DEVELOPMENT OF A HYBRID PARTICLE CONTINUUM SOLVER

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

by

Anthony J. Gay

March 2021

COMMITTEE MEMBERSHIP

TITLE:	Development of a Hybrid Particle Continuum Solver
AUTHOR:	Anthony J. Gay
DATE SUBMITTED:	March 2021
COMMITTEE CHAIR:	David Marshall, Ph.D. Aerospace Engineering Department Chair
COMMITTEE MEMBER:	Amelia Greig, Ph.D. Assistant Professor, University of Texas
COMMITTEE MEMBER:	Arnold Deffo, Ph.D. Professor of Aerospace Engineering
COMMITTEE MEMBER:	Robert Martin, Ph.D. Aerospace Engineer, Air Force Research Laboratory

ABSTRACT

Development of a Hybrid Particle Continuum Solver

Anthony J. Gay

When simulating complex flows, there are some physical situations that exhibit large fluctuations in particle density such as: planetary reentry, ablation due to arcing, rocket exhaust plumes, etc. When simulating these events, a high level of physical accuracy can be achieved with kinetic methods otherwise known as particle methods. However, this high level of physical accuracy requires large amounts of computation time. If the simulated flow is in collisional equilibrium, then less computationally intensive continuum methods, otherwise known as fluid methods, can be utilized. Hybrid Particle-Continuum (HPC) codes attempt to blend particle and fluid solutions in order to reduce computation time for transitional flows that exhibit both continuum and rarefied flow in a single domain. This thesis details the development of an HPC code in OpenFoam for Cal Poly's Aerospace Engineering department. The primary benchmark for the solver, named hybridFoam, was to simulate a 1D sod-shock simulation. This primary goal was achieved and a collection of test simulations were conducted to map out the solvers current capabilities and identify where future development efforts should focus.

ACKNOWLEDGMENTS

My sincere gratitude to:

- Jenna Stephens, for helping me whenever I was stuck
- David Marshall, for helping me finish strong
- Amelia Grieg, for helping me start off right
- Robert Martin, for helping me understand every concept

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

α	Fixed mesh ratio
$\bar{N}$	Time averaged number of simulated molecules in a cell
β	Reciprocal of the most probable molecular speed
χ	Angle of deflection
δ	Mean molecular spacing
ϵ	Energy mode
Γ	Gamma function
γ	Fraction of particle trajectory before intersection with cell face
λ	Molecular mean free path
μ	Coefficient of viscosity
Ω	Solid angle about resulting collisional velocity vectors
ω	Weighting coefficient
Φ	Subrelaxation parameter for Boyd and Suns' scheme to reduce statistical scattering
ϕ	Volumetric flux i.e. volume of fluid flow per second
Π_{ij}	Stress tensor
Ψ	General dependant tensor field
ρ	Mass density

σ	Molecular cross section
σ_T	Total collisional cross section
$\mathbf{c}$	Molecular velocity, i.e. position in velocity space
$\mathbf{c}'$	Thermal or random molecular velocity tensor
$\mathbf{c}'_0$	Thermal or random molecular velocity tensor of a maxwellian distribution
$\mathbf{F}$	External body force
$\mathbf{q}$	Rate of heat exchange
$\mathbf{r}$	Position vector in physical space
$\mathbf{S}$	Cell face normal vector
$\mathbf{S}_f$	Normal face vector pointing out of face owner cell
$\mathbf{v}$	Bulk fluid velocity
θ	Most probable molecular speed
a_j	Maximal local speed
b	Miss distance impact parameter
C_f	Cell face center
c_r	Relative molecular speed
d	Diameter of a molecular species
d_{ij}	Combined diameter of collisional pair ij
E_t	Total energy per unit volume
F	Flux of a conserved quantity

f Subscript denotes property evaluated at cell face

$f(u)$ Nonlinear convection flux

$f, f(c)$ Velocity distribution function

f_0 Maxwellian or equilibrium velocity distribution function

F_N The number of real molecules represented by one virtual molecule

$H_j(t)$ Numerical flux

K Coefficient of heat conduction

k Boltzmann constant

Kn Knudsen number

L Control volume characteristic length

l_{SV} Sampling volume characteristic length

m Molecular mass

N Total number of molecules; Total number of simulated molecules

n Number density, superscript denotes current time step

N_{sub} Number of sub-cycles for the LD-DSMC method

P Probability function

Q Net heat produced per unit volume; Macroscopic parameter of interest

T_{ref} reference temperature

T_{ROT} Total rotational temperature

T_{TRA} Total translational temperature

u	Approximate value of a conserved quantity
u, v, w	Dimensions of velocity space
v	Relative speed exponent of the VHS model
V_C	Cell Volume
w	Macroscopic viscosity temperature exponent, Relaxation coefficient
x, y, z	Dimensions of physical space
CFL	Courant number
r	Slope
R(r)	Slope limiting function

Chapter 1

INTRODUCTION

1.1 Motivation

When modeling gas flow, the method utilized often depends on the physical nature of the simulation. For continuum flow, where mean free path of particles is small in relation to the local scale length such that molecular thermal equilibrium is achieved, fluid methods, such as a Navier Stolkes solver, produce accurate results efficiently. For lower density rarefied flow, fluid solvers produce physical inaccuracies and more computationally expensive particle methods, such as a direct simulation monte carlo (DSMC), are required. However, during dynamic transient flows, where length scales and the average mean free path of particles can change rapidly, it becomes necessary to create a solver that can break down a simulation domain and delegate sections of flow to the appropriate solver. This is the motivation behind hybrid particle continuum (HPC) codes.

Several scenarios exhibit transitional flow including Re-entry vehicles, which undergo highly dynamic hypersonic flows. Typically, as the vehicle re-enters the atmosphere, the majority of the flow can be considered continuum. However, typical flow length scales within the thin bow shock and boundary layer are small enough to be considered rarefied, while the average mean free path in the wake region is also sufficiently large to indicate continuum breakdown [6]. Typically these regions are not significantly affected by the inaccuracies of a CFD solver, however Wright et al. [35] attempted to analyse the heat transfer of the AS-202 flight case from the Apollo program with a CFD solver. The capsule had a unique 'skip' trajectory where the vehicle used the

lift from the initial re-entry to fly out of atmosphere before making its final descent. During the crest of the skip re-rentry, Wright et al. found that the CFD solver over predicted heating on the lee side of the aft-body by a factor of two when compared to the Apollo data. They concluded this inaccuracy could be due to the solvers inability to capture microscopic effects and this particular flow would have required significant computation time to model if a particle method was used [6, 35]. In this case, a hybrid solver would have been the ideal software tool.

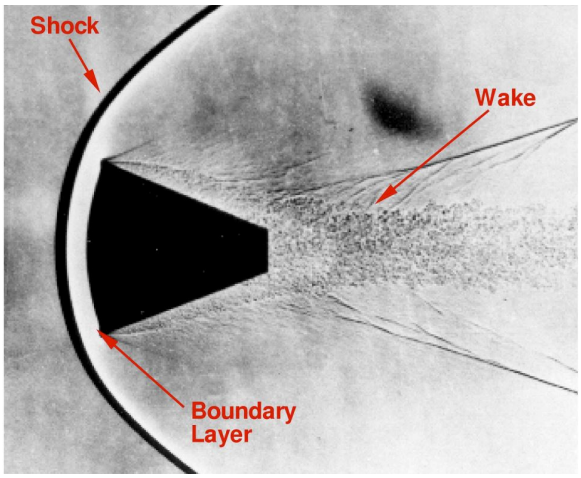


Figure 1.1: Schematic of hypersonic flow over a blunted body with regions that typically exhibit non-continuum flow [6]

Thruster exhaust plumes are another scenario that exhibit transitional flow. As plumes expand into a vacuum, particle density and collisional frequency change rapidly, resulting in continuum flow near the thruster and rarefied flow deeper in the plume. The Northrop Grumman Cygnus spacecraft has 32 monopropellant thrusters for its guidance navigation and control system, and in order to better understand the momentum loss due to thruster plume impingement, a research group at Northrop Grumman Innovation Systems is developing hybrid modeling capabilities. Their model utilizes a continuum breakdown parameter first developed by Bird [4] to divide the continuum and rarefied regimes. A CFD solver from Mississippi State university called CHEM solves for flow behavior in the continuum regime and provides the inflow

parameters for the rarefied regime, which is modeled using NASA's DSMC analysis code (DAC). It is in the rarefied regime where plume impingement is modeled with the molecular level accuracy of DAC. With this hybrid architecture the group hopes to steadily improve the accuracy of their thrust predictions for the Cygnus control systems [3].

1.2 Goals and Objectives

The primary objective of this thesis was to develop an initial HPC code, called hybridFoam, for the California Polytechnic State University, San Luis Obispo (Calpoly) Aerospace department. The code was developed using the Open Source Field Operation and Manipulation (OpenFOAM) framework. The primary benchmark of the solver was to solve a 1D sod-shock scenario with static fluid and particle regions. Five additional tests were performed to further validate hybridFoams performance and identify areas that require improvement during future development. The results of this effort will facilitate future research of HPC codes at Calpoly and contribute to the eventual capability of analyzing transitional flows.

Chapter 2

BACKGROUND

2.1 Continuum Methods

2.1.1 Governing Equations

Before discussing Hybrid solvers it is necessary to have an understanding of the two distinct families of solvers they contain; fluid and particle. All fluid solvers fall under the umbrella of computational fluid dynamics (CFD). The purpose of any CFD code is to solve fluid flow problems which are in turn governed by the Navier-Stokes (NS) equations or a simplification of the NS equation set such as Euler or full potential equations. In their most general form, the NS equations are as follows:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{2.1}$$

$$\frac{\partial}{\partial t}(\rho \mathbf{v}) + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = \rho \mathbf{F} + \nabla \cdot \Pi_{ij} \tag{2.2}$$

$$\frac{\partial E_t}{\partial t} + \nabla \cdot (E_t \mathbf{v}) = \frac{\partial Q}{\partial t} - \nabla \cdot \mathbf{q} + \rho \mathbf{F} \cdot \mathbf{v} + \nabla \cdot (\Pi_{ij} \cdot \mathbf{v}) \tag{2.3}$$

In equation 2.1, the first term in the equation represents change of mass within the control volume (CV) while the second term represent the mass flux across the various control surfaces of the CV. In equation 2.2 the first term is the change of momentum within a CV and the second term represents the change of momentum due to convection across control surfaces. On the right hand side of the equation, $\rho\mathbf{F}$ is the body force per unit volume while the $\nabla \cdot \Pi_{ij}$ term represents the surface forces per unit volume. Body forces are any forces that act at a distance and apply

to the entire CV such as gravity and surface forces are the result of stresses on the fluid element. Normal and shearing stresses are represented by the components of Π_{ij} [1]. Lastly, in the equation 2.3, the first term on the left hand side the rate of change of energy within the CV and the second representing the energy change via convection through control surfaces. On the right hand side, the first term is the rate of heat change in the CV while the second is heat exchanged across control surfaces via conduction. The third term on the right hand side is the work done on the CV by body forces and the fourth term is the work done on the CV by surface forces [1].

Each of these equations are based on universal laws of conservation. Equation 2.1 is the result of applying conservation of mass to fluid flow in a control volume, equation 2.2 is the same application but with Newton's Second Law resulting in the conservation of momentum equation, while equation 2.3 is the conservation of energy and identical to the First Law of Thermodynamics [1]. It is also important to note that in this form, the NS equations do not form a closed set, and must be simplified by applying assumptions such as: an ideal gas, a calorically perfect gas, negligible body forces, etc. Many of these assumptions exist to provide relations between the unknowns such as pressure, density and temperature, and transport coefficients such as the coefficient of thermal conductivity and viscosity. These additional equations are known as equations of state, due to their relation to the state principle of thermodynamics [1].

2.1.2 Kurganov and Tadmor Central Scheme

There are several methods that exist to solve the NS equation set however for the sake of brevity, this thesis will focus on the central scheme developed by Kurganov and Tadmor (KT). Central schemes are akin to central differencing schemes in that they compute the convective flux of a conserved variable at a face of a control volume from the cell average of that variable on both sides of the face. Typically these schemes

are coupled with a slope limiting scheme such as minmod or SuperBee, to introduce artificial dissipation and minimize oscillations around shocks [5]. The first central scheme was introduced in 1954 by Lax and Friedrichs and is given by

$$u_j^{n+1} = \frac{u_{j+1}^n + u_{j-1}^n}{2} - \frac{\alpha}{2}[f(u_{j+1}^n) - f(u_{j-1}^n)] \tag{2.4}$$

where α is given by $\frac{\Delta t}{\Delta x}$ and u_j^n is an approximate value of a conserved quantity at the grid-point $x_j = j\Delta x, t^n = n\Delta t$. Equation 2.4 shares advantages common to central schemes such as simplicity and the absence of approximate Riemann solvers, which can be computationally expensive in that they require the resolution of Riemann fans by staggered (x,t) integration. The KT scheme retains this simplicity and achieves accuracy comparable to alternative schemes, including Riemann solvers, by decreasing numerical viscosity in order to more effectively resolve shocks [17].

The KT scheme can be applied to the NS equation set to provide an approximate solution, however to discuss how, it is useful to recast the NS equation set into a more generic form such as a set of nonlinear conservation laws of the form

$$\frac{\partial}{\partial t}u(x,t) + \frac{\partial}{\partial x}f(u(x,t)) = 0 \tag{2.5}$$

where $u(x,t) = (u_1(x,t), ..., u_N(x,t))$ is an N-vector of conserved quantities i.e. mass, momentum and energy [17].Using the same nomenclature the semi discrete compact form ($\Delta t \to 0$) of the KT scheme is as follows

$$\begin{aligned}\frac{d}{dt}u_j(t) = &-\frac{(f(u_{j+1/2}^+(t)) + f(u_{j+1/2}^-(t))) - (f(u_{j-1/2}^+(t)) + (f(u_{j-1/2}^-(t))))}{2\Delta x} \\ &+ \frac{1}{2\Delta x}\{a_{j+1/2}(t)[u_{j+1/2}^+(t) - u_{j+1/2}^-(t)] - a_{j-1/2}(t)[u_{j-1/2}^+(t) - u_{j-1/2}^-(t)]\}\end{aligned} \tag{2.6}$$

where the terms $u^{\pm}_{j+1/2}$ are defined by

$$u^{+}_{j+1/2}(t) = u_{j+1}(t) - \frac{\Delta x}{2}(u_x)_{j+1}(t) \tag{2.7}$$

$$u^{-}_{j+1/2}(t) = u_{j+1}(t) + \frac{\Delta x}{2}(u_x)_{j+1}(t) \tag{2.8}$$

where $(u_x)_j(t)$ are the numerical derivatives reconstructed from the computed cell averages, $u_j(t)$. It is important to note that 2.6 is a second order scheme and that in general, for first and second order schemes', cell averages can be identified with their corresponding point values. Kurganov and Tadmor therefore use cell averages interchangeably with point values, and their approximate solution for $u_j(t)$ is a cell averaged value despite the absence of a typical bar nomenclature such as $\bar{u}_j(t)$ [17]. In the more approachable conservative form 2.6 becomes

$$\frac{d}{dt}u_j(t) = -\frac{H_{j+1/2}(t) - H_{j-1/2}(t)}{\Delta x} \tag{2.9}$$

with the numerical flux, $H_j(t)$, defined as

$$H_{j+1/2}(t) = \frac{f(u^{+}_{j+1/2}(t)) + f(u^{-}_{j+1/2}(t))}{2} - \frac{a_{j+1/2}(t)}{2}[u^{+}_{j+1/2}(t) - u^{-}_{j+1/2}(t)]. \tag{2.10}$$

Here the intermediate values $u^{\pm}_{j+1/2}$ are given by

$$u^{+}_{j+1/2} = u_{j+1}(t) - \frac{\Delta x}{2}(u_x)_{j+1}(t) \tag{2.11}$$

$$u^{-}_{j+1/2}(t) = u_j(t) + \frac{\Delta x}{2}(u_x)_j(t). \tag{2.12}$$

It is the conservative form, 2.10, of the KT scheme that would be applied to solve the NS equations along with a slope limiting scheme. Slope limiters are used to avoid spurious oscillations in high order schemes and to satisfy the total variation diminishing

(TVD) criteria, meaning that the total variation of the discrete solution will diminish with time. Such oscillations can occur when modeling shocks or discontinuities. To avoid this, the KT scheme is often combined with a minmod slope limiter, where

$$minmod(a, b) = \frac{1}{2}[sgn(a) + sgn(b)] \cdot min(|a|, |b|) \tag{2.13}$$

although several slope limiters exist [17, 32]. In the case of this thesis, the slope limiter used was the Van Albada slope limiter, which for 1D space is given by

$$R(r) = \frac{2}{1 - r^2} \tag{2.14}$$

where $r \geq 0$ [21, 31]. When $r < 0$ then $R(r) = 0$.

