $2-3$ of the square and $b-c$ of the triangle. The first step of the Weiler-Atherton algorithm is to label each of the polygon vertices as inside or outside the other polygon. The points of intersection are then inserted into the appropriate location of each polygon's list of vertices. For the example in Figure A.3, the square's vertices are now 1, α, 2, β, 3, and 4 and the triangle's vertices are a, b, β, c, and α. The primary polygon may have to be reordered such that its first point is outside the second polygon. Similarly, the second polygon is reordered such that the first point is the first intersection point encountered when visiting the first polygon's list of vertices.

The polygons of union or intersection can now be determined by parsing the two lists. The polygon of union is found by including vertices that are labelled as outside or are intersection points. The polygons of intersection are found by including the intersection and points labelled as inside. Parsing of the lists are switched when the even numbered intersection point is encountered. This process is shown in Table A.1 where the square polygon is the primary polygon. The first point of the polygon, vertex 1, is outside the secondary polygon. This point is neglected when

Table A.1: *Example of the Weiler-Atherton algorithm for finding the intersection and union of two polygons.*

Square	1	α	2			β	3	4	
Triangle		α		a	b	β			c
Intersection		α	2			β			c
Union	1	α		a	b	β	3	4	

looking for the polygon of intersection and kept for the union polygon. The same is done for the second point, vertex 2. The third point is the first point of intersection, α. It is included in the union and intersection polygons. The fourth point, vertex 2, is inside the triangle and it is added to the polygon of intersection list and is ignored for the union polygon. The next point is the second intersection point, β. Before this point is included, the secondary list is parsed until the point β is reached. Both vertices a and b are external to the primary polygon and are neglected when compiling the polygon of intersection and included when finding the union polygon. The second intersection point, β, is now included and the parsing returns to the vertices of the primary polygon. The final two points, vertices 3 and 4, are both external to the secondary polygon are included or disregarded as is appropriate. Finally, all leftover vertices of the secondary polygon, in this case just c, are considered. The vertex c is inside the secondary polygon and is, therefore, included in the polygon of intersection and neglected for the union polygon. As is indicated in Table A.1, the polygon of intersection includes four vertices: α, 2, β, and c. The union polygon is a seven-sided polygon with the following vertices: 1, α, a, b, β, 3, and 4. Note that vertices of both resulting polygons are returned in counter-clockwise order.

Intersection of Two Lines

The ray-casting and Weiler-Atherton algorithms requires the use of an algorithm for the intersection of two line segments. The mesh adjustment scheme presented in Chapter 4 also requires extensive use of such an algorithm. Therefore, a highly efficient method is essential.

The method used here is based on the solution of the equations resulting from the parameterization of the two lines. The parameterization of two line segments are given by

$$\vec{r}_a = \vec{x}_{a_1} + s(\vec{x}_{a_2} - \vec{x}_{a_1}) = \vec{x}_{a_1} + s\triangle\vec{x}_a \tag{A.4}$$

$$\vec{r}_b = \vec{x}_{b_1} + s(\vec{x}_{b_2} - \vec{x}_{b_1}) = \vec{x}_{b_1} + t\triangle\vec{x}_b \tag{A.5}$$

where $\vec{x}_{a_1}$ and $\vec{x}_{a_2}$ are the end-points of one of the line segments and $\vec{x}_{b_1}$ and $\vec{x}_{b_2}$ are the end-points of the other line segment. The parameters s and t are constrained to lie between zero and one, $s \in [0,1]$ and $t \in [0,1]$.

The point of intersection requires that $\vec{r}_a = \vec{r}_b$ at some value of s^* and t^*. This results in the following system of equations,

$$\begin{bmatrix} \triangle x_a & -\triangle x_b \\ \triangle y_a & -\triangle y_a \end{bmatrix} \begin{bmatrix} s^* \\ t^* \end{bmatrix} = \begin{bmatrix} x_{b_1} - x_{a_1} \\ y_{b_1} - y_{a_1} \end{bmatrix}. \tag{A.6}$$

The determinant of the coefficient matrix is given by $\det = \triangle y_a \triangle x_b - \triangle x_a \triangle y_b$. If the determinant is zero then the two lines are either parallel and no intersection points exist or they are coincident and an infinite number of intersection points exist. Otherwise, the parameters for the point of intersection can be found by using Cramer's rule and are given by

$$s^* = \big[(y_{b_1} - y_{a_1})\triangle x_b - (x_{b_1} - x_{a_1})\triangle y_b\big]/\det\,, \tag{A.7}$$

$$t^* = \big[(y_{b_1} - y_{a_1})\triangle x_a - (x_{b_1} - x_{a_1})\triangle y_a\big]/\det\,. \tag{A.8}$$

Note that $s^* \in [0,1]$ and $t^* \in [0,1]$ must hold if the point of intersection is contained within both line segments. For the ray-casting routine, only the parameters s and t need to be determined. If the actual point of intersection is required then it can be found from

$$\vec{x} = \vec{x}_{a_1} + s^* \triangle \vec{x}_a = \vec{x}_{b_1} + t^* \triangle \vec{x}_{b_2}. \tag{A.9}$$