2.1.3 The Continuum Assumption

While the NS equations and CFD solvers, such as the KT scheme, are applicable to many physical situations, there are underlying assumptions that can breakdown when modeling certain flows. The continuum assumption is one such characteristic. If a flow is a continuum, it is assumed that there are enough collisions between particles such that microscopic fluctuations do not have a significant impact on macroscopic averaged quantities [8]. Local macro-properties can then be described as averages over elements that are large compared to the microscopic structure of the fluid, but small enough with respect to the macroscopic phenomena to permit the use of differential calculus[34]. This assumption then relies on two distinct criteria to be met for a sampling volume; that the volume is in thermodynamic equilibrium and that statistical fluctuations can be ignored. For statistical fluctuations to be ignored the ratio of a sampling volume characteristic length must be significantly larger that the mean molecular spacing. Consequently the characteristic length of the control vol-

ume must be much larger. In an example taken from [8], a volume containing 10,000 molecules leads to 1% fluctuations in the macroscopic quantities. A fluctuation of this level requires an l_{SV} such that $l_{SV}/\delta = 10^{\frac{4}{3}}$. Therefore

$$\frac{L}{\delta} >> 10^{\frac{4}{3}} \tag{2.15}$$

and statistical fluctuations can be neglected. For a sampling volume to be in thermodynamic equilibrium, the mean free path of molecules must be small such that

$$\frac{\lambda}{L} << 1. \tag{2.16}$$

This ratio is known as the Knudsen number, and is typically the metric used when determining if a flow is a continuum. A low Knudsen number indicates inter-molecular collisions are dominant, meaning there are enough particle collisions for thermodynamic equilibrium to be reached in a very short time compared to the macroscopic time scale [34]. If this is not true, then shear stresses and heat flux (the transport terms) can no longer be correctly expressed in lower order macroscopic terms, preventing the NS conservation equations from forming a closed set [4]. Figure 2.1 shows the threshold values proposed by [4] for equations 2.15 and 2.16 as well an additional criteria for defining a dilute and dense gas. If the continuum assumption is valid, then statistical fluctuations can be ignored and variations in macroscopic quantities are thought of as continuous with respect to the scale of the sampling volume. If this assumption is not valid for a flow, then applying a CFD solver would introduce significant artificial smoothing, whereas applying a solver that accounted for micro-scale variations, such as a particle based solver, would yield results closer to the actual behavior of the flow.

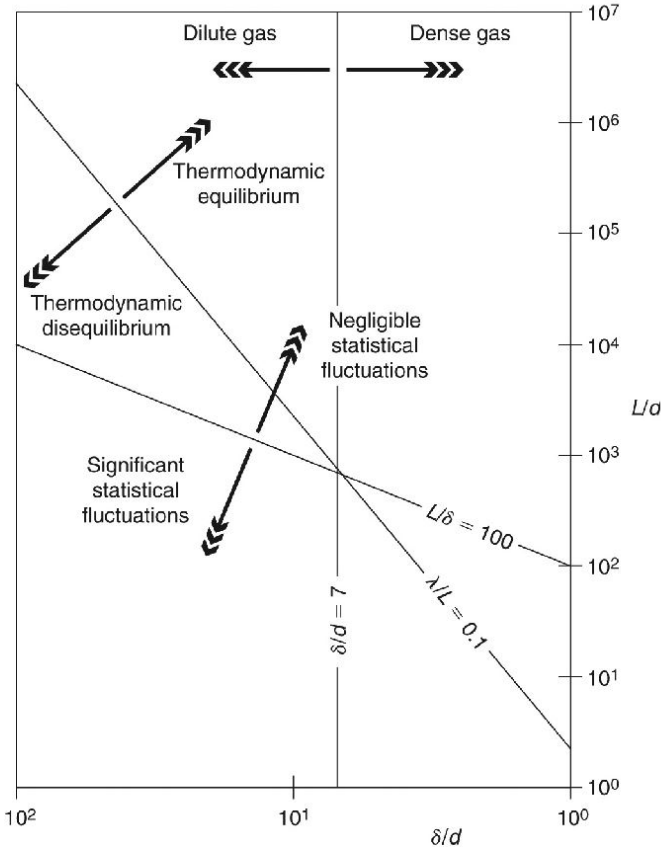


Figure 2.1: Limits of the mean approximations for modeling gas flows [8]

Figure 2.2 provides a more detailed breakdown of the distinct flow regimes determined by the Knudsen number. As $Kn \to 0$ molecular diffusion can be ignored, nullifying the transport terms in 2.2 and 2.3, and reducing the NS equations to the inviscid Euler equation set [34]. A regime not shown in figure 2.2 is the slip flow regime, which is typically placed at $0.001 < Kn < 0.1$. The slip flow regime is where the NS equations are valid within the flow but rarefied areas begin to appear near surfaces. This can be observed from a macroscopic point of view as the fluid velocity and temperature at a surface not obtaining the same values as the surface itself. This is known as velocity slip and temperature jump and can be accounted for within the NS framework by using Maxwells velocity slip and Von Smoluchowski's temperature-jump boundary conditions [34]. As the Knudsen number increases to the right past the

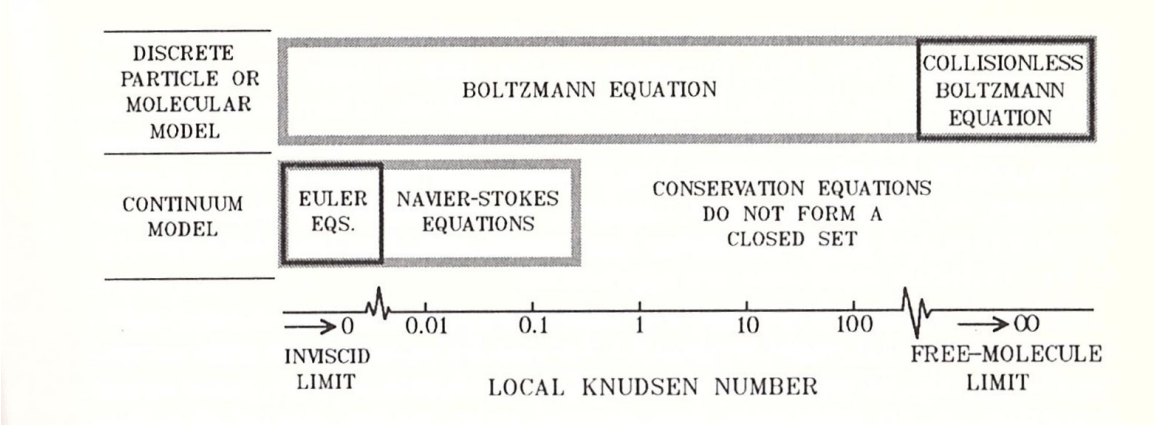


Figure 2.2: Knudsen number regimes [4]

continuum region, particle-particle and particle-surface collisions begin to dominate flow behavior and the NS equations no longer correctly form a closed set. Instead an alternate solution to the parent equation of the NS equations, the Boltzmann equation, must be found. This is accomplished through particle methods, such as the direct simulation monte carlo (DSMC) method detailed in 2.2.

2.2 Particle Methods

2.2.1 Governing Equation

The Boltzmann equation is, in essence, a general description of particle movement through an arbitrary volume. The following section will derive an expression for the Bolztmann equation from this concept, following the process detailed in chapter 3 of Bird [4]. Such a description of particle motion relies on two concepts; the phase space and the velocity distribution function. Where a position in physical space would be denoted by a 3 element vector $\mathbf{r}$ with components x, y and z, a position in velocity space can be defined in the same way, by a vector $\mathbf{c}$ with components u, v, and w. After using this notation to describe a physical space that contains N homogeneous

molecules, the velocity distribution function $f(\mathbf{c})$ is defined by

$$dN = Nf(\mathbf{c})dudvdw \tag{2.17}$$

where dN is the number of molecules in the space with velocity components u to $u + du$, v to $v + dv$, and w to $w + dw$[4]. Alternatively equation 2.17 can be rewritten by omitting the $f(c)$ notation and combining the velocity space terms to form the volume element dc.

$$dN = Nfd\mathbf{c} \tag{2.18}$$

However, macroscopic flow properties are generally functions of position and time, so equation 2.18 is not sufficient to describe a range of particles with specific velocities as well as positions. This combination of physical and velocity space is called phase space and has a volume element defined by $d\mathbf{c}d\mathbf{r}$. Now it is possible to derive a general expression for the Boltzmann equation. At a particular instant, the rate of change of the number of molecules in a constant phase space volume element is given by

$$\frac{\partial}{\partial t}(nf)d\mathbf{c}d\mathbf{r}. \tag{2.19}$$

The processes that contribute to 2.19 are: (i) the convection of molecules across the face of $d\mathbf{r}$ by the molecular velocity $\mathbf{c}$, (ii) the convection of molecules across the surface of $d\mathbf{c}$ as a result of external forces $\mathbf{F}$, and (iii) the scattering of molecules into and out of $d\mathbf{c}d\mathbf{r}$ from intermolecular collisions. Process (i) can be expressed with equation 2.20

$$\int_{d\mathbf{r}} \nabla \cdot (nf\mathbf{c})d(d\mathbf{r})d\mathbf{c} \tag{2.20}$$

where $\int_{d\mathbf{r}} d(d\mathbf{r})$ is a volume integral over $d\mathbf{r}$. As n, f and $\mathbf{c}$ are constant within $d\mathbf{r}$ equation 2.20 can be written as

$$\nabla \cdot (nf\mathbf{c}) d\mathbf{r} d\mathbf{c}. \tag{2.21}$$

Additionally, when considering only molecules of class $\mathbf{c}$, the velocity term can be excluded from the divergence operator. Therefore the flow of molecules of class $\mathbf{c}$ across the surface of $d\mathbf{r}$ due to velocity is

$$\mathbf{c} \cdot \frac{\partial(nf)}{\partial \mathbf{r}} d\mathbf{c} d\mathbf{r}. \tag{2.22}$$

Following this same method for process (ii), the flow of molecules across the surface of $d\mathbf{c}$ from external forces is given by

$$\mathbf{F} \cdot \frac{\partial(nf)}{\partial \mathbf{c}} d\mathbf{c} d\mathbf{r}. \tag{2.23}$$

Lastly, the molecules scattered into and out of element $d\mathbf{c}d\mathbf{r}$ via collisions can be obtained after considering the two particle velocity classes in question; before ($\mathbf{c}$) and post collision ($\mathbf{c}^*$). In addition to this, the DSMC method is concerned with a dilute gas, so non-binary collisions are considered negligible. Collisional pairs may then be represented with $\mathbf{c}, \mathbf{c}_1 \rightarrow \mathbf{c}^*, \mathbf{c}_1^*$. By considering a test particle of class $\mathbf{c}$ with a speed c_r travelling among a field of stationary molecules of class $\mathbf{c}_1$, the number of class $\mathbf{c}_1$ collisions per unit time is given by

$$n f_1 c_r \sigma d\Omega d\mathbf{c}_1. \tag{2.24}$$

The term $c_r \sigma d\Omega$ is the volume swept out in physical space by this class of collision and $n f_1 d\mathbf{c}_1$ is number of $\mathbf{c}_1$ molecules per unit volume. When considering class

$\mathbf{c}, \mathbf{c}_1 \rightarrow \mathbf{c}^*, \mathbf{c}^*$ collisions in this same manner but in phase space, equation 2.24 becomes

$$n^2 f f_1 c_r \sigma d\Omega d\mathbf{c}_1 d\mathbf{c} d\mathbf{r} \tag{2.25}$$

Now the rate of increase of molecules of class $\mathbf{c}$ in the phase space element $d\mathbf{c}d\mathbf{r}$ as the result of both inverse and direct collisions of class $\mathbf{c}, \mathbf{c}_1 \rightarrow \mathbf{c}^*, \mathbf{c}^*$ is obtained by subtracting the original velocity distribution terms from the starting distributions. This can then be integrated over all velocity space to give the total rate of increase of class $\mathbf{c}$ molecules from collisions with class $\mathbf{c}_1$ molecules, giving

$$\int_{-\infty}^{\infty} \int_{0}^{4\pi} n^2 (f^* f_1^* - f f_1) c_r \sigma d\Omega d\mathbf{c}_1 d\mathbf{c} d\mathbf{r} \tag{2.26}$$

Finally, the total rate of increase of class c molecules from processes (i), (ii), and (iii) can be obtained by combining 2.22, 2.23, and 2.26. If the results of processes (i) and (ii) are limited to inflow terms and process (iii) is limited to representing particles scattering out of a phase space volume due to collisions, then this gives the Boltzmann equation for a simple dilute gas

$$\frac{\partial}{\partial t}(nf) + \mathbf{c} \cdot \frac{\partial}{\partial \mathbf{r}}(nf) + \mathbf{F} \cdot \frac{\partial}{\partial \mathbf{c}}(nf) = \int_{-\infty}^{\infty} \int_{0}^{4\pi} n^2 (f^* f_1^* - f f_1) c_r \sigma d\Omega d\mathbf{c}_1. \tag{2.27}$$

As can be seen in figure 2.2, equation 2.27 is only valid up to the free molecule limit, where particle collisions are considered negligible. It is also important to note that there is no analytical solution to the Boltzmann equation except for unique cases [4]. In comparison to the NS equation set, the Boltzmann equation does have nf as the only dependant term on the right hand side while the NS equations have velocity components and two thermodynamic properties as dependant variables, when allowing for state equations. However when considering a one dimensional homogeneous gas problem, the velocity distribution function f becomes spherically symmetrical in ve-

locity space and axially symmetric in physical space, resulting in a three dimensional problem. For an unsteady three dimensional flow, f has no symmetries in velocity space. After including time as an additional dimension, the problem becomes seven dimensional, drastically increasing the Boltzmann equations analytical complexity [4]. In place of an analytical solution to complex problems such as these, the DSMC method offers a numerical solution to equation 2.27.

2.2.2 The DSMC Algorithm

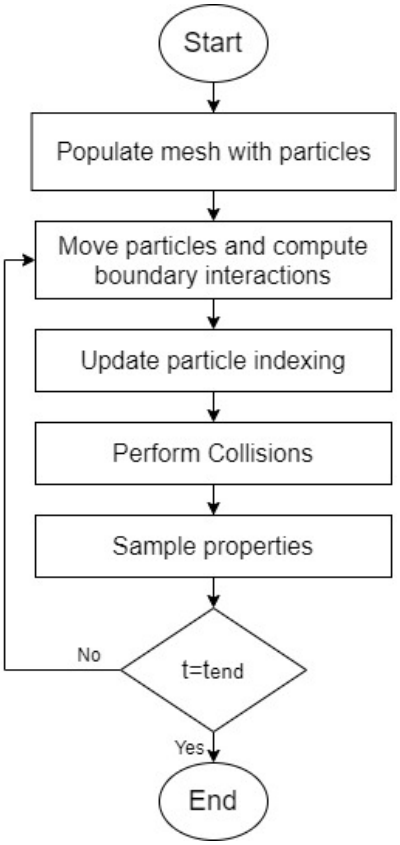


Figure 2.3: A typical DSMC flowchart

First invented by GA Bird in 1976, the algorithm shown in figure 2.3 is typical for any DSMC solver. It begins with an initialization phase where particle locations, densities, velocities are defined as well as gas species information such as internal degrees of freedom and mass. After the domain is populated with particles, the solver loop starts

with moving particles by $\mathbf{c}(t)\Delta t$ and tracking them. The complexity of the particle tracking process is determined by the mesh involved, as meshes involving non-planar cell faces require special consideration in order to maintain computational efficiency [19]. For a simple uniform Cartesian grid, an example particle path can be seen in figure 2.4 where a particle travels from point **a** in cell A to **b** in cell B while crossing two cell faces along its path. In the first portion of its trajectory from **a** to **p**, point p is found using

$$\mathbf{p} = \mathbf{a} + \gamma_{\mathbf{a}}(\mathbf{b} - \mathbf{a}) \tag{2.28}$$

where $\gamma_{\mathbf{a}}$ is the fraction along the line **ab** where **ab** intersects the plane defined by the face center and the face normal vector. Because **p** lies on this plane

$$(\mathbf{p} - \mathbf{C}_f) \cdot \mathbf{S} = 0 \tag{2.29}$$

giving

$$\gamma_{\mathbf{a}} = \frac{(\mathbf{C}_f - \mathbf{a}) \cdot \mathbf{S}}{(\mathbf{b} - \mathbf{a}) \cdot \mathbf{S}}. \tag{2.30}$$

γ_a is calculated for each face, with the lowest value on the interval $0 \leqq \gamma_a \leqq 1$ indicating the cell face crossed by the particle. The particle is moved to **p** and its occupancy information is changed based on cell connectivity logic, meaning that since cell A shares face 2 with cell C and the particle crossed face 2, the particle now occupies cell C [19]. The same process would then be used for the crossing at **p'**. It is also at this stage where boundary conditions (BCs) are applied. Where the process in figure 2.4 would apply to internal cells, boundary cells will have their own restraints on particle motion. For example a common BC is a specular wall, where upon crossing a cell face shared with a specular wall boundary, the normal component of a particles velocity vector switches directions, simulating specular reflection [34].

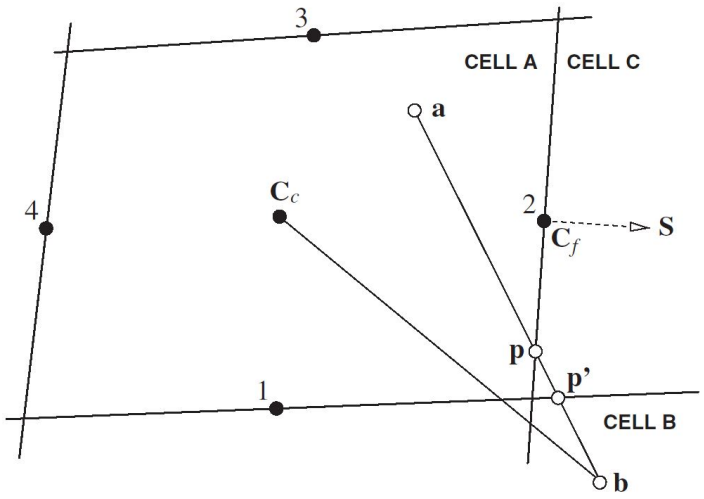


Figure 2.4: Particle tracking diagram with cell center, C_c, and trajectory from point b to point a shown [19]

After particles are moved appropriately and particle indexes are updated to reflect their location in the mesh, the probability and outcomes of particle collisions are computed. This stage sets the DSMC method apart from other particle methods, namely Molecular Dynamics. It is also important to note that two procedural techniques are used to simplify the simulation of trillions of particles. In order to facilitate the selection of individual collisional pairs, cells are broken up into subcells and virtual super-particles are used to represent a collection of real particles. Only a portion of these super particles are then selected for collisional analysis. After they are selected, whether or not the pair collide is based on a probability proportional to the product of their total collisional cross sections, σ_T, and their relative speed, c_r. The number of virtual particle pairs selected from a cell at a time step is given by

$$\frac{1}{2}N\bar{N}F_N(\sigma_T c_r)_{max}\Delta t/V_C. \tag{2.31}$$

The value of N is given by $N = nV_C/F_N$ where n is the real number density. The collisional pairs then collide based on the probability of

$$\frac{\sigma_T c_r}{(\sigma_T c_r)_{max}}. \tag{2.32}$$

The value of σ_T depends on the molecular model used. For the hard sphere model it is defined as

$$\sigma_T = \int_0^{4\pi} \sigma d\Omega = \pi d_{12}^2 \tag{2.33}$$

where d_{12} is the combined diameter of a collisional pair. For the variable hard sphere model used in this thesis, the value of d is a function of c_r given by

$$d = d_{ref} \left(\frac{c_{r,ref}}{c_r} \right)^v \tag{2.34}$$

where v is a constant and the subscript ref denotes a reference value. After a collision is confirmed, an instantaneous velocity change is applied to the molecules involved. The deflection angle is found via

$$\chi = 2cos^{-1}(b/d) \tag{2.35}$$

which in turn gives the resulting velocity vector($\mathbf{c}_r^*$) visible in figure 2.2.2 [4]. The method detailed by equations 2.31 and 2.32 is called the no-time-counter (NTC) method and is a successor to the time-counter method originally developed by Bird in 1976. The NTC method avoids calculating the probability of every collisional pair colliding as, on average, the probability of collision is typically low. As the number of collisional pairs is proportional to the number of virtual particles squared, a time counter method would have a computation time proportional to N^2 whereas the NTC method has a computation time that responds linearly with a change in N [4].

Despite the advantage of the NTC method, it still suffers from having a computation time that is dependant on the number of particles. This is the primary weakness of DSMC methods, and is what makes them significantly slower than fluid methods. Even though a fraction of real particles are considered, there are still millions of calculations required to solve the majority of physical scenarios at a useful resolution. It is also important to note that there are two primary contributors to the computation time of DSMC methods: the number of virtual particles and the number of particle collisions. As particle density increases, not only does the number of virtual particles increase (assuming a constant real to virtual particle ratio) but the likelihood of a collision increases, requiring additional computational resources to determine the resulting velocities.

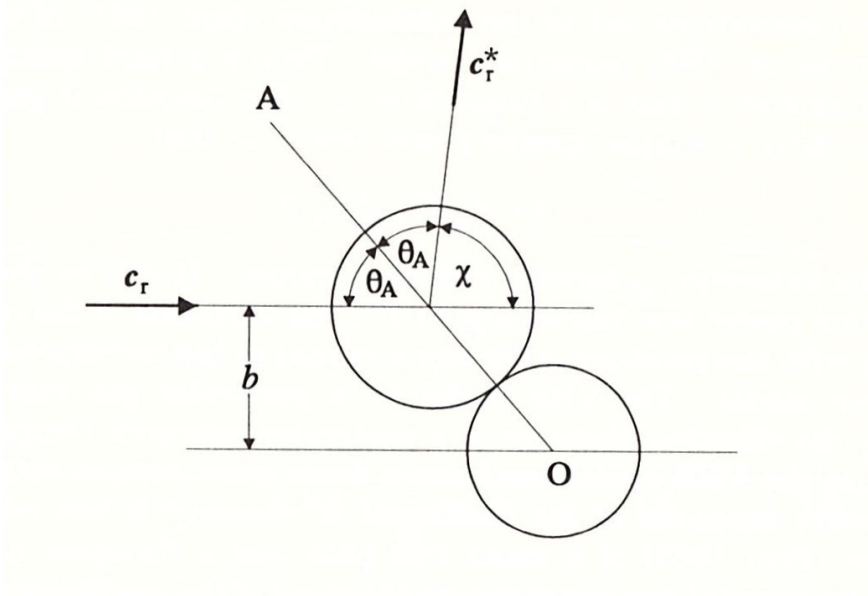


Figure 2.5: Collision geometry of hard sphere molecules [4]

2.3 Hybrid Particle Continuum Methods

2.3.1 Elements of HPC Codes

In flows that have a range of Knudsen numbers from continuum to rarefied regimes, using either a NS or DSMC solver for the entire domain would either be computationally inefficient or physically inaccurate. This is the primary motivation behind hybrid particle continuum (HPC) codes. An HPC code consists of a few major elements: the particle and continuum solvers, a breakdown parameter, and a coupling method. Historically HPC codes have used Euler or NS solvers in combination with a DSMC solver in order to analyse transitional flow as the combined solvers cover a very large range of Knudsen numbers. However, different hybrid codes exist to study narrower Knudsen number ranges such as hybrid CFD codes that combine Euler and full potential solvers [25]. What all hybrid codes have in common is that they delegate different portions of the simulation domain to the solver that is most appropriate in order to increase computational efficiency. For HPC codes, the domain is designated either continuum or particle according to a breakdown parameter, which is an empirical value that predicts the breakdown of the continuum assumption. Several breakdown parameters exist and most are based on the Chapman-Enskog distribution. During transitional flows, there is a point where molecular velocity distributions begin to deviate from the Maxwellian distribution inherent within a continuum. When this perturbation is small, the velocity distribution can be predicted by the Chapman-Enskog distribution [4]. This distribution is given by

$$f = f_0(1 - \frac{4K\beta^2}{5nk}(\beta^2 c'^2 - 5/2)\mathbf{c}' \cdot \frac{\partial(lnT)}{\partial \mathbf{r}} - \frac{4\mu\beta^4}{\rho}\mathbf{c}'^0\mathbf{c}' : \frac{\partial \mathbf{c}_0}{\partial \mathbf{r}}) \tag{2.36}$$

where the reciprocal of the most probable molecular speed, β, is given by $\beta = (2RT)^{-1/2}$ and the superscript 0 on the thermal velocity tensor, $\mathbf{c}'_0$, indicates the sum of the diagonal is zero. The coefficient of viscosity, μ, is defined by the collisional model in use, which in turn defines the diameter of molecules in collisional pairs. For the variable hard sphere model used in this thesis, the coefficient of viscosity is defined by

$$\mu = \frac{(15/8)(\pi mk)^{1/2}(4k/m)^v T^{1/2+v}}{\Gamma(4-v)\sigma_{T,ref} c_{r,ref}^{2v}} \tag{2.37}$$

where total collisional cross section, σ_T, is given by equation 2.33. Due to its validity at the start of continuum breakdown, the Chapman-Enskog distribution can offer a point of conversion between NS and DSMC domains as well as predict where the breakdown will occur. A breakdown parameter based on the Chapman-Enskog distribution was proposed by Boyd [6]. The parameter is called a gradient length Knudsen number and is given by

$$Kn_{GL_Q} = \lambda \left| \frac{\Delta Q}{Q} \right|. \tag{2.38}$$

where Q is some macroscopic parameter of interest. λ, the mean free path, is found via

$$\lambda = \frac{1}{\sqrt{2} n \sigma_{ref}} \left(\frac{T_{TRA}}{T_{ref}}^{w-1/2} \right) \tag{2.39}$$

where

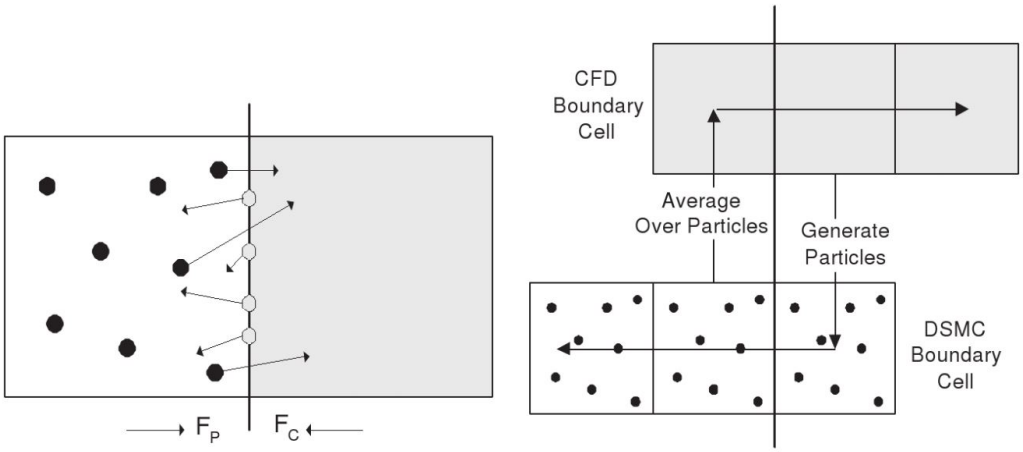
$$\sigma_{ref} = \frac{15\sqrt{\pi mkT_{ref}}}{2(5-2w)(7-2w)\mu_{ref}}. \tag{2.40}$$

Above, T_{ref} is the temperature that the reference cross section, σ_{ref}, is calculated at, which is consistent with the variable hard sphere model [6]. It is also important to note that μ_{ref} is the coefficient of viscosity at T_{ref}. This breakdown parameter predicts the deviation of CFD results from full DSMC data and a Kn_{GL_Q} value of 0.05 indicates the difference of over 5%. For hybrid simulations, Boyd found that a more

relaxed value of 0.1 as a breakdown parameter produced results with good agreement with a full DSMC simulation [6]. A variation on the gradient length Knudsen number was also proposed by Schwartzentruber [24]. As part of thermal equilibrium being a criteria for a continuum, rotational and transnational molecular temperatures must be in equilibrium. In near equilibrium flows over blunt bodies the two energy modes can deviate, especially in regions behind strong expansion shocks. With this in mind, Schwartzentruber proposed the following parameter

$$Kn_{ROT-NEQ} = \frac{T_{TRA} - T_{ROT}}{T_{ROT}} \tag{2.41}$$

and found that a value of 0.01 ensured physical accuracy for flows over blunt bodies, at the cost of a larger DSMC region and increased computation time [6].



(a) Flux-based coupling.

(b) State-based coupling.

Figure 2.6: Schematic of typical HPC coupling methods [6]

Once the simulation domain has been delegated to each solver, it then becomes important to pass flow information between the CFD and DSMC domains. How this is achieved can typically be split into two categories: by maintaining consistent fluxes or by maintaining consistent state properties in reservoir cells. A flux coupling method,

depicted in figure 2.6(a), requires the calculation of the flux of conserved quantities across the hybrid interface location for both the particle, F_p, and continuum, F_c, cells. While the particle fluxes can be found by tracking the particles that cross the interface, the continuum flux must be extrapolated using cell averaged values and their gradients. Often the two fluxes have slight errors when compared, and are modified to ensure mass, momentum and energy conservation. Each flux can then be applied as a boundary condition to the apposing solver [6]. State based coupling, shown in figure 2.6(b), involves the use of ghost cells to provide a smoother transition from one regime to the other. Particle information is averaged over cells along the interface to provide the necessary macroscopic quantities for the continuum ghost cells, while the cell averaged continuum values are used to estimate the probability density functions in the DSMC ghost cells. Through this process each domain provides the Dirichlet boundary conditions to the apposing domain [6]. In addition to ghost cells, some state based methods will have the particle domain overlap with the continuum domain and both solvers will calculate a solution. By combining the results, it is possible to correct inaccuracies from an initial solution or to redetermine the location of the hybrid interface [6]. It is important to note that when particle data is passed into the continuum domain, it is often highly erratic compared to the CFD data and has a chance of creating numerical instabilities. This can be mitigated by reducing the statistical scatter of averaged DSMC data. Several techniques exist to reduce statistical scatter such as Boyd and Suns' novel subrelaxation scheme given by

$$\langle Q \rangle_j = (1 - \Phi)\langle Q \rangle_{j-1} + \Phi Q_j \tag{2.42}$$

where j is the current iteration. This scheme was later used by Schwatzentruber and Boyd in the development of a modular particle continuum method. They found that

with a typical subrelaxation parameter value of $\Phi = 0.001$, the method successfully reproduced multiple full DSMC simulations [6, 24].

2.4 Review of Literature

> Currently there is no comprehensive code to simulate the formation of the shock wave in the rarefied flow during the onset of strong ablation that would be analogous to the CFD and DSMC numerical packages that deal with hyper-sonic re-entry vehicles gas dynamics both in rarefied and continuum flow regimes.

The above quote from *"Physics of meteor generated shock waves in the Earth's atmosphere – A review"* by Elizabeth Silber et al. characterizes the current state of HPC codes. As of 2020, no commercial or open source software packages exists that offers HPC capabilities. Institutions with a need for HPC codes typically turn to internal resources and develop in house codes. This has led to a disparity in code maturity and a large variety of techniques within the field.

2.4.1 Expanding HPC Tools

HPC codes have been the subject of research for many institutions attempting to improve the accuracy of transitional flow simulations. As a result of HPC codes being complex from an information handling standpoint, significant efforts have been made to improve their usability and overall efficiency as CFD tools. At the University of Minnesota, Thomas Schwartzentruber made significant improvements to the universities HPC code, Molecular Gas Simulator (MGDS), while working under contract for the Kirtland Air Force Research Laboratory [23]. Schwartzentruber had three pri-

mary objectives during this development effort: to get the universities DSMC code to more efficiently simulate hypersonic flow around complex 3D geometries, to allow for uncoupled DSMC simulations to be run within the HPC framework, and to develop the DSMC chemistry models to be more compatible with CFD models.

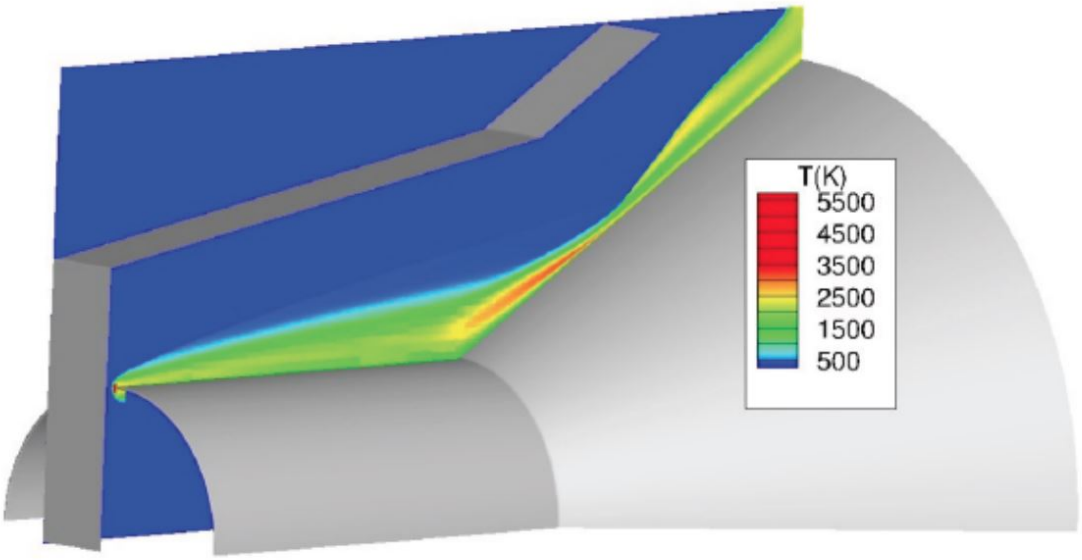


Figure 2.7: Hollow Cylinder Flare problem employing arbitrary inflow/outflow planes [23]

To accommodate complex geometries Schwartzentruber applied several techniques. He separated an adaptive mesh refinement algorithm (AMR) from DSMC source code so that it could be run in parallel and selectively during post processing. Before this modification, in order to perform any sort of post processing the entire grid and solution would need to be loaded into memory. Allowing the user to run AMR on a selected region of interest circumvented this process. In the same spirit of making the DSMC code more efficient at analyzing user selected portions of a geometry, Schwartzentruber also implemented arbitrary boundary conditions. By setting inlet and outlet boundary conditions anywhere within the simulation domain, a user could introduce particles from a known CFD solution via an inlet BC and remove particles when they were redundant via an outlet BC [23]. This allowed the user to decouple

the DSMC and CFD portions of the universities code, allowing for more flexibility and greater memory efficiency.