Appendix B

Weighted Essentially Non-Oscillatory Scheme

A weighted essentially non-oscillatory (WENO) scheme is used to determine all discrete approximations to the spatial derivatives required for the level set equation, Eikonal equation, and the scalar extension equation. These equations were described in section 4.3. The WENO scheme was originally proposed by Liu *et al.* [114] as a method to achieve high-order accurate approximation to the derivatives of a scalar conservation law by taking a convex combination of the three ENO stencils. In an ENO scheme, the stencil is generated by choosing the interpolating stencil which provides the smoothest solution [75, 176, 177]. Taking a linear combination of the ENO stencils can improve the accuracy (from 3rd to 4th order) of the scheme in smooth regions and the contribution of a stencil interpolating across a discontinuity can easily be minimized. Jiang and Shu [91] determined a set of weights that allows the scheme to achieve the optimal 5th order accuracy in smooth regions. The WENO scheme was extended to Hamilton-Jacobi equations by Jiang and Peng [90]. The construction of the WENO stencil is presented in this Appendix and follows closely the discussion found in the textbook by Osher and Fedkiw [138].

The WENO approximation to a spatial derivative is determined from a convex combination of three ENO stencils. The x-direction spatial derivative is determined from

$$\psi_x = \omega_1 \psi_x^1 + \omega_2 \psi_x^2 + \omega_3 \psi_x^3 \,, \tag{B.1}$$

where ψ_x^1, ψ_x^2, and ψ_x^3 are the ENO approximations to ψ_x and ω_i are the weights that must sum to one, $\omega_1 + \omega_2 + \omega_3 = 1$. The ENO stencils can be expressed as a combination of the local derivatives for the cells involved in the stencil. Defining

$$\begin{gathered} v_1 = \frac{\psi_{i-2,j} - \psi_{i-3,j}}{x_{i-2,j} - x_{i-3,j}} \,, \quad v_2 = \frac{\psi_{i-1,j} - \psi_{i-2,j}}{x_{i-1} - x_{i-2,j}} \,, \quad v_3 = \frac{\psi_{i,j} - \psi_{i-1,j}}{x_{i,j} - x_{i-1,j}} \,, \\ v_4 = \frac{\psi_{i+1,j} - \psi_{i,j}}{x_{i+1,j} - x_{i,j}} \,, \quad v_5 = \frac{\psi_{i+2,j} - \psi_{i+1,j}}{x_{i+2,j} - x_{i+1,j}} \,, \end{gathered} \tag{B.2}$$

for determining an approximation to the backward-biased spatial derivative, ψ_x^-, and

$$\begin{gathered} v_1 = \frac{\psi_{i+3,j} - \psi_{i+2,j}}{x_{i+3,j} - x_{i+2,j}}\,, \quad v_2 = \frac{\psi_{i+2,j} - \psi_{i+1,j}}{x_{i+2,j} - x_{i+1,j}}\,, \quad v_3 = \frac{\psi_{i+1,j} - \psi_{i,j}}{x_{i+1,j} - x_{i,j}}\,, \\ v_4 = \frac{\psi_{i,j} - \psi_{i-1,j}}{x_{i,j} - x_{i-1,j}}\,, \quad v_5 = \frac{\psi_{i-1,j} - \psi_{i-2,j}}{x_{i-1,j} - x_{i-2,j}}\,, \end{gathered} \tag{B.3}$$

for determining an approximation to the forward-biased spatial derivative, ψ_x^+. The ENO approximations are

$$\psi_x^1 = \frac{v_1}{3} - \frac{7v_2}{6} + \frac{11v_3}{6}\,, \quad \psi_x^2 = -\frac{v_2}{6} + \frac{5v_3}{6} + \frac{v_4}{3}\,, \quad \psi_x^3 = \frac{v_3}{3} + \frac{5v_4}{6} - \frac{v_5}{6}. \tag{B.4}$$

Jiang and Peng [90] defined the following smoothness estimates of the stencils

$$\begin{aligned} s_1 &= \frac{13}{12}(v_1 - 2v_2 + v_3)^2 + \frac{1}{4}(v_1 - 4v_2 + 3v_3)^2\,, \\ s_2 &= \frac{13}{12}(v_2 - 2v_3 + v_4)^2 + \frac{1}{4}(v_2 - v_4)^2\,, \\ s_3 &= \frac{13}{12}(v_3 - 2v_4 + v_5)^2 + \frac{1}{4}(3v_3 - 4v_4 + v_5)^2. \end{aligned} \tag{B.5}$$

These smoothness estimates are used to define the weights

$$\alpha_1 = \frac{0.1}{(s_1 + \epsilon)^2}\,, \quad \alpha_2 = \frac{0.6}{(s_2 + \epsilon)^2}\,, \quad \alpha_3 = \frac{0.3}{(s_3 + \epsilon)^2} \tag{B.6}$$

where $\epsilon = 10^{-6}\max(v_1^2, v_2^2, v_3^2, v_4^2, v_5^2) + 10^{-99}$. The required weightings for the spatial derivatives are then determined by normalizing these initial weights as follows:

$$\omega_i = \frac{\alpha_i}{\alpha_1 + \alpha_2 + \alpha_3}. \tag{B.7}$$

The WENO scheme is described above is used to calculate approximations to the spatial derivative in the forward and backward directions for each spatial dimensions, *e.g.*, ψ_x^+ and ψ_x^- for the x-directions derivatives. Once these approximations have been determined, the correct approximation can be chosen such that the appropriate upwinding is obtained. For the passive advection term of the level set equation, equation (4.1), and for the scalar extension equation, equation (4.6), upwind differencing is used. Here the flux can be found directly. The flux in the x-direction, for example, is determined from

$$u\psi_x = \begin{cases} \min(u\psi_x^-, u\psi_x^+), & \text{if } \ \psi_x^- < \psi_x^+, \\ \max(u\psi_x^-, u\psi_x^+), & \text{if } \ \psi_x^- > \psi_x^+, \end{cases} \tag{B.8}$$

where u is the advection speed in the x-direction. The approximation of ψ_y is similarly constructed. Note that this is exactly Godunov's scheme for a scalar equation [59]. For motion in the normal

direction, Godunov's scheme was expressed simply by Rouy and Tourin [165] as

$$\psi_x = \begin{cases} \sqrt{\max\left(\max(\psi_x^-, 0)^2, \min(\psi_x^+, 0)^2\right)}, & \text{if} \quad F > 0, \\ \sqrt{\max\left(\min(\psi_x^-, 0)^2, \max(\psi_x^+, 0)^2\right)}, & \text{if} \quad F < 0, \end{cases} \tag{B.9}$$

where F is given by $S(\psi)$ when using the Eikonal equation for re-distancing the signed-distance field.

Appendix C

Contour Tracing by Piecewise Linear Approximations

In Section 4.3, the level set method is described for determining the motion and evolution of embedded boundaries associated with various physical processes. The level set method is applied and solved on a separate Cartesian mesh that overlaps the adjusted body-fitted mesh used to predict the fluid equations. However, the mesh adjustment scheme requires an explicit representation of the embedded boundary at its current location. Therefore, a contour tracing algorithm is required to provide an approximation of the zero level set. The algorithm proposed by Dobkin *et al.* [42] is used for this purpose. This method generates a piecewise linear approximation for the zero level set contour. A brief illustration of this algorithm now follows.

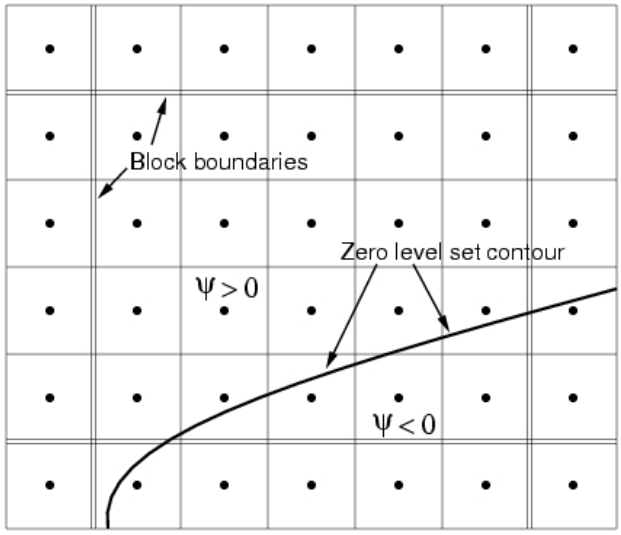


Figure C.1: *Example level set solution block. The zero level set contour is indicated by the thick line. The dots are the locations of the cell centres and the thin lines are the mesh lines. The block boundaries are denoted by the the double lines. Only one layer of ghost cells are required.*

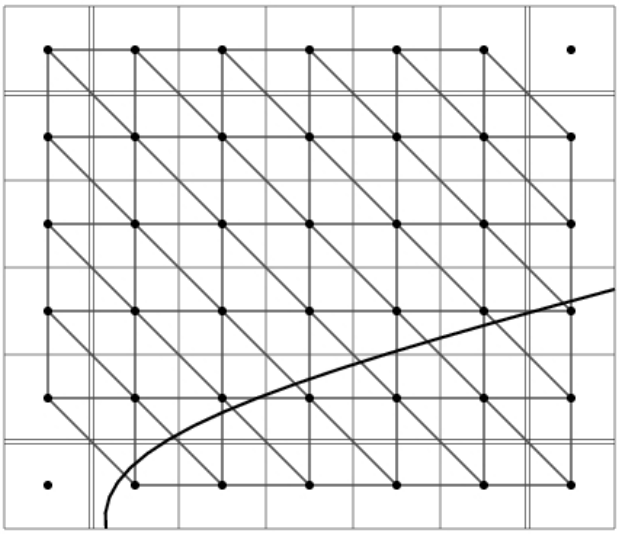


Figure C.2: *Triangulation used for the contour tracing algorithm. The vertexes of the triangles are located at the centres of the cells.*

Consider the zero level set contour shown in Figure C.1. The values of the level set function are known at the cell centres (indicated by the dots). The mesh lines and the boundaries of the block are also noted. The algorithm of Dobkin *et al.* requires the domain to be triangulated as done in Figure C.2. The vertexes of the triangle coincide with the centroids of the cells. It can be seen in this figure that if the contour of interest penetrates an edge of a triangle then it must exit one of its other edges. This forms the basis of the method. Refer to Figure C.3. Given a starting edge, linear interpolation can be used to determine the penetration point. The neighbour edges of the triangle can then be sampled for the next contour point which is simply determined by the vertexes with values opposite in sign. For contours that cross the boundaries of a block, the starting edge is chosen such that it is parallel to a block boundary and within the first ghost cell. This choice of a starting point eliminates the need to trace forwards and backwards from the starting point.