The final addition Schwartzentruber made to MGDS was developing a dissociation model for reacting flows that was consistent at both the molecular and continuum level. Standard chemistry models had not been developed with HPC codes in mind. As a result, the only chemistry models available for CFD and DSMC were empirical models that had been developed separately and not been successfully applied to a hybrid simulation. The new model, called direct molecular simulation (DMS), simulated rovibrational excitation and coupled dissociation of shock heated gas. For DMS, coupled dissociation meant that it took in multiple energy modes when calculating the probability of dissociation upon a confirmed DSMC particle collision, in this case vibrational and translational. The new model for calculating rovibrational excitation included deviations from a Boltzmann distribution due to overpopulation of high vibrational energy states from rapid excitation, and the depletion of these states from dissociation. This non-Bolztmann distribution function was unique in that it could be integrated with respect to vibrational energy levels to get an overall probability of dissociation for a gas, which in turn could be used in CFD simulations [23, 27]. The DSMC dissociation model, non-Boltzmann distribution function and bulk probability of dissociation are given respectively by the following equations.

$$P(d|v) = C_1 exp\left[-\alpha\frac{\epsilon_d}{\langle\epsilon_t\rangle}\right] exp\left[\frac{\epsilon_v}{\langle\epsilon_t\rangle}\right] \frac{\Gamma[\zeta_{tr}/2, (\epsilon_{lj}-\epsilon_{vo})/\langle\epsilon_t\rangle]}{\Gamma[\zeta_{tr}/2]} \tag{2.43}$$

$$f(\epsilon_v) = \frac{C_2}{\Gamma(\zeta_v/2)}\frac{1}{\langle\epsilon_v\rangle}\left(\frac{\epsilon_v}{\langle\epsilon_v\rangle}\right)^{\frac{\zeta_v}{2-1}} exp\left[-\frac{\epsilon_v}{\langle\epsilon_v\rangle} + \gamma_1\left(\frac{\epsilon_t}{\langle\epsilon_v\rangle} - \frac{\epsilon_v}{\langle\epsilon_t\rangle}\right)^{\psi}\left(\frac{\epsilon_v}{k_B\theta_v}\right)\right] \tag{2.44}$$

$$\langle P_d\rangle = \int_0^\infty P(d|\epsilon_v) f(\epsilon_v) d\epsilon_v \tag{2.45}$$

In equation 2.43 ϵ_v is the vibrational energy state, $\langle \epsilon_t \rangle$ is the average transnational energy of the gas, and ϵ_d is the dissociation energy. The equation itself gives the probability of dissociation for a molecule. In equation 2.44, λ_1 accounts for overpopulation during excitation while λ_2 accounts for depletion during dissociation [23].

2.4.2 Pulsed Laser Ablation

HPC codes are applicable to any physical situations with rapid changes in particle density. This means that in addition to aerospace, there are a wide variety of applications in industry as numerical models for technologies such as pulsed laser ablation become more mature. In industry pulsed laser ablation is used to create thin film depositions on substrates such as silicon wafers. This is achieved by placing the substrate opposite the ablation target. Then, while both are under vacuum, the ablation target is vaporized with a laser, allowing the resulting plasma to expand evenly within the vacuum chamber and deposit particles onto the substrate surface [7]. In 2018 researchers at the CFD Research Corporation in Huntsville AL examined this process with a custom HPC code called Unified Flow Solver (UFS), specifically the jets of plasma that result from material vaporizing off an ablation source. The researchers performed simulations of two physical scenarios: the expansion of evaporated material into a vacuum and background gas, and the expansion of plasma into a vacuum [2].

The group performed several iterations of the expansion of evaporated copper with different solvers. The first was the 1D expansion of Copper (Cu) into Argon (Ar) with a Naiver-Stokes solver. The background Ar was at 200 mTorr and 300 K while the high density Cu region had a number density of $10^{19} cm^{-3}$ giving a resulting pressure ratio of 35000. The solver produced expected results with the Cu plume front propagating at 3000 m/s and dropping in density by 3 orders of magnitude. The

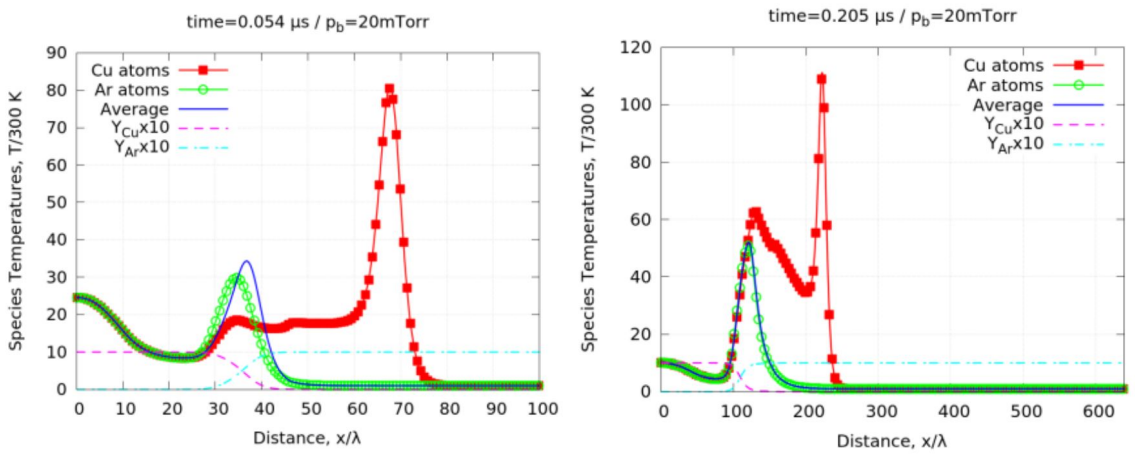


Figure 2.8: Species macroparameters obtained by UFS-Boltzmann solver for 20 mTorr. Shown are species mean temperatures together with mixture average temperatures and species mass fractions at 2 time instances of 0.054 and 0.2 μs. Spatial scale is normalized to λ=100μm [2].

pressure of the background gas was then varied to 20 mTorr and 2 Torr and showed an expected decrease in the plume propagation velocity and temperature with increasing gas pressure. Finally, the mass density and energy initial conditions were changed from a free expansion scenario to one more closely resembling pulsed laser ablation. This resulted in an expected plume velocity of 5000 m/s [2]. The same simulation was ran with a Boltzmann solver and matched previous results. In the Boltzmann solution two distinct velocity distribution functions (VDF) formed, a Maxwellian distribution in the highly collisional Cu explosion core and non-equilibrium VDF in front of the plume/core boundary. The non-equilibrium VDF was characterized by two velocity peaks, one low energy Maxwellian distribution and a high energy peak caused by high energy particles escaping the dense core region. As these high energy particles were heavy Cu particles, any collision they underwent with the background Ar gas resulted in a very small momentum exchange, meaning that once the high energy particles escaped the dense core region, they became collisionless. This results in a low density and high temperature region at the plume front, one that is well captured with a Boltzmann solver as apposed to a statistical DSMC method [2].

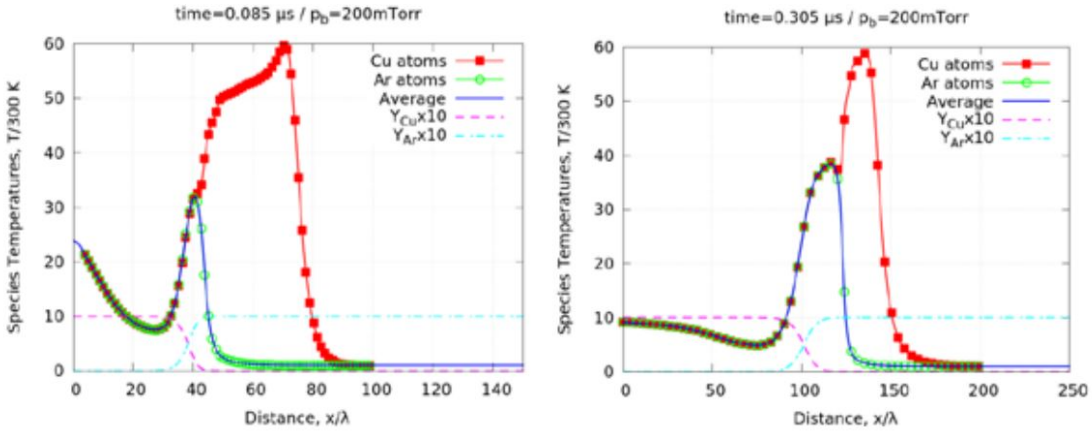


Figure 2.9: Species macroparameters obtained by UFS-Hybrid solver for 200 mTorr. Shown are species mean temperatures together with mixture average temperatures and species mass fractions at 2 time instances of 0.09 and 0.3 μs. Spatial scale is normalized to λ=10μm [2].

After running the Cu plume expansion into Ar gas on both an NS and Boltzmann solver, the researchers ran the simulation on UFS. The UFS code uses a Knudsen number based break down criterion given by

$$S_{NS} = K_n \sqrt{\left(\frac{\Delta p}{p}\right)^2 + \frac{1}{U^2}\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 + \left(\frac{\partial w}{\partial z}\right)^2\right]} \tag{2.46}$$

where S is a set threshold value [16]. UFS also has a unique shared flux coupling scheme between Boltzmann and NS solvers. At each time step a velocity grid is introduced to a boundary cell in the continuum domain that matches the neighboring Bolztmann cell. On the velocity grid the following Maxwellian distribution functions are constructed.

$$f_0 = f_M^1[1 - \tau(a^1\xi_n + A^1)] \tag{2.47}$$

ξ_n is the normal velocity to each cell face,f_0 denotes the velocity distribution function at $t = 0$, f_M denotes the Maxwellian distribution around the cell faces [16]. The term a^1 represents the discrete velocity space polynomial function whose coefficients are calculated using the gradients of the macroparameters in the continuum and the

neighboring kinetic cells, while A^1 represents another discrete polynomial function whose coefficients are calculated from the relationship of conservation of the moments on the velocity grid given by

$$\int f_M^1 (a^1 \xi_n + A^1) \psi_\alpha d\xi = 0 \tag{2.48}$$

The researchers claimed the results of the simulation on the hybrid solver had good agreement with the Boltzmann solver and had a gain factor of 10 in terms of CPU time. They also anticipated even greater gains in computation time were possible if their breakdown criterion were optimized for their scenario [2].

2.4.3 Alternative Hybrid Codes

As mentioned in section 2.3.1, various hybrid codes exist beyond NS and DSMC hybrid solvers. Researchers at the university of Michigan have implemented a DSMC to solve rarefied flow regimes and the low diffusion (LD) particle method for continuum regions. This type of solver belongs to a class of hybrid codes 'all particle' codes. Their primary advantage is that virtual DSMC superparticles can be used throughout the entire domain, allowing for relatively simple code development and information transfer between the two solvers. However, 'all particle' methods are prone to large errors in continuum regions due to numerical diffusion [15]. To account for this, the LD method significantly reduces random particle motion by limiting it to the macroscopic motion of Lagrangian cells, resulting in particle trajectories that closely resemble gas streamlines. The LD method begins after particles are moved due to collisionless motion. Here, cell averaged mass density, bulk velocity, and characteristic thermal speed are calculated for each cell in the LD regime. Using these macroscopic properties, Lagrangian face velocities are calculated and Lagrangian cell faces are superimposed

over the fixed Eulerian cell faces. Then momentum and energy exchange is calculated across these Lagrangian faces and the resulting bulk velocity and temperature values are assigned to all particles in a cell. While the particles remain stationary relative to their assigned Lagrangian cell, it is the Lagrangian cell vertex that moves based on bulk velocity values. By taking this macroscopic approach, the LD method maintains minimal numerical diffusion and statistical scattering when compared to alternative methods [15].

The LD-DSMC solver used a maximum gradient length Knudsen number given by

$$Kn_{GLL,max} = max\left(\frac{\lambda}{\rho}|\Delta\rho|, \frac{\lambda}{T}|\Delta T|, \frac{\lambda}{a}|\Delta u|\right) \tag{2.49}$$

to determine continuum breakdown, which was assumed to occur at a $Kn_{GLL,max}$ of 0.05. The solver was applied to two simulations: Mach 10 nitrogen flow over a sphere and Mach 40 carbon dioxide flow over the Mars pathfinder re-entry capsule. For both scenarios, the LD-DSMC method was compared to full DSMC and CFD simulations. The method largely matched DSMC results, with macroscopic flow characteristics never exceeding 5% error, however both cases saw an increase in computational efficiency over DSMC. For the flow over a sphere, there was a 20% improvement to computation time, from 1880 total CPU hours to 1520 hours and for the re-entry capsule, there was a 50% improvement, from 2283 hours to 1128 hours. However despite the gain in computation time, the LD-DSMC method also proved to be very sensitive to what relaxation coefficient and maximum allowable CFL number were used. For DSMC regions, large statistical scattering can occur when transferring instantaneous cell averaged data. A method to alleviate this is to apply a sub-relaxation procedure given by

$$Q_{cell} = w * Q^{n}_{cell} + (1 - w)Q^{n-1}_{cell} \tag{2.50}$$

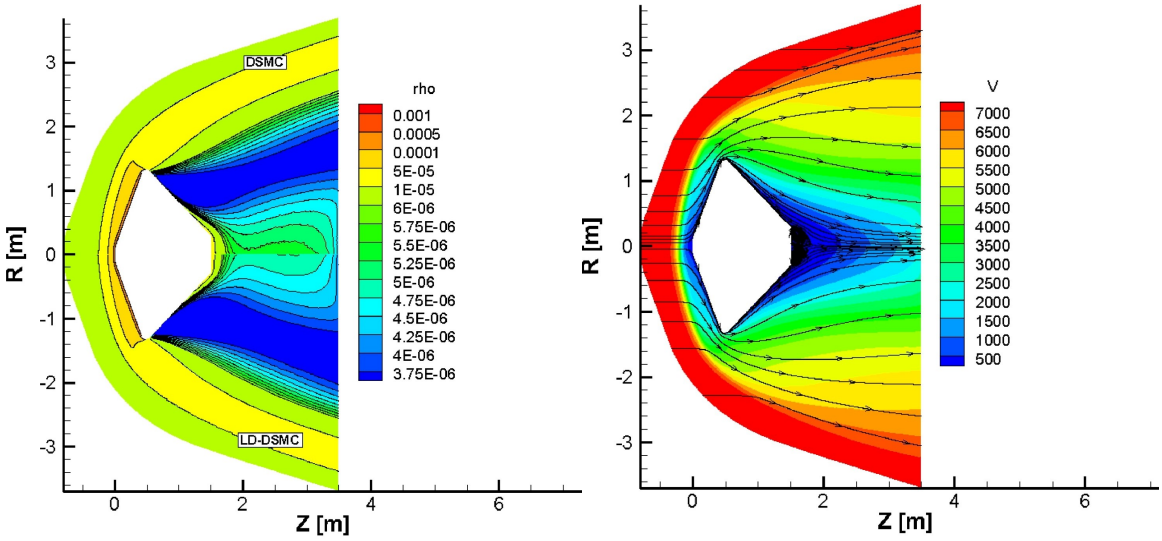
where Q_{cell} is some cell averaged quantity of interest. For the LD-DSMC method, numerical instabilities develop when w is greater than 0.01 and if flow conditions are extreme, this value may need to be reduced. The CFL criterion is used by this method, in part, to determine the number of sub-cycles performed in the LD domain and avoid instabilities due to a large time step size. A greater number of sub-cycles means the LD procedures will be repeated more times before proceeding to the next time step. The number of sub-cycles is found via

$$N_{sub} = 1 + \left| \frac{CFL_{LD,max}}{CFL_{allowable,max}} \right| \tag{2.51}$$

where

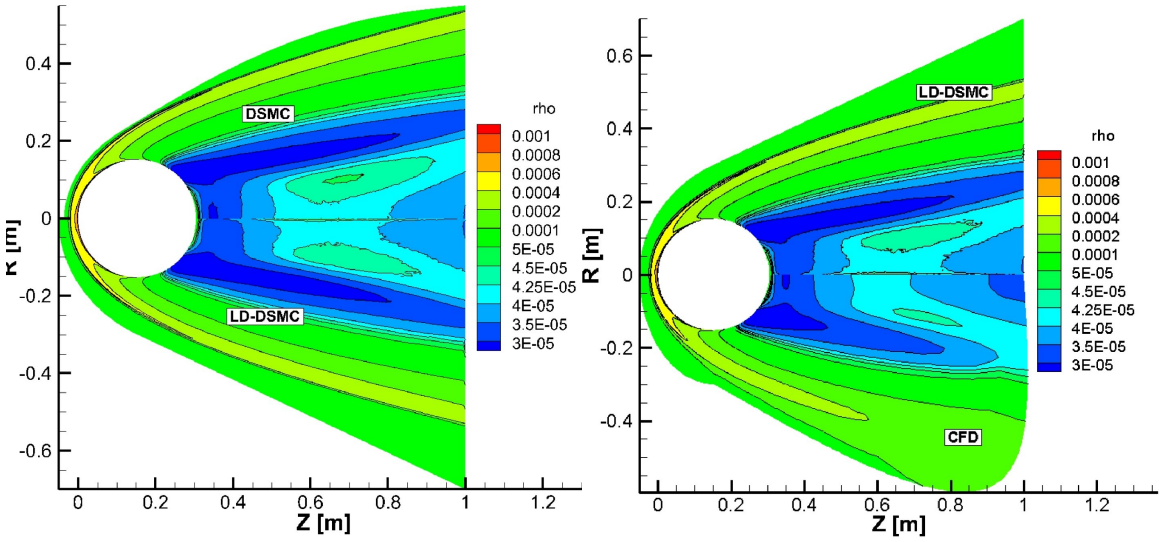
$$CFL_{LD,max} = \frac{\Delta t}{\Delta x} \left[\mathbf{v} + \theta + \frac{5}{\Delta x} \frac{\mu}{\theta} \right]. \tag{2.52}$$

The $CFL_{allowable,max}$ was maintained at 0.8 for these simulations, but the authors advised decreasing the value whenever the method became unstable.