The preceding algorithm results in a piecewise linear representation of the zero level set contour contained in each block. These curve segments are clipped to conform to the block boundaries and are concatenated with the segments from neighbour blocks as necessary. It should be noted that each processor will search for and trace all zero level set contours contained in the blocks that it owns. Therefore, this algorithm takes advantage of the domain decomposition and requires no information of neighbour beyond what is contained in the ghost cells. However, the concatenation of the line segments does require communication between the processors.

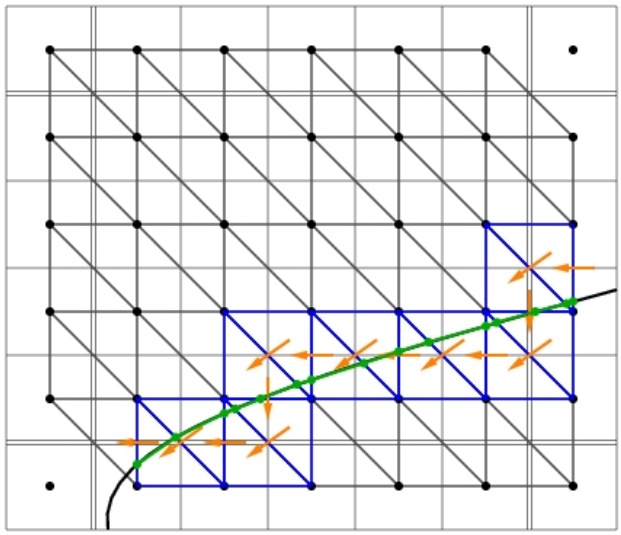


Figure C.3: *Illustration of the tracing process. Given a starting point, the contour must penetrate one of the other edges of the triangle. The arrows indicate the direction of the trace and the visited triangles are highlighted in blue.*

Appendix D

Burning Surface Boundary Condition

Boundary conditions for a regressing burning surface that injects gas-particle products into the flow cavity is formulated in terms of a Riemann problem and used here to specify boundary data at the surface of the burning propellant. The treatment of the burning propellant boundary is very similar in spirit to the methods proposed by Gottlieb and Groth [61] for imposing boundary data at a variety of flow boundaries based on the solution of Riemann problems.

For the current analysis, the burning rate is computed by the application of the pressure dependent empirical St. Robert relation,

$$\rho_{bs} v_{bs} = (1-\alpha_s)\rho_s r_{bs} = (1-\alpha_s)\rho_s \beta\, p_{bs}^n \tag{D.1}$$

where ρ_{bs}, v_{bs}, and p_{bs} are the density, normal velocity, and pressure of the gas injected from the burning surface and ρ_s is the solid propellant density. The burning rate constants β and n are functions of the chemical composition of the solid and the adiabatic flame temperature, T_f. The mass fraction of solid particles in the propellant is given by α_s. Note that the speed of the propagating surface is $-r_{bs}$. The particles are injected into the combustion chamber at the same speed as the gas-phase and at the adiabatic flame temperature.

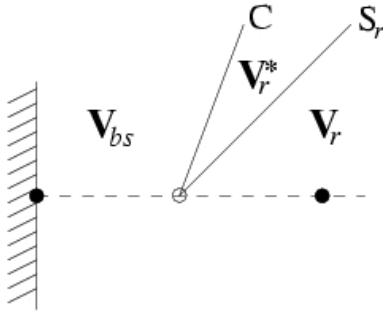


Figure D.1: *Burning surface wave pattern.*

The wave diagram corresponding to the solution for the burning surface Riemann problem is shown in Figure D.1. There are two waves: a contact surface and a second wave which can be a shock or a rarefaction wave. An exact Riemann solver has been constructed to solve this problem.

For the left-hand side boundary depicted in Figure D.1, the primitive variable states $\mathbf{V}_r$, $\mathbf{V}_r^*$, and $\mathbf{V}_l = \mathbf{V}_l^* = \mathbf{V}_{bs}$ are given by

$$\mathbf{V}_r = [\rho_r,\, v_{xr},\, v_{yr},\, p_r,\, \sigma_{pr},\, u_{xr},\, u_{yr},\, T_{pr}]^T\,, \tag{D.2}$$

$$\mathbf{V}_r^* = \left[\rho_r^*,\, v_{bs},\, v_{yr}^*,\, p_{bs},\, \sigma_{pr}^*,\, u_{bs},\, u_{yr}^*,\, T_f\right]^T\,, \tag{D.3}$$

$$\mathbf{V}_{bs} = [\rho_{bs},\, v_{bs},\, 0,\, p_{bs},\, \sigma_{pbs},\, u_{bs},\, 0,\, T_f]^T \tag{D.4}$$

where $u_{bs} = v_{bs}$. The resulting system of equations, the shock/rarefaction equations and the burning rate equation (D.1), is nonlinear and implicit in the unknown burning surface pressure or normal velocity, p_{bs} and v_{bs}. A Newton iteration scheme has been developed to solve this scalar nonlinear equation and iterates on the burning surface pressure. The propagation of the burning surface is accounted for through an application of a Galilean transformation to a frame in which the surface is stationary. In this frame, the right-state velocity is given by $v_{xr}^s = v_{xr} + r_{bs}$. The burning surface velocity in the stationary frame, v_{bs}^s, can now be determined by the iterative scheme.

The right state pressure can be used as the initial guess. If the burning surface pressure is greater than or equal to the right state pressure, $p_{bs} \geq p_r$, then the right hand wave is a shock. The following equations are used to find the speeds and density across the shock:

$$v_{bs}^s = v_{xr} + r_{bs} + \frac{1}{\sqrt{C_1}}\frac{a_r}{\gamma}\left(\frac{p_{bs}}{p_r} - 1\right), \tag{D.5}$$

$$v_{bs}^{s\,\prime} = \frac{\mathrm{d}v_{bs}}{\mathrm{d}p_{bs}} = r_{bs}' + \frac{1}{\sqrt{C_1}}\frac{a_r C_2}{\gamma p_r}, \tag{D.6}$$

$$\rho_r^* = \rho_r\left[\frac{(\gamma+1)p_{bs} + (\gamma-1)p_r}{(\gamma-1)p_{bs} + (\gamma+1)p_r}\right], \tag{D.7}$$

$$C_1 = \frac{\gamma+1}{2\gamma}\frac{p_{bs}}{p_r} + \frac{\gamma-1}{2\gamma}, \tag{D.8}$$

$$C_2 = 1 - \frac{1}{C_1}\frac{\gamma+1}{4\gamma}\left(\frac{p_{bs}}{p_r} - 1\right). \tag{D.9}$$

If $p_{bs} < p_r$, then the right-hand wave is a rarefaction wave. The following equations can be used to

determine the speeds and density across the rarefaction wave:

$$a_r^* = a_r \left(\frac{p_{bs}}{p_r} \right)^{\frac{\gamma-1}{2\gamma}}, \tag{D.10}$$

$$v_{bs}^s = v_{xr} + r_{bs} + \frac{2}{\gamma-1}(a_r^* - a_r), \tag{D.11}$$

$$v_{bs}^{s\,\prime} = r_{bs}' + \frac{a_r^*}{\gamma p_{bs}}, \tag{D.12}$$

$$\rho_r^* = \rho_r \left(\frac{p_{bs}}{p_r} \right)^{\frac{1}{\gamma}}. \tag{D.13}$$

The gas density and particle concentration injected into the core by the burning surface can be determined from $p_{bs} = \rho_{bs} \mathrm{R} T_f$ and $\sigma_{bs} = \alpha_s \rho_s r_{bs} / v_{bs}^s$. The derivative of St. Robert's burning rate equation, (D.1), with respect to the pressure at the burning surface is given by

$$r_{bs}' = n \frac{r_{bs}}{p_{bs}}. \tag{D.14}$$

The pressure is then up-dated by

$$p_{bs}^{(n+1)} = p_{bs}^{(n)} - \frac{\rho_s r_{bs} - \rho_{bs} v_{bs}^s}{\rho_s r_{bs}' - (\rho_{bs} v_{bs}^s)'} \tag{D.15}$$

where the derivative of the gas flux from the burning surface, $(\rho_{bs} v_{bs}^s)'$, with respect to p_{bs} is

$$(\rho_{bs} v_{bs}^s)' = \frac{\rho_{bs} v_{bs}^s}{p_{bs}} + \rho_{bs} v_{bs}^{s\,\prime}. \tag{D.16}$$

The iteration process is complete when $|1 - p_{bs}^{(n)} / p_{bs}^{(n+1)}| < \epsilon$. Finally, the burning surface velocity in the initial frame can be determined by $v_{bs} = v_{bs}^s - r_{bs}$.

Appendix E

Parallel Multigrid Method

The multigrid method is a popular method for accelerating the convergence of steady-state fluid flow calculations [88, 124, 89, 116, 120, 105, 106, 112, 55]. The parallel multigrid method developed by Li [112] was adopted in this dissertation. A brief overview of this multigrid approach is provided next. The following sections describe modifications to the scheme required for turbulent flows and flows with embedded boundaries, respectively.

Overview of the Multigrid Method

The multigrid method accelerates the convergence of iterative methods through the use of multiple scales of discretization. On a given grid, well-approximated low frequency error components are more difficult to eliminate than the poorly supported high frequency error components. However, on a coarser grid, the low frequency error components from the fine mesh appear to be of a higher-frequency. The use of multiple grid levels in combination with a relaxation or iterative scheme having good high frequency damping provides the basis of the multigrid method.