(a) (b)

Figure 2.10: Mass (a) and velocity (b) contours for the Pathfinder re-entry capsule. Lower radial half is LD-DSMC hybrid method while upper half is full DSMC [15].



(a) (b)

Figure 2.11: Comparison of mass density contours between the LD-DSMC hybrid solver and full DSMC (a) and CFD (b) solutions for N2 flow over a sphere [15].

Chapter 3

METHODOLOGY

3.1 hybridFoam Development

3.1.1 Development Milestones

The development of hybridFoam was kept relatively simple when compared to a fully functional HPC code due to the time restraints of the Calpoly masters program. With this in mind the following limitations were made to the project scope: the breakdown location would be static and fixed to geometry rather than knudsen number and only 1D capabilities would be required. Additionally, the openscource code base OpenFOAM would be used to provide the basis for the code, utilizing its built in Euler/Navier-Stolkes solver and DSMC solver. With these scope restrictions, a number of critical path tasks were identified. These are characterized by the following project milestones.

1. **Modify dsmcFoam**: The built in DSMC solver for OpenFOAM (dsmcFoam) models multi-species simulations but only for uniform initial temperatures and number densities throughout the domain. In order to run the shocktube test case, it would need to be modified.

2. **Validate dsmcFoamMod**: A standard shocktube simulation would need to be used to validate the performance of the modified dsmcFoam code.

3. **Create Meshing Hybrid Framework**: The final hybrid solver would need two separate meshes; one mesh for the CFD solver and another for the DSMC solver. Then both solvers would need to be run simultaneously on the meshes.

4. **Implement Boundary Conditions**: hybridFoam would require information to be passed between the solvers in a way that accounted for the inherent differences between CFD and DSMC data. Custom boundary conditions would need to be developed.

3.1.2 hybridFoam Code Structure

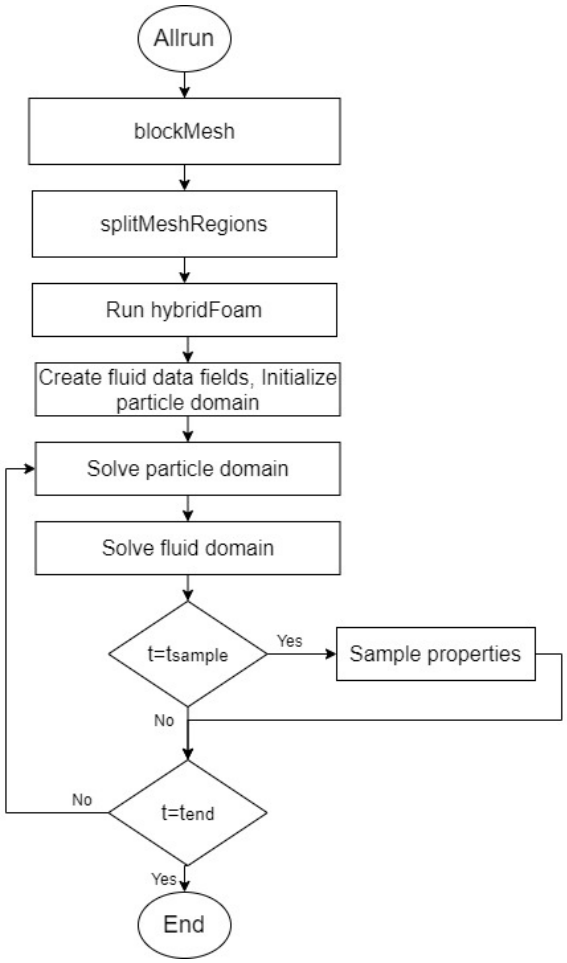


Figure 3.1: High level flow chart of the hybridFoam command sequence

hybridFoam was developed in OpenFOAM using windows subsystem for linux (WSL). It utilizes a CFD solver that uses the Kirganov and Tadmore scheme detailed in appendix A to solve a user defined fluid domain and a DSMC solver, detailed in section 2.2.2, to solve a similarly defined particle domain. Both solvers are modified versions of OpenFOAM solvers, rhoCentralFoam for the fluid regime and dsmcFoam for the particle. The hybrid solver operates much like standard custom OpenFOAM solvers in that there are three primary input directories: zero, constant, and system. The zero directory holds initial and boundary conditions for the simulation. The constant directory contains essential parameters that do not change during the runtime. This includes transport and thermodynamic properties for the CFD solver and the virtual-real particle ratio, collision model label, inflow model label, and species information for the DSMC solver. The system directory holds mesh and time step information as well as pointers to additional libraries and functions not encompassed in the source code of the solver. Each directory also has subfolders for fluid and particle specific inputs. These input files are accessed by the solver after it is called by an Allrun bash script. The high level command sequence called by the Allrun script can be seen in figure 3.1. The first command in the sequence calls the blockMesh utility, an OpenFOAM program used for dividing a defined domain into distinct hexahedral blocks by reading a blockMesh "dictionary" or input file [13]. The blockMesh dictionary contains the mesh dimensions and orientation in addition to boundary condition information. The blockMesh command also reads in the initial data from the zero directory for both the fluid and particle domains. For the fluid domain, pressure, temperature and the velocity data fields are created and uniformly populated with values from the "internal field" label in the input files. For the particle region, although fields are created from the 0 dictionary files, the majority of fields created do not provide actual initial conditions. Only the boundary velocity and boundary temperature fields that are set in the 0 directory affect the simulation by providing

initial conditions for standard particle inflow boundary conditions. The actual initialization of the DSMC domain is handled by the modified dsmcInitialise function in the dsmcCloudMod class, dsmcInitialiseMod2. This function reads in a dedicated dsmcInitialise dictionary, that sets the total temperature and velocity for each molecule in the domain, distributing them according to an input number density. After the blockMesh command is called, the splitMeshRegions utility divides the domain into different cellzones, creating a new database branch and mesh for each zone. This allows the zones to operate largely independently from each other, while still allowing some access between the meshes through the database hierarchy. A simple validation of the splitMeshRegions utility was performed by running a shocktube scenario in the DSMC domain and constant homogeneous conditions in the CFD domain with a zero gradient BC on all CFD faces and a symmetry BC on all DSMC faces. The results in appendix B.1 showed nominal behavior and each domain operated successfully during the the same runtime.

After the meshes are totally defined, the hybrid solver is called and the CFD and DSMC solve for flow behavior on their respective meshes. The solvers step through their respective algorithms in figure 3.2 while flow information at the hybrid interface is constantly exchanged via the BC's in table 3.1.

Table 3.1: Boundary conditions for HPC interface

hybridFoam Interface Boundary Conditions			
Fluid Boundary		Particle Boundary	
Variable	**BC**	**Variable**	**BC**
p	Linear interpolation	ρ	Matching via ideal gas law
T	Linear interpolation	T	Matching
U	Linear interpolation	U	Matching

The boundary conditions used at the hybrid interface have been adapted from Gott [12]. Gott assumed one directional flow for their boundary conditions when simulating material ejection off of a laser ablation target i.e. only fluid to particle flow. The assumption allows for the reduction of terms passed between the two domains, meaning that only pressure would need to be passed into the fluid domain while pressure and temperature could maintain a zero-gradient boundary condition. The validity of that assumption breaks down when Brownian motion is significant and there is not a large gradient across the hybrid interface. In the case where the hybrid interface has homogeneous conditions on either side, such as the early time steps of a shock tube simulation, the one directional flow assumption produces non conserved values, which can be seen in appendix D. For hybridFoam's temperature, velocity and pressure terms in the fluid domain, values at the hybrid interface are found through linear interpolation of the boundary cells in both domains.

$$Q^{n}_{CFD,face} = (1-\omega)Q^{n-1}_{CFD,face} + \omega\left(\frac{1}{2}Q^{n}_{CFD,center} + \frac{1}{2}Q^{n}_{DSMC,center}\right) \tag{3.1}$$

Combined with a relaxation coefficient, w, the interpolation formula in equation 3.1 is meant to ease the transition from erratic particle data to smooth fluid data in the absence of more sophisticated techniques to limit statistical scatter. It is also important to note that the terms in the DSMC boundary cells in equation 3.1 are cell averaged values calculated at each time step. In the DSMC domain, density, temperature and velocity at the boundary are set to match the fluid values on the opposite side of the shared boundary face. Once these variables are set, new particles are generated and injected into the DSMC domain via a modified inflow boundary model called HybridInflow. The model uses equation 4.22 from Bird [4] to determine the number of incoming molecules, and uses the boundary temperature to assign the bulk of injected molecules a most probable speed. The boundary velocity, derived from the CFD face

velocity, is used to determine the stream velocity of incoming particles. After the number of incoming molecules are determined, individual velocities are assigned to form a distribution equivalent to the particles being diffusely reflected off of a surface via equation 12.5 from Bird [4]. The boundary conditions are sourced differently

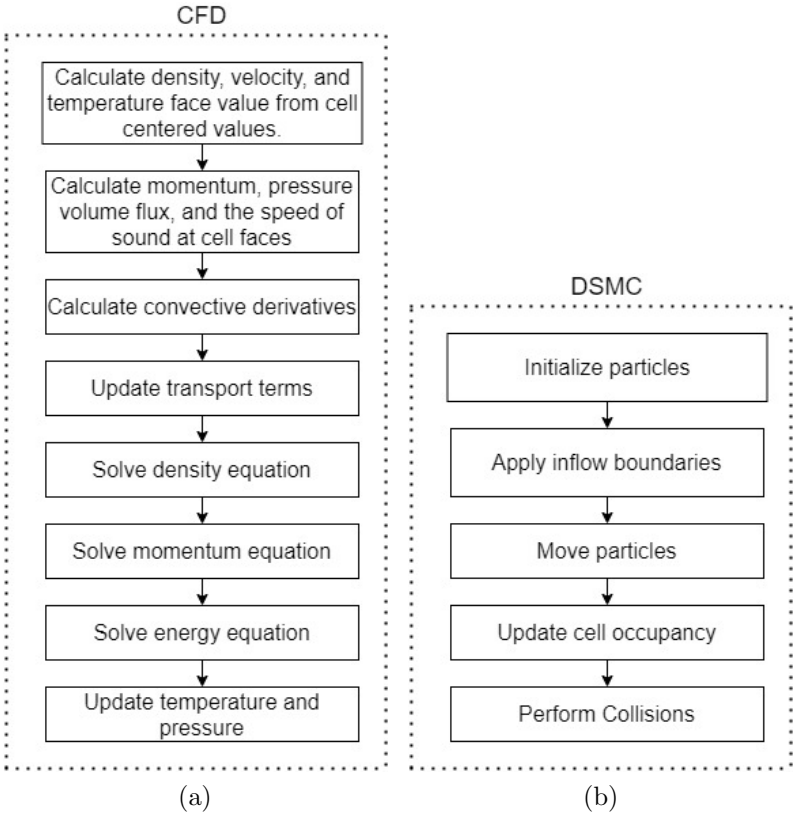


Figure 3.2: Fluid (a) and particle (b) solver algorithms

for each solver. The inflow model for the DSMC solver is built into the library for dsmcFoamMod, while the source code for the fluid BC's are produced via a standard openfoam utility called coded fixed value. This utility reads in the coded fixed value BC tag in the 0 dictionary for the fluid solver, which in turn points to another dictionary that defines the primary script used for the boundary condition. This process condenses the number of files required to define the custom boundary conditions and is particularly easy to approach for someone who is used to coding with scripts, as is common for Cal Poly which almost exclusively uses Matlab in its coursework. This was the primary motivation to keep the custom boundary conditions in this form, as

it will hopefully make experimentation with alternative HPC BC's easier in future development. The code for the coded fixed value BC's are compiled on every run, before hybridFoam is called. As hybridFoam solves the simulation, at particular write-times dictated in the control dictionary, folders are created in the simulation directory with output data. These data files contain the internal and boundary field information for each data field created throughout the simulation. While OpenFOAM comes with an installation of ParaView for data analysis, the majority of post-processing for this thesis was done in Matlab.

3.1.3 Modifications to dsmcFoam and Validation

In milestone 1, modifications were made to dsmcFoam to form dsmcFoamMod and eventually dsmcFoamMod2 which is a class utilised by hybridFoam rather than a separate solver. As stated, the primary goal of this effort was to run a shocktube test case that could be validated in milestone 2. These efforts reflect the development needed for the particle solver, while the fluid solver only required superficial modifications to be integrated with the hybridFoam solver and utilise custom boundary conditions. The original dsmcFoam solver could only solve homogeneous multispecies problems, however these limitations were primarily due to the initialization function. To accommodate the shocktube simulation, the initialization function was modified to read in one number density and temperature for each region as well as the cell locations dividing each region. After these numbers are read in via the dsmcInitialise dictionary, the particle mesh is populated with a number of molecules according to the input number density, with each being assigned a velocity and internal energy according to the input temperature. In order to simplify the initialization process, multispecies capabilities were removed.

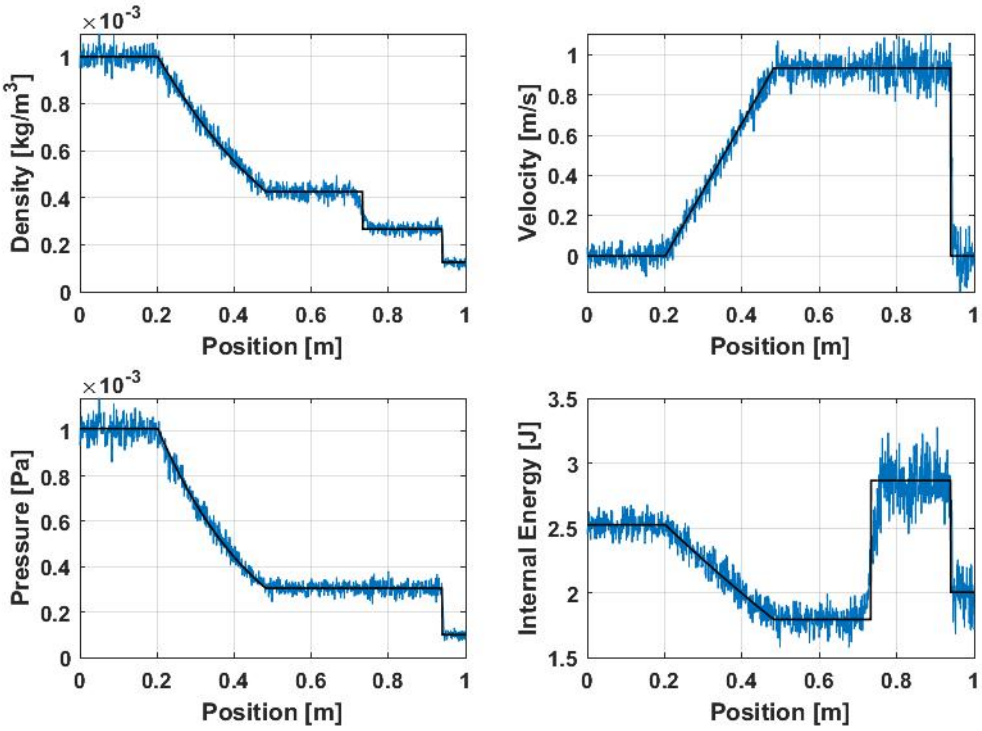


Figure 3.3: Validation of dsmcFoamMod Performance. Density plot at $\Delta t = 0.25s$

A modified Sod shock simulation was used to test the performance of the modified DSMC solver. The standard Sod Shock simulation has the following normalized initial conditions over a 1m domain; $\rho_L = 1$, $u_L = 0$, $p_L = 1$, $\rho_R = 0.125$, $u_R = 0$, $p_R = 0.1$. The domain was kept to $1m$ with 1000 cells along the x axis and $480,000$ DSMC particles were used resulting in Knudsen number values of roughly 0.34 and 2.7 for the high and low density areas respectively. The results were compared to an analytical shocktube solver in Matlab adapted from FORTRAN code taken from E.F. Toro [30] in figure 3.3. The DSMC results closely follow the analytical solution, however there is significant smearing at the contact discontinuity located at $x = 0.73m$. This matches previous DSMC results where the shock and contact discontinuities were smeared over several mean free paths [33, 20, 22]. As the smearing of the contact discontinuity occurs early in the simulation domain and does not change significantly, it is most likely the result of molecular diffusion, i.e. the increase of particle mobility

due to low densities. Although it would mean more computation time, increasing the initial number density in the particle region would provide a sharper resolution of the contact discontinuity. The smearing of the shock is due to viscous effects in the flow as the shock thickness is a function of the viscosity temperature index and the shock mach number when considering viscous floe [4]. The random statistical scatter seen in figure 3.3 is inherent to the DSMC method and can be reduced by increasing the number of simulation particles and therefore, increasing the computation time.