An initial solution on the finest grid can be determined by the application of the relaxation scheme. The explicit optimally-smoothing multi-stage temporal discretization scheme discussed in Section 3.2 is used to provide damping of the high-frequency solution content [203]. The approximate solution residual on grid level ℓ can be written as

$$\mathbf{R}_\ell = \mathbf{R}_\ell\big(\mathbf{U}_\ell^{(n)}\big)\,, \tag{E.1}$$

at iteration n of the relaxation scheme. On structured meshes in two-dimensions, the coarse grids can be generated by agglomerating four fine grid quadrilateral cells into one cell. The solution on grid level $\ell+1$ is approximated by $\mathbf{U}_{\ell+1} = I_\ell^{\ell+1}\mathbf{U}_\ell$ where $I_\ell^{\ell+1}$ is the restriction operator from grid level ℓ to grid level $\ell+1$. An area-weighted average of the fine grid cells is used to restrict the solution information in this work. The residual on grid level $\ell+1$ is determined by

$$\mathbf{R}_{\ell+1} = \mathbf{R}_{\ell+1}\big(\mathbf{U}_{\ell+1}^{(n)}\big) + \mathbf{P}_{\ell+1}\,, \tag{E.2}$$

where the forcing function, $\mathbf{P}_{\ell+1}$, is an additional term constructed from the the fine grid solution that ensures that the accuracy of the residual on the fine grid is maintained on the coarse mesh. This scheme is known as a full approximation storage scheme since a complete approximation of the current fine grid solution is assembled on the coarse grid. The forcing term is given by

$$\mathbf{P}_{\ell+1} = \hat{I}_{\ell}^{\ell+1}\mathbf{R}_{\ell} - \mathbf{R}_{\ell+1}\left(I_{\ell}^{\ell+1}\mathbf{U}_{\ell}^{(n)}\right). \tag{E.3}$$

As is done for the restriction of the solution, an area-weighted average of the fine grid residuals is used for the residual restriction operator, $\hat{I}_{\ell}^{\ell+1}$. An improvement to the approximate solution on grid level ℓ is made by adding a coarse-grid correction as given by

$$\mathbf{U}_{\ell}^{(n+1)} = \mathbf{U}_{\ell}^{(n)} + I_{\ell+1}^{\ell}\left(\mathbf{U}_{\ell+1}^{(n)} - I_{\ell}^{\ell+1}\mathbf{U}_{\ell}^{(n)}\right), \tag{E.4}$$

where the prolongation operator, $I_{\ell+1}^{\ell}$, is used to transfer the solution information from grid level $\ell+1$ to grid level ℓ. Bilinear interpolation is used in the solution prolongation process.

The procedure described above can be used with as many grid levels as the initial mesh can support. Various multigrid cycles can then be formulated. Both V and W multigrid cycles have been implemented and a full-multigrid cycle can be used for start-up purposes.

An example of the convergence history for a calculation using the multigrid method is given in Figure E.1. The inviscid transonic flow over a semi-circular bump in a channel is used as the test

Figure E.1: *L_2-norm of density residual for the inviscid transonic flow over a semi-circular bump. The mesh contains 8 blocks and 8,192 cells.*

problem as described in Section 3.6. The mesh consisted of eight solution blocks each with 1024 cells, $(N_x, N_y) = (32, 32)$. A four-level V-cycle scheme is used coupled with the five-stage explicit optimally-smoothing multi-stage time-stepping scheme with full multigrid start-up. Five-hundred cycles are used for each full multigrid stage. It can be seen in Figure E.1 that the L_2-norm of the density residual has been reduced by twelve orders of magnitude in less than 750 regular multigrid cycles. Although not shown in the figure, 10,000 iterations of the five stage explicit optimally-smoothing multi-stage time-stepping scheme without multigrid are required to reduce the residual by only four orders of magnitude.

Multigrid for Turbulent Flows

Due to numerical stiffness, the nonlinear behaviour of the source terms of turbulence models may prevent multigrid methods applied to turbulent flows from converging [180, 56, 144]. At coarse grid levels, the amplitude of the turbulence model source terms may be severely affected because of the lack of resolution to support accurate representation of the velocity gradients [180, 144]. To alleviate these issues, Gerlinger and Brüggemann and Park and Kwon [144] solve the turbulence equations on the finest mesh and restrict the nonlinear source terms to the coarse meshes. The turbulence equations are then solved on the coarse grids with constant source terms. Steelant *et al.* [180] found no benefit in solving the turbulence quantities on the coarse grids. This is the approach used in the current work as well as in the method proposed by Gao and Groth [55]. The turbulence

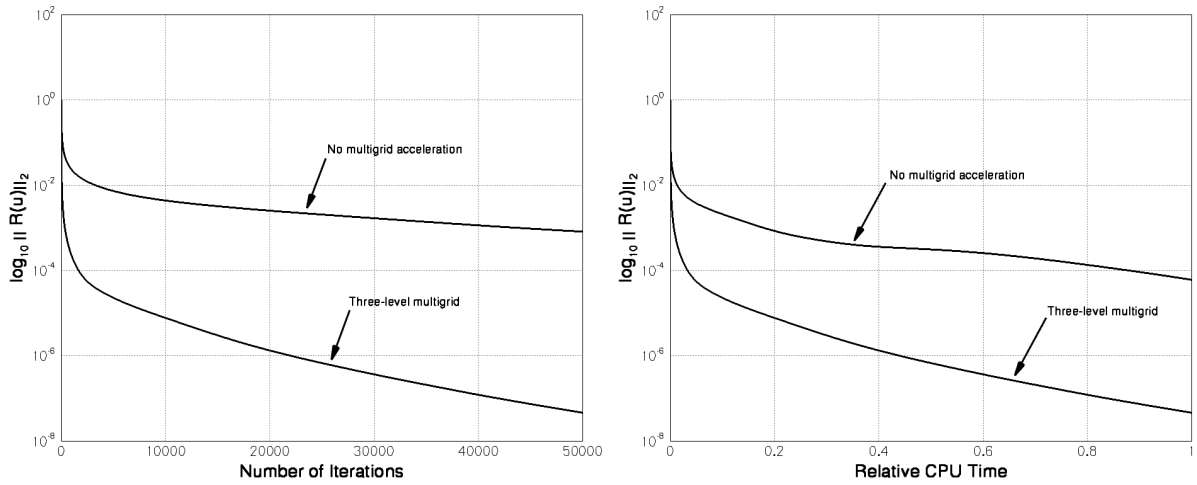


Figure E.2: *L_2-norm of axial velocity residual plotted versus the number of iterations (left figure) and the relative CPU time (right figure) for the turbulent pipe flow for computations without multigrid convergence acceleration and with three multigrid levels.*

equations are solved only on the finest mesh and the turbulence quantities are restricted to the coarse meshes but not solved.

The turbulent pipe flow results described in sub-Section 3.4.3 were computed with and without multigrid convergence acceleration. The convergence histories for these two cases are plotted in Figure E.2 versus the number of iterations (left figure) and the relative CPU time (right figure). In both cases, the five-stage explicit optimally-smoothing multi-stage time-stepping scheme is used. A three-level multigrid without any full multigrid cycles is used for the multigrid case. After 50,000 iterations, the L_2-norm of axial velocity residual has been reduced by over seven orders of magnitude with the multigrid convergence acceleration and only by three orders of magnitude otherwise. The same amount of CPU time is used by the two methods after 50,000 multigrid cycles and nearly 250,000 iterations of the time-marching scheme, however, the L_2-norm of axial velocity residual is reduced by nearly three orders of magnitude when using the three-level multigrid method.

Multigrid with Embedded Boundaries

Some modifications of the multigrid method described above are required when the mesh has been adjusted to an embedded boundary. Similar alterations are required when using multigrid with the meshes generated by the Cartesian cut-cell method [2]. The coarse grids required for the multigrid method are generated by the standard approach on the unadjusted initial mesh. The mesh on each level is then adjusted to the embedded boundary using the mesh adjustment algorithm. This method of coarse grid generation is straight-forward, however, the standard restriction and prolongation operators near the embedded boundary are no longer valid since the $(2i, 2j)$-indexing on the fine mesh may not correspond to the (i, j)-indexing on the coarser mesh as indicated in Figure E.3. Refer to Figure E.4 for illustration of the restriction and prolongation operators near the embedded boundary. The restriction operator for cells near the embedded boundary requires the calculation of the area of intersection between the coarse grid and fine grid cells allowing for an accurate area-weighted average,

$$\mathbf{U}_{i,j} A_{i,j} = \sum_m \sum_n \mathbf{U}_{2i+m,2j+n} (A_{2i+m,2j+n} \cap A_{i,j}) \,, \tag{E.5}$$

where $A_{i,j}$ and $\mathbf{U}_{i,j}$ are the area and vector of conserved solution quantities of the coarse grid cell and $A_{2i+m,2j+n}$ and $\mathbf{U}_{2i+m,2j+n}$ are the area and vector of conserved solution quantities of the contributing fine grid cells. The area of intersection between the fine and coarse grid cells, $A_{2i+m,2j+n} \cap A_{i,j}$, is determined using the Weiler-Atherton algorithm [210] which is described in Appendix A. The search space for contributing fine grid cells can be limited to $(m, n) \in [-2, 2]$.

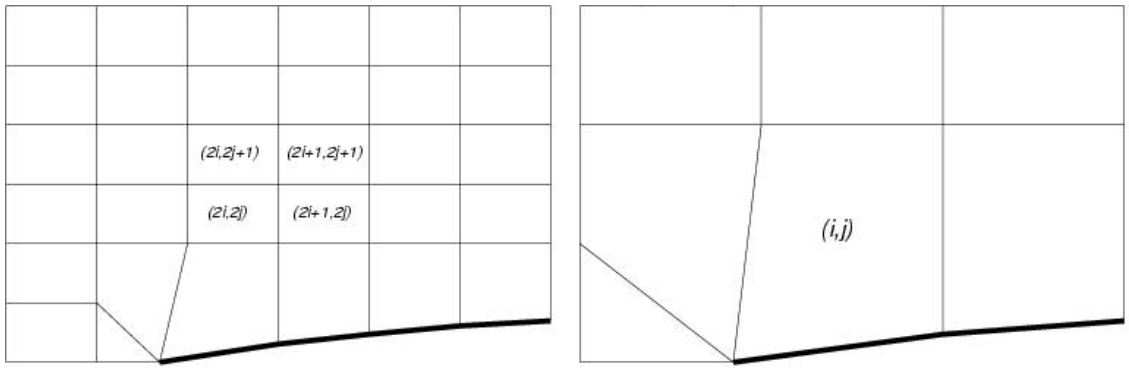


Figure E.3: *The fine and coarse grids for a two-stage multigrid method are shown in the left and right panels, respectively. The embedded boundary is indicated by the thick solid line and the corresponding indexing of the fine and coarse grids are indicated.*

The prolongation operator is reduced to injection for cells near the embedded boundary. Bilinear interpolation is used elsewhere.