3.2 Test Cases

Two types of test cases were used to test the performance of hybridFoam: validation and stability tests. Of the former, a modified sod shock with the same initial conditions as the DSMC validation case was performed as well as a custom strong shock test. The purpose of the strong shock test case was to use initial conditions with Knudsen numbers in the continuum and rarefied regimes across the rarefaction wave. This is meant to showcase the primary advantage of HPC codes, to cover large Knudsen number regimes while maintaining physical accuracy and computational efficiency. For code stability, three tests were performed: a mirrored shocktube test, a multi-shock test, and a low density test. The purpose of the mirrored shocktube test case, which is a modified sod shock case with the low and high density regimes switched, is to test the flexibility of the solver and detect hard coded values. The multi shock and low density test are meant to test the solvers ability to deal with extreme situations. The multi shock test case is Test 5 in section 4.3.3 of Torro [30]. The test shows the collision of two strong shocks and consists of a pair of shocks traveling to the right with a contact discontinuity between them. This test case is meant to investigate possible spurious energy or velocity fluctuations at the contact discontinuity as it passes through the continuum and rarefied regimes as well as the

ability of the hybrid interface boundary conditions to transfer discontinuities. The low density test case is test 2 in section 4.3.3 of Torro [30]. The test is commonly referred to the 123 problem and consists of two strong rarefaction waves moving apart from one another, and a stationary contact discontinuity. The test was meant to examine hybridFoams ability to render very low density simulations. The results of each test were compared to an inviscid analytical shocktube solver in Matlab, adapted from FORTRAN code taken from E.F. Toro [30]. It is important to note that the analytical inviscid solution will differ from the DSMC solution, as the DSMC solver is viscous. As will be shown in section 4.1, the shock thickness in the DSMC solution will be spread over several mean free paths. This is expected since shock thickness is a function of the viscosity temperature index and the shock mach number [4]. Additionally each simulation held a consistent 1000 cells along the x-domain and used a relaxation coefficient of $\omega = 0.4$ at the linear interpolation boundary condition. The tables showing the initial conditions for each test are divided into left and right properties. These refer to the left and right hand side of the initial discontinuity, which in each test has been placed in the middle of the domain at $x = 0.5m$ or, in the case of the strong shock test, $x = 5m$. Each test also assumed an ideal gas, using $N2$ as the simulation species, and the fluid solver in each case assumed inviscid flow, solving for the Euler equations rather than the full NS equations. The slope limiter used for the fluid solution was vanAlbada in every case. It is also important to note that although its individual solvers have the capability, hybridFoam cannot currently run in parallel using the standard OpenFOAM utilities, so each test was run on a single processor. In addition to these stability and validation tests, a single shocktube test case was run using three different methods: fluid, DSMC, and HPC. This was done to confirm the computational gains of the HPC method and demonstrate the difference in computation time between particle and fluid methods.

Table 3.2: Summary of test cases for hybridFoam validation and stability testing

Test Simulation	**Rational**
Sod Shock	Initial verification and performance validation of solver and boundary conditions
Strong Shock	Demonstration of HPC advantage: the ability to handle a wide range of Knudsen numbers
Multi-shock	Examine hybridFoam's ability to transfer shocks and contact discontinuities across the hybrid interface as well as to identify spurious energy or velocity spikes at the contact discontinuity in either domain.
Low Density	Examine hybridFoam's ability to render very low density simulations
Mirrored Sod Shock	Check code flexibility

Chapter 4

RESULTS AND ANALYSIS

4.1 hybridFoam Performance Validation

4.1.1 Sod Shock

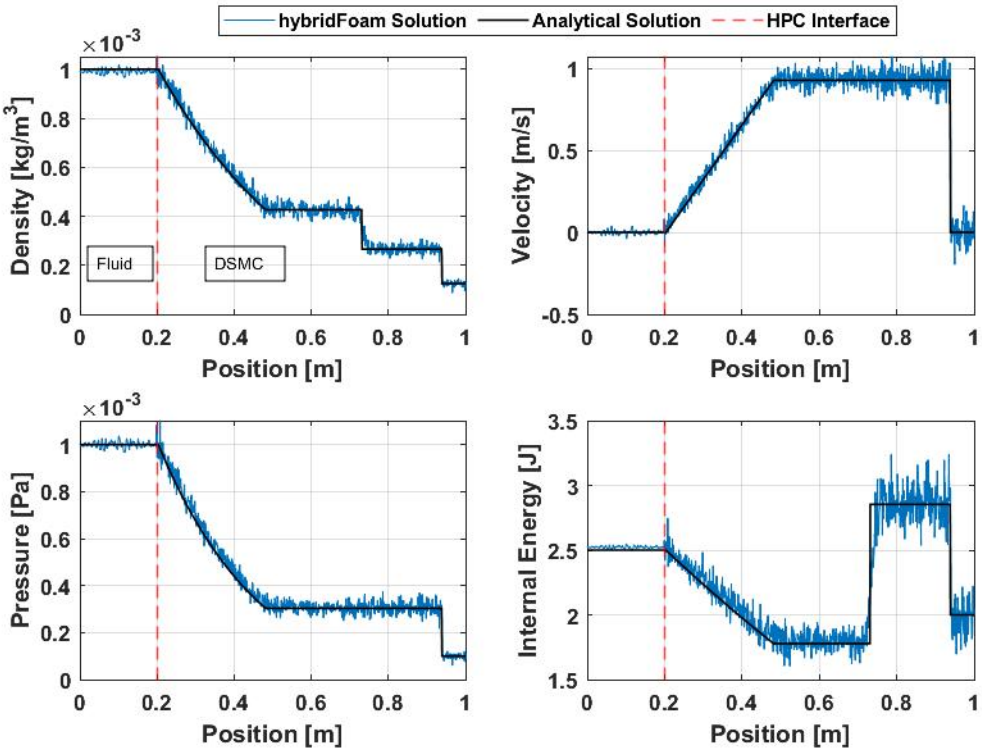


Figure 4.1: Shocktube validation test for hybridFoam, $\Delta t = 0.25s$

Table 4.1: Shock Tube Initial Conditions

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
1×10^{-3}	0	1×10^{-3}	1.25×10^{-4}	0	1×10^{-4}

The first test performed to validate hybridFoams performance was a modified Sod Shock case. The test ran successfully at a total cpu time of 11.5 hours and 400 time

steps. The particle domain ($0.2m$ to $1m$) used $389,000$ DSMC particles and the initial Knudsen numbers were 2.7 and 0.34 for the right and left regions respectively. This meant that the CFD solver was just on the edge of continuum breakdown, in the transitional regime. The particle and fluid domains were divided such that the rarefaction wave would not cross the hybrid interface, and conditions immediately on either side of the interface would remain homogeneous throughout the test. As can be seen in figure 4.1 there is good agreement between the inviscid analytical solution and hybridFoam. There is a smearing of the contact discontinuity as was seen in the DSMC validation test and the DSMC portion of the solution shows significant variation in every conserved value. The variation increases significantly at large velocity and internal energy values after the contact discontinuity. As can be seen in figure 4.2, the fluid solution, rather than being constant, exhibits similar variation in its values, with smoother but still erratic jumps. This is due to the data transferred via the HPC interface and can be seen as a wave traveling through the fluid solution at earlier timesteps. The presence of the fluctuations shows that the linear interpolation boundary condition combined with a relaxation coefficient is not entirely sufficient with this number of DSMC particles to maintain the smoothness of the CFD solution. These fluctuations could be avoided by increasing the number of virtual particles or applying another statistical scatter limiting method such as using a chapman-enskogg distribution to provide an intermediate "transition" distribution between the domains or by applying a more robust relaxation scheme. The average value of the density in the fluid region also slightly deviates below the ideal value by an average of 0.3%. This deviation grows slightly in the area closer to the HPC interface where variations due to particle input data are from more recent timesteps. Although this variation is not statistically significant and well within the range of standard deviation of the particle data, it could indicate slight numerical inaccuracies caused by the particle injection or interpolation boundary conditions.

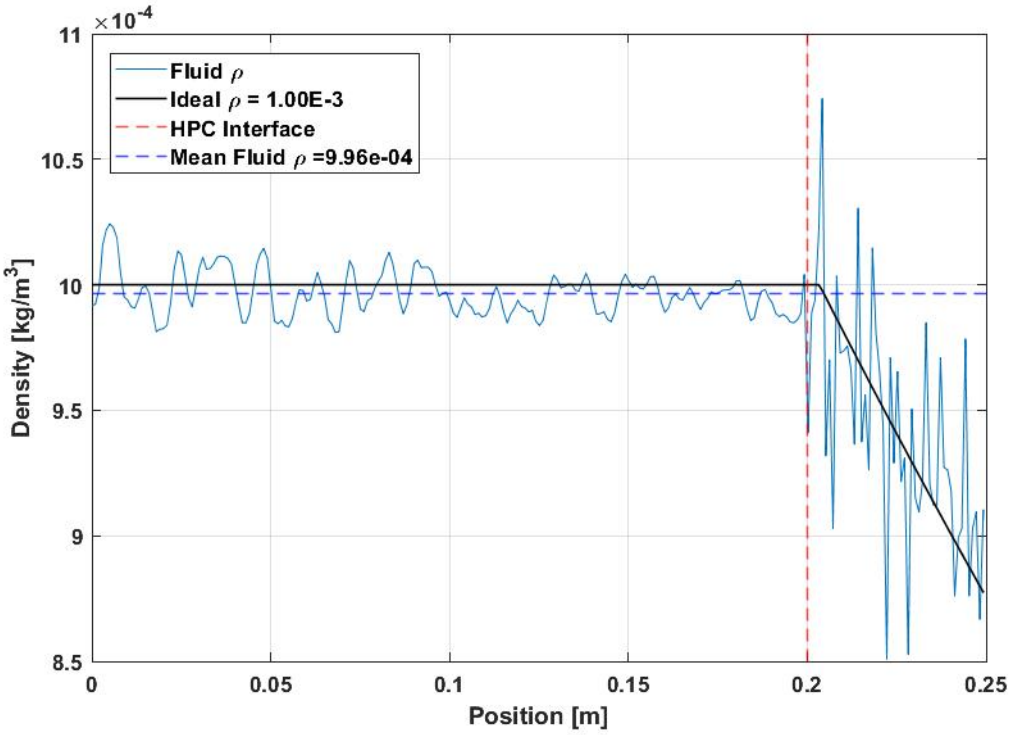


Figure 4.2: Low density sod-shock HPC interface

4.1.2 Strong Shock

The final validation test case performed was a custom strong shock test case where initial density and pressure ratios were chosen to ensure the local knudsen numbers of each regime started in either rarefied or continuum flow. The initial Knudsen numbers for the DSMC and CFD regions were $Kn = 27$ and $Kn = 0.11$ respectively. The domain was also expanded from the standard $1m$ to $10m$ in order to accommodate the enlarged rarefaction wave and, like the Sod Shock test case, the regions were chosen so the rarefaction wave did not pass over the HPC boundary. The test was run successfully with a total CPU time of 2.4 hours and used 400 time steps. The number of DSMC particles used was 490361 for a DSMC region from $3.5m$ to $10m$. Although hybridFoam shows reasonable agreement with the density and pressure solutions in figure 4.3, the contact discontinuity and shock in the internal energy and velocity

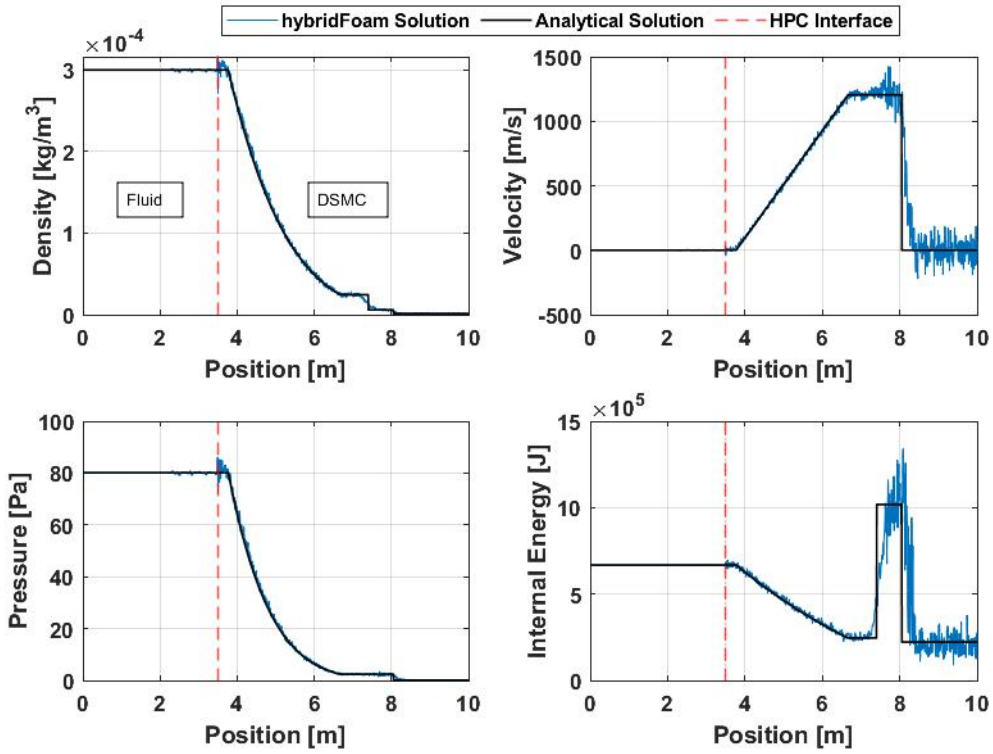


Figure 4.3: Shocktube validation test for hybridFoam, $\Delta t = 0.002s$

Table 4.2: Strong Shock Initial Conditions

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
3.00×10^{-4}	0	80.136	1.25×10^{-6}	0	1.11×10^{-1}

graphs are very under resolved and skewed. The cause becomes more evident in figure 4.4 where it can be seen that the contact discontinuity and shock are smeared over a distance of $0.8m$ and $0.5m$ respectively. The initial mean free path of the low density region is very large, $27cm$, meaning that the shock is smeared over 2 mean free paths and the contact discontinuity is smeared over 3. As the DSMC method is particle based, and relies particle interactions to transfer information throughout the medium, resolving flow structures at a resolution below the local mean free path is unfeasible for such low densities and large length scales. While the contact discontinuity is most likely smeared due to molecular diffusion, the shock is most likely smeared due to

viscous effects. Because of this, the shock front appears to lead the ideal solution as the smeared shock begins to resolve at the ideal shock front, unlike the contact discontinuity where the midpoint of the computational solution intersects the ideal. This smearing becomes magnified in the velocity and internal energy plots as their ranges do not cover the several orders of magnitude that the pressure and density plots do. In addition to differences, there is a spurious dip in density at the HPC interface on the DSMC face. This could be a random occurrence as the fluid data does not show evidence of any significant spikes from earlier time steps. However, it is more likely that this spike is the result of the particle injection code on the DSMC side of the HPC interface as this same spike occurs in the Sod Shock test case. In the Sod Shock test case the spike is of a similar magnitude as the variation in the particle data and not significant. However, when compared to the Strong Shock test, the placement and magnitude of the spike is almost identical. This dip in density could be caused by an error in the rounding scheme used in the injection code. As the number of particles injected is discrete and the flow across the interface from the fluid region is a non-discrete value, the number of injected particles must be rounded off. If the injection code consistently rounds down, this could result at a dip in density at the interface. Another possibility is that this spike is the result of a mismatch in temperature terms at the interface. As the fluid solver uses a cell averaged temperature and assumes inviscid flow, the conversion to a most probable thermal velocity for the incoming particles may not capture the coupling between temperature and viscosity in the DSMC region that is not present in the fluid region. Such a mismatch could explain why the dip in density at the particle boundary cell is preceded by a rise in density at the fluid boundary cell. Beyond these irregularities at the HPC interface, the relative smoothness of the CFD solution is improved when compared to the sod shock test in figure 4.2, although as very little real time has passed in the simulation, much of the CFD domain is unaffected by the particle data

passed through the HPC interface. The larger number of DSMC particles at the high starting number density could also mean the area next to the interface would have less statistical variations due to increased particle collisions.

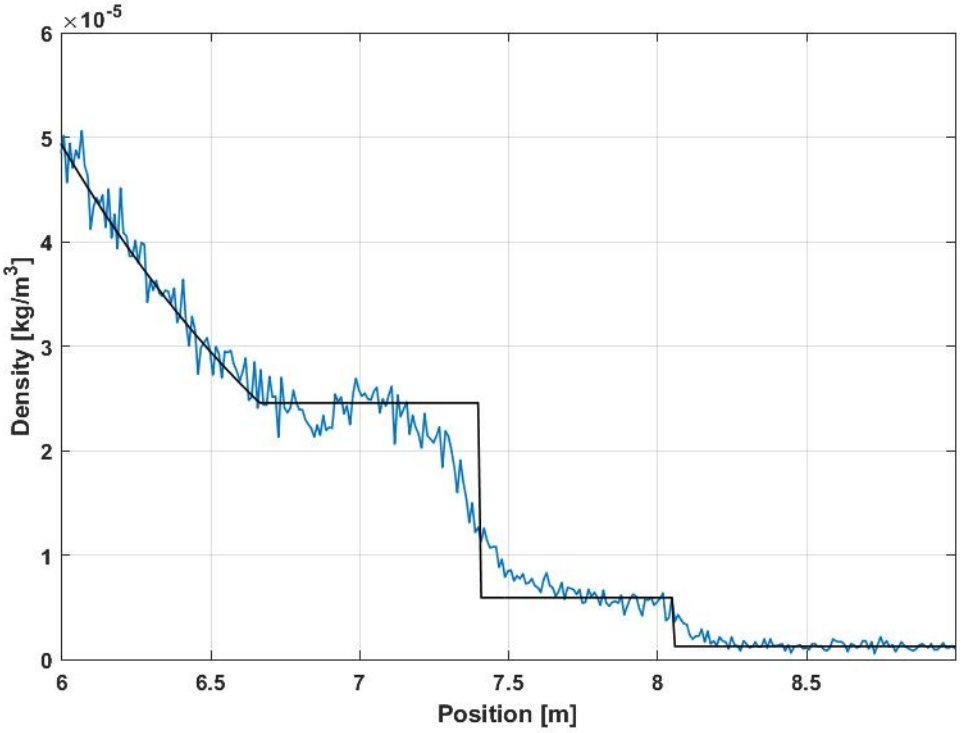


Figure 4.4: Shock and contact discontinuity of strong shock test

4.2 Code Flexibility and Stability Tests

4.2.1 Low Density Test

The first stability test performed was a low density test taken from test 2 in section 4.3.3 of Toro [30]. The test consists of two rarefaction waves moving away from each other and a stationary contact discontinuity at the center. The test had to be run a number of times until the proper number of virtual DSMC particles was found. If the number was too low, certain cells would exhibit zero densities as the density

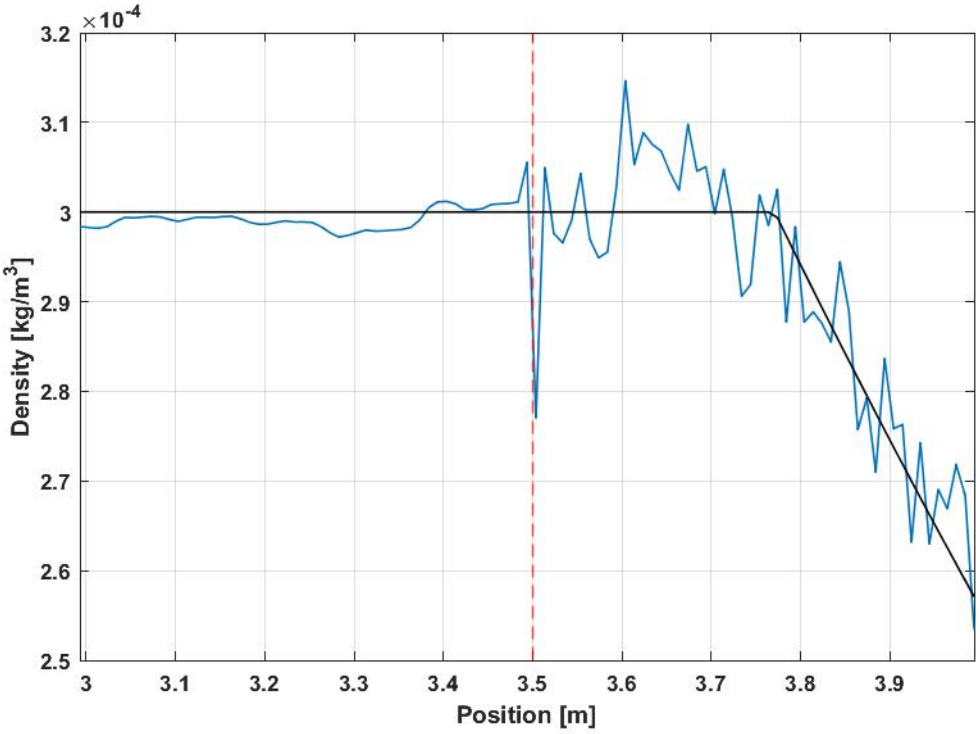


Figure 4.5: HPC interface of strong shock test

at the contact discontinuity became lower and lower. The final number of DSMC particles was $275,000$ for the particle domain from $0.5 - 1.0m$. The test ran for a total CPU time of 2.5 hours with 240 time steps and an initial Knudsen number of 0.34 throughout the domain, on the edge of continuum breakdown. There are a

Table 4.3: Low density test initial conditions

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
1.00×10^{-3}	-2	4.00×10^{-4}	1.00×10^{-3}	2	4.00×10^{-4}

few artifacts in the results of the test that indicate inaccuracies of the solver. In the fluid domain there is a mismatch between the CFD and ideal solution at $x = 0.5$, where the CFD solution overshoots the beginning of the rarefaction wave. There are a few oscillations further along the solution, and while the pressure and density plots appear to follow the ideal solution closely, there is a slight offset throughout

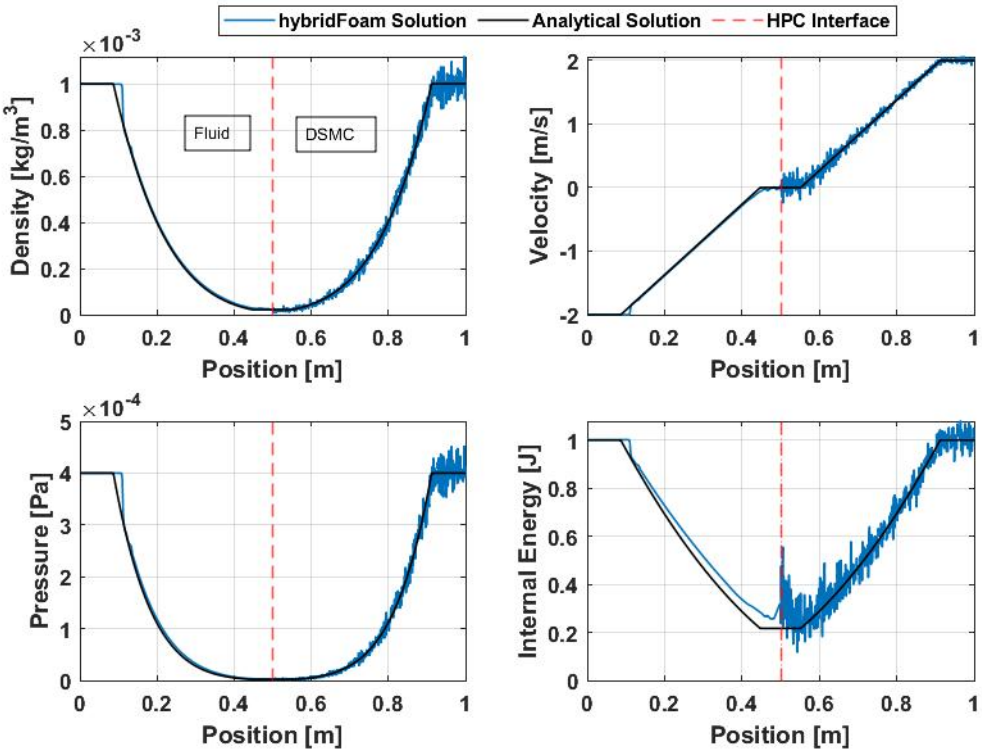


Figure 4.6: Low density rendering test of hybridFoam: $\Delta t = 0.15s$

the CFD solution, one that is magnified in the internal energy plot. It is possible that the overshoot at the beginning of the rarefaction wave as well as the smaller oscillations are caused by the inadequacies of the van Albada slope limiter, as the beginning of the rarefaction wave is characterised by a sharp change in slope. At the contact discontinuity, which also coincides with the hybrid interface, there is a spike in internal energy on either side. The equation used to calculate internal energy for each solver is given by

$$E_i = \frac{p}{(\gamma - 1)\rho} \tag{4.1}$$

from Toro [30] making it entirely dependant on the difference between the density and pressure solutions. This is also where the density and pressure values are at their lowest magnitude (on the order of 1×10^{-5} and 1×10^{-6} respectively), increasing the effects of numerical inaccuracies and variations from the particle regime. The DSMC solution follows much of the same patterns that have been seen before such as

increased statistical variations at large plateaued values. Besides this and the spike in internal energy at the HPC interface, the DSMC solution has good agreement with the ideal.

4.2.2 Multi-shock Test

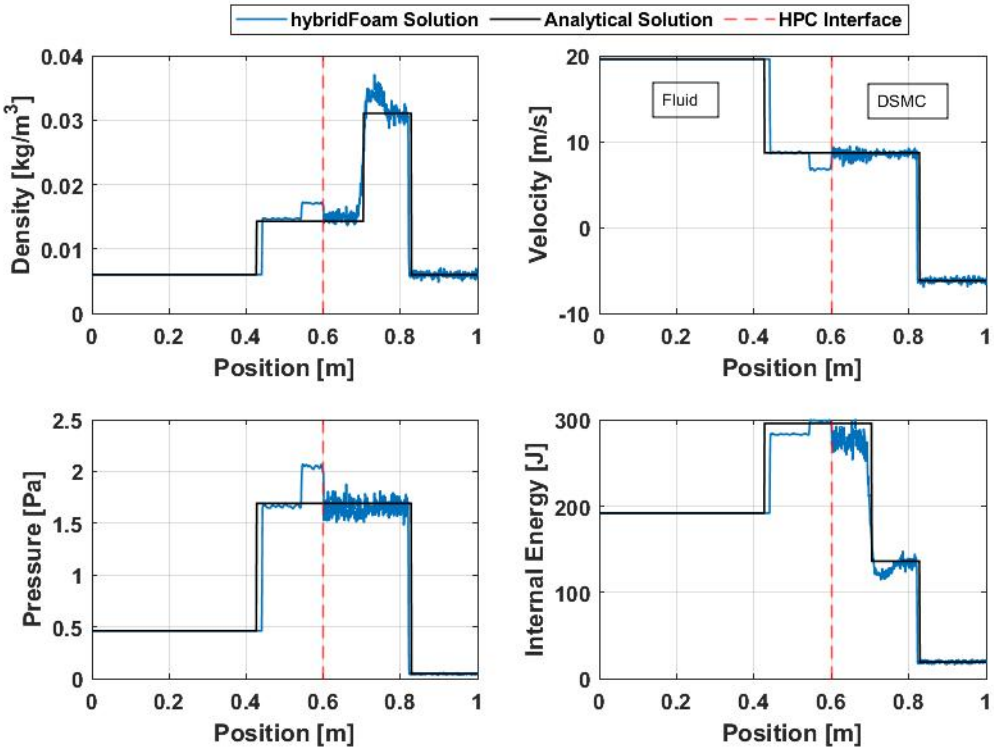


Figure 4.7: Mutli-shock test of hybridFoam: $\Delta t = 0.035s$

The multi-shock test case was intended to be a significant stress test of the hybridFoam code. The test shows the collision of two strong shocks, consisting of a pair of shocks travelling to the right and a contact discontinuity between them. In addition to the scenario being very dynamic, it involves passing shock information through the HPC interface, something that was not in the scope of development for this thesis, but will be necessary for the future development of hybridFoam. The test revealed a number of shortcomings within the solver, primarily in the capabilities of the fluid solver as well as the effectiveness of the interface. In figure 4.7 there is a large anomaly

on the fluid side of the HPC interface. The anomaly is a left moving shock that emanates from the interface after the contact discontinuity has passed into the DSMC region, this can be seen more readily in figure 4.8. The anomaly holds a relatively constant value of $1.7kg/m^3$ when isolating the density plot. It begins to form at approximately $\Delta t = 0.025s$ and has a wave speed of $-5.6m/s$, well below the local

Table 4.4: Multi-shock test initial conditions

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
6.00×10^{-3}	19.60	4.61×10^{-1}	5.99×10^{-3}	-6.20	4.61×10^{-2}

speed of sound which can be seen in appendix E. Although the anomaly extends into the fluid domain as if it had Dirichlet boundary conditions at the interface, it only extends into the first cell of the particle domain as a result of particle injection. It then quickly dampens out, however its lasting effects can be seen through the higher than nominal density values surrounding the contact discontinuity in the DSMC region.

As the passage of the right most shock did not produce similar artifacts when it passed into the DSMC region, the reason for the anomaly most likely relates to the uniqueness of the contact discontinuity. If there had been a similar anomaly created by the shock, then the cause would most likely be from the HPC boundary conditions, such as a be a feed back loop or a mismatch between the translated values. A contact discontinuity is simply defined as a discontinuity of density and temperature. There is no pressure change across the discontinuity, and no particle motion. It is clear that the density is not properly conserved at the interface, which could be driven by a temperature change from the fluid solver attempting to close the solution to he energy equation. In figure 4.7 the internal energy is the only term that matches the ideal solution in the anomalous shock, while in previous time steps the internal energy continuously undershoots the ideal solution (this can be seen in appendix E).

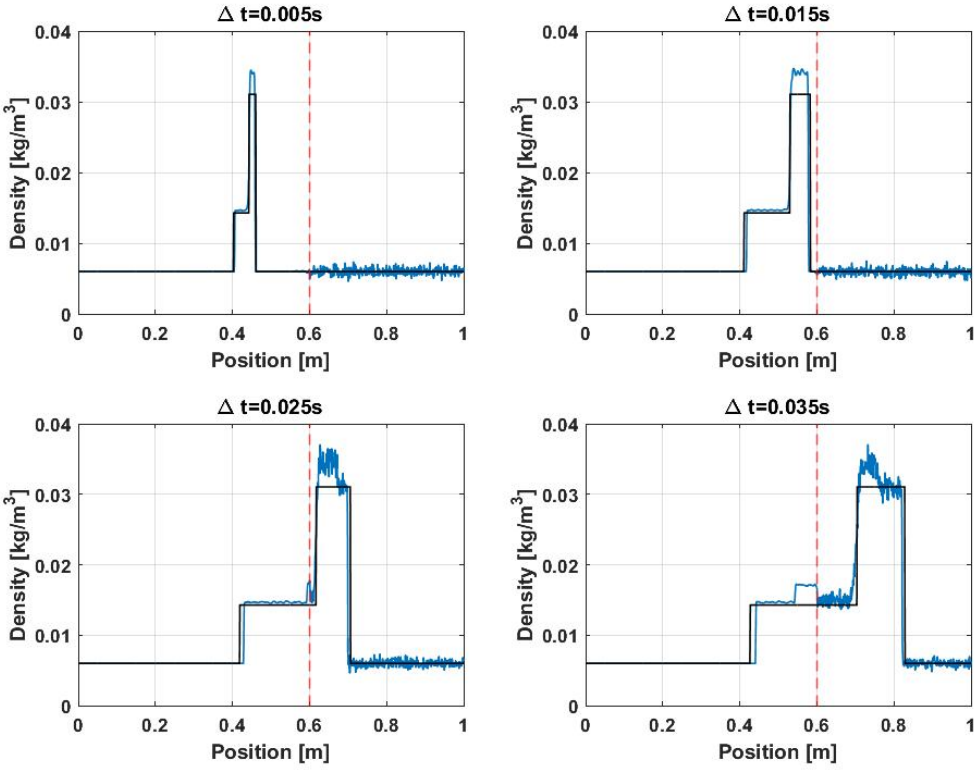


Figure 4.8: Mutli-shock test of hybridFoam

The other significant error present in the results of the multi-shock test is the poor performance of the fluid solvers slope limiter. The fluid solver used can select from a variety of limiters, however throughout every test the Van Albada limiter was used. A comparison between this limiter and the Van Leer limiter are shown in figure 4.9. The Van Leer limiter has much more significant overshoots and struggles to recover from severe oscillations after the contact discontinuity, while also severely overshooting the ideal solution after the shock. The Van Albada limiter produces a similar performance, with oscillations after the leftmost shock and discontinuity, however its undershoot behavior is negligible at the rightmost shock. However, its overshoot at the contact discontinuity is still significant. Instead of oscillating erratically, it oscillates slightly around a significantly larger value than nominal. Using an NS solver would likely prevent this initial overshoot as viscous effects would smear the contact

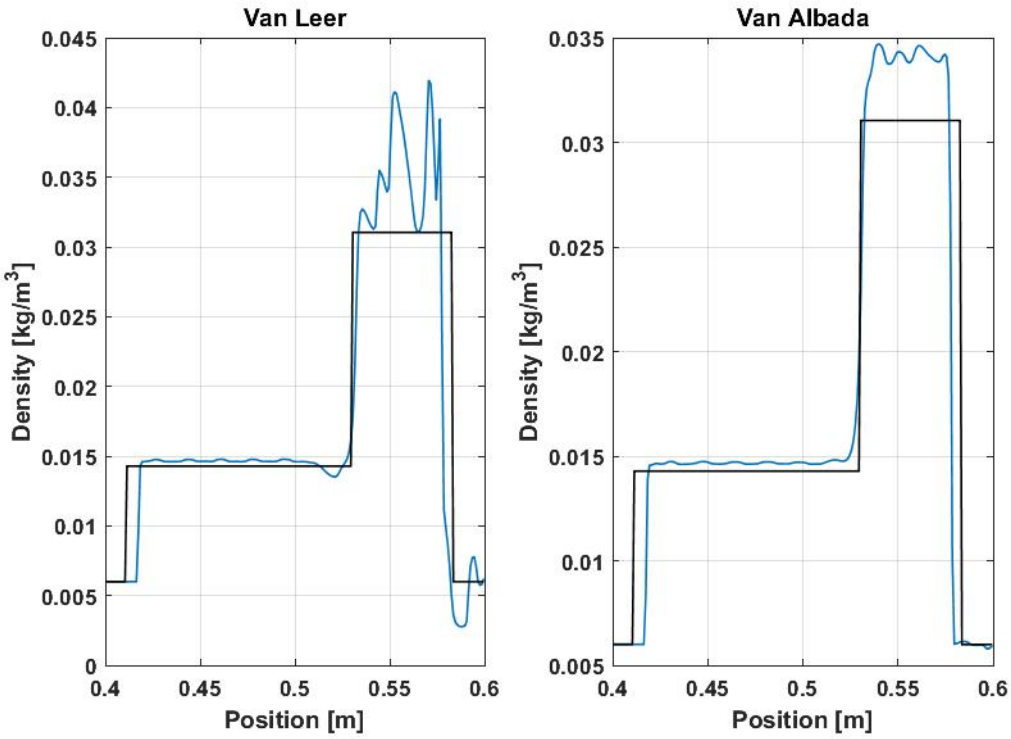


Figure 4.9: Example of slope limiter performance in the fluid domain during the multi-shock test: $\Delta t = 0.015s$

discontinuity and not place the burden of resolving the discontinuity on the slope limiter. Beyond the previously mentioned vertical errors in the multi-shock test, the fluid solver slightly leads the leftmost shock, and falls behind the rightmost shock. This may be an additional effect of the slope limiting scheme, which may have a delayed reactions to step responses in the system. It should also be noted that like the anomalous shock at $\Delta t = 0.035s$, which passed higher than average values into the DSMC regime, the overshoot from the Van Albada limiter has also passed its profile into the DSMC region. As previous other tests indicate that the DSMC solver does not exhibit overshoot behavior at contact discontinuities, this is the more likely cause of the density overshoot at the contact discontinuity. Beyond these increased values left over from the errors in the fluid solver, the standard smearing of the contact discontinuity is present in the DSMC solver.

4.2.3 Mirrored Shocktube Test

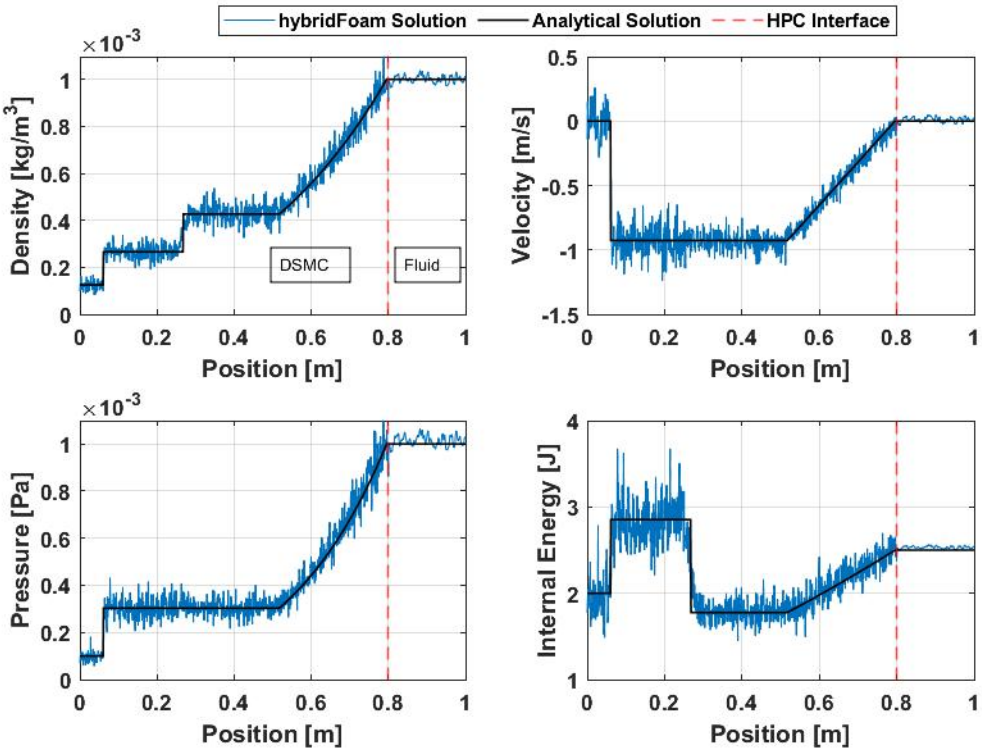


Figure 4.10: Mirrored shocktube test of hybridFoam: $\Delta t = 0.035s$

Table 4.5: Mirrored Shock Tube Initial Conditions

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
1.25×10^{-4}	0	1×10^{-4}	1×10^{-3}	0	1×10^{-3}

The final test of hybridFoam was a mirrored shocktube test. The test was completed successfully with the same initial conditions as the previous shocktube test, however only $130,000$ DSMC particles were used, allowing the test to be completed in just under two hours. Although the results yielded no additional information, the test did reveal hard coded values in the fluid boundaries at the HPC interface. Currently the boundaries perform a linear interpolation using the cell centered (fluid) and cell averaged (DSMC) values of the boundary cells on either side of the interface. Unfortunately this means the boundary face cannot be referenced via its tag (fluidBound)

in order to retrieve the necessary values. Instead cell indexing is used, and depending on which side of the domain the interface is on, the index used is either 0 or the number of cells in the domain, retrieved with the member function nCells. As the current HPC boundary condition code must be updated in order to run the mirrored test, this was an additional motivation to keep the boundary conditions defined as coded fixed value boundaries rather compiling them in their own library. However, beyond this no other hardcoded values were found as the DSMC boundary conditions can retrieve boundary cell data via the tag of each face in the domain.

4.3 Computational Efficiency

Table 4.6: Computation Time For Various Methods

Test	**CPU Time** [s]
Single Core DSMC	444.08
Single Core HPC	245.65
Multi-core DSMC	276.17
Single Core Fluid	0.72

As discussed in section 2.2.2, the two primary factors that effect the computation time of a DSMC simulation are: the number of virtual particles and the number of particle collisions. At the end of his book detailing the DSMC method, G.A. Bird [4]

Table 4.7: Initial Conditions for Computation Time Analysis

ρ_L [kg/m^3]	u_L [m/s]	p_L [pa]	ρ_R [kg/m^3]	u_R [m/s]	p_R [pa]
1.25×10^{-5}	0	1×10^{-4}	1×10^{-4}	0	1×10^{-3}

provides some insight into why he chose particular simulations to include as examples. He explains that the examples in his book, all of which are one dimensional, were chosen because they could be run in under 24 hours on a "contemporary top of

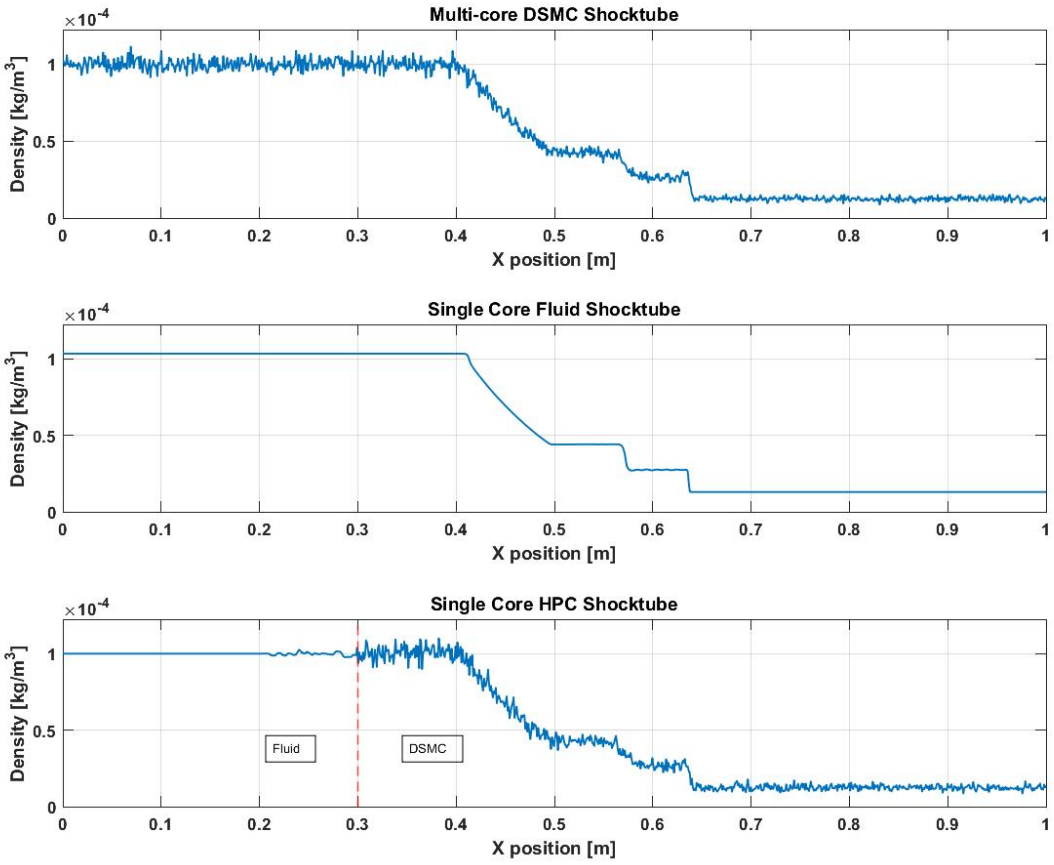


Figure 4.11: Shocktube simulation with results from DSMC, Fluid, and HPC methods: $\Delta t = 0.035s$

the line computer". He goes on to explain that two dimensional versions of these tests would exceed the 24 hour constraint and that "it is not yet possible to make three dimensional calculations for non-trivial applications in the continuum regime". Although his book was written in 1994, this showcases the primary limitation of the DSMC method; computation time. HPC codes are intended to alleviate the computational loads of pure DSMC methods, so to verify the computational gains of hybridFoam, four separate runs of the same shocktube scenario were performed and their results were compared. The four runs, listed in table 4.6, differed in their method and number of processing cores used. The DSMC solver used was the customized

DSMC code detailed in chapter 3.1.1. As expected, the single core DSMC run was the slowest of the four, taking over 7 minutes to complete while the single core hybrid run saw an increase of computation time of over 100% with a very similar resolution in the particle regime. HybridFoam provided computational gains comparable to the multi core DSMC run, which was performed across 4 processors. The final run was performed with OpenFOAM's built in fluid solver, rhoCentralFoam, which produced an Euler solution in less than a second. The shocktube test was run with reduced densities, listed in table 4.7, which gave Knudsen Number values of 2.7 and 0.34 for the low and high density regimes, respectively. This placed the simulation in the rarefied regime, meaning that the full DSMC solution was more accurate. However, the randomness of the solution and the smoothing out of the contact discontinuity from molecular diffusion (as can be seen in figure 4.3) are completely preserved in the hybrid solution. The only inaccuracy the hybrid solution introduces in this case is the overly smooth solution between $x = 0$ and $x = 0.3$ meters, which was approximated with an Euler solution. This leads to the question of weather or not the additional computation time is worth the added fidelity in this portion of the domain? The full Euler solution gave almost instantaneous results, but of course has no randomness in the solution besides small oscillations between the contact discontinuity and shock due to its slope limiter.

Chapter 5

CONCLUSION

The goal of this thesis was to develop a hybrid particle continuum code, hybridFoam, capable of modeling a 1D sod shock scenario using an Euler and DSMC solver. The solver was developed as a custom solver in the Open Source Field Operation and Manipulation code base, OpenFoam, and utilizes numerous OpenFoam utilities. The primary benchmark was met and a collection of other test cases were run in order to assess the solvers current limitations. These tests included sod shock and strong shock validation tests and a multi-shock, low density, and mirrored sod shock stress tests. Each test case was compared to an ideal inviscid solution produced by a solver written in Matlab, based on code from E.F. Torro [30]. From the validation tests, it was found that statistical scatter from the particle domain of the solver interfered with the fluid solution and the resolution of the particle domain proved to be the limiting factor on computation time. Additionally, the particle solution in both tests exhibited molecular diffusion at contact discontinuities and a smearing of shocks at low virtual particle densitiesdue to anticipated viscous effects. The HPC interface exhibited a dip in density in the particle boundary cells, indicating possible shortcomings in the particle injection code or a mismatch in converting viscous flow to inviscid flow. Beyond this, the solver had good agreement with the inviscid solution. The low density stress case showed that low density regions need to be treated carefully in order to avoid non-physical values. It also showed an inaccuracy in the fluid solution at the beginning of its rarefaction wave, where hybridFoam had a slower response time to the jump in density. An additional artifact was a spurious increase in internal energy at the HPC interface, however this was most likely due to negligi-

ble variations in the pressure and density solutions which became magnified due to the low densities involved. The multi-shock stress test showed numerous errors in the fluid solver. These included a large overshoot and strong oscillations caused by the slope limiter attempting to resolve a contact discontinuity which could have been avoided by using an NS solver which would have smeared the discontinuity. There was also a slight latency when responding to discontinuities, and an anomalous shock wave that emanated from the HPC interface after a contact discontinuity passed into the particle region. All the inaccuracies in the DSMC solution could be traced back to the fluid solution. The mirrored sod-shock test showed identical results to the validation test case, however it was found that the code for the fluid interface BC's had to be changed in order to identify if the HPC boundary was on the left or right hand side of the domain. In addition to the validation and stress tests, the computational gain of hybridFoam was confirmed by comparing its performance to a full DSMC solution.

Chapter 6

FUTURE WORK

6.0.1 Code Development

Although OpenFOAM is a useful and versatile code base, the fact that it is open source means it has certain areas it is stronger in than others. The DSMC solver is unfortunately largely incompatible with the majority of the standard OpenFOAM utilities. In order to set boundary conditions, the user must typically dictate analytical BC's in the 0 directory on a field by field basis and geometric BC's in the blockmesh dictionary. dsmcFoam, the standard DSMC solver in OpenFOAM, does not use this format. Its BC's are unique in that they must act on the particles that come into contact with it, so the solver has its own set of BC's broken into wall inter-action models and inflow boundary models. These inflow models take the incoming velocity and temperature data from fixed value BC's in the 0 dictionary and imitate inflow/outflow conditions normally available in openfoam. However, dsmcFoam is also limited to homogeneous initial density distributions, and rather than having their own independent density values, the inflow models read in this uniform density in DSMC initialize and only allow for one density per species. Although hybridFoam has adapted dsmcFoam's initialisation code and allows for non-uniform densities,this architecture unfortunately limits the number of available BC's for hybridFoam.Currently the only compatible inflow model is the one used for particle injection at the HPC interface while the standard wall interaction models are still compatible. Future development of more varied boundary conditions for the DSMC portion of hybridFoam would be advantageous for the adaptability of the solver and its overall usefulness as a research tool for Cal Poly. Additionally, a current limitation

of each set of models is that only one inflow model or wall interaction model can be applied for an entire domain, with the dsmc Properties dictionary providing the name of the model and the source code for each model being tasked with finding and interacting with the appropriate faces in the mesh. Obviously, it would be good to be able to assign different models to different facings from the input dictionaries so the solver could again, be more adaptable to possible user needs. An additional issue with dsmcFoam and by extension, hybridFoam is the absence of proper methods to establish Dirichlet BC's. Although zeroGradient and symmetry BC's were used in many of the test simulations to provide constant values at the boundaries, their interaction with hybridFoam was not predictable. As all the simulations were one dimensional, the BC's should have been equivalent, however a common issue that would arise is using one or the other, depending on the simulation, would cause particles to disappear at the boundary. Sometimes switching BC's would solve the issue, but sometimes the domain would need to be extended far enough so the low pressure wave wouldn't effect the area of interest, as in the case of the multi-shock test. So the development of a Dirichlet inflow BC would be particularly helpful to hybridFoam in its current state. Focusing on the fluid solver in hybridFoam, adapted from rhoCentralFoam, the solver produced a number of errors due to failings in the slope limiter and proved to be inaccurate in extreme conditions, as can be seen in the multi-shock test. Although the solver is very computationally efficient, it would be useful to investigate other fluid solver schemes, slope limiters, or methods that can improve the accuracy of the solver, especially when resolving discontinuities. In addition to general inaccuracies, the solver also proved to be sensitive to nonphysical values, extreme initial conditions, and mesh geometry. For example, whenever a zero density is produced in the particle domain, this causes hybridFoam to crash when the fluid solver solves for the energy equation. Although this difficulty was overcome through an iterative search to find a stable number of simulation particles, it would be more advantageous

to achieve this through more automatic means, such as dynamically splitting and merging simulation particles as needed. In addition to this, efforts to increase the robustness of the solver and to add more error catches to the source code may make it more accommodating to new users who are not experienced in OpenFOAM or in diagnosing crashes. Although the documentation details parallel capability when using the splitmesh regions utility, there are currently bugs in the process caused by interference from automatically generated directories from the utility. Unfortunately a workaround