The inviscid transonic flow over a semi-circular bump in a channel described above is used here to examine the performance of the multigrid method with an embedded boundary. Here a uniform Cartesian mesh with 8,192 cells is adjusted to the embedded semi-circular bump. The adjusted mesh in the region of the leading edge of the bump is shown in Figures E.3 and E.4. A four level V-cycle is used coupled with the five stage explicit optimally-smoothing multi-stage time-stepping scheme with full multigrid start-up. The convergence history for the L_2-norm of the density residual

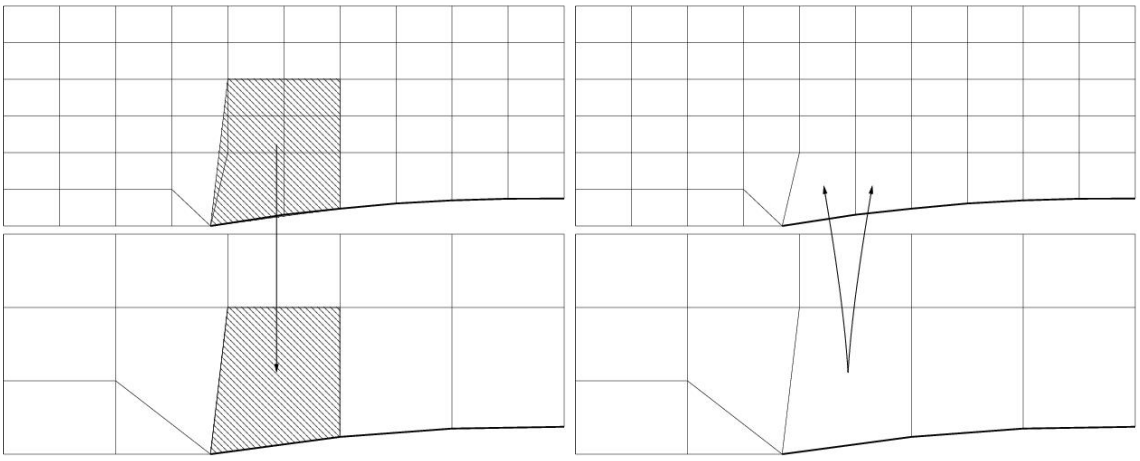


Figure E.4: *Illustration of the modified multigrid operators near the embedded boundary. The restriction operator, shown in the left side, involves the area weighted average of the conserved solution quantities of the set of fine grid cells that overlap with coarse grid cell. The prolongation operator, shown in the right side, is reduced to injection for cells near the embedded boundary.*

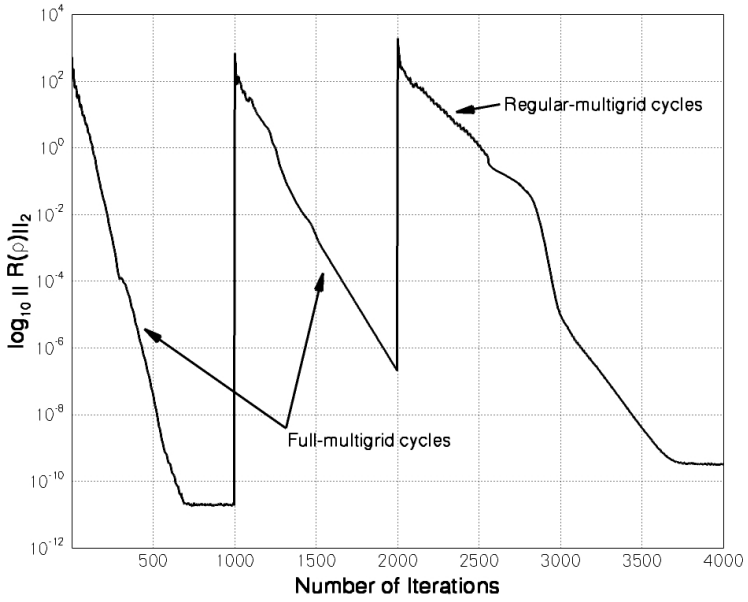


Figure E.5: *L_2-norm of density residual for the inviscid transonic flow over a semi-circular bump. The uniform Cartesian mesh contains 8 blocks and 8,192 cells and is adjusted to the location of the embedded semi-circular bump.*

is given in Figure E.5. Although the multigrid method with the modified operators was still able to reduce the L_2-norm of the density residual has been reduced by twelve orders of magnitude, over twice the number of iterations were required. This reduction in performance is thought to be attributed to the relative spareness of the mesh at the embedded boundary as compared with the body-fitted mesh. In addition, using simple injection as the prolongation operator at the embedded boundary also reduces the accuracy of the update of the coarse-grid correction to the fine grid solution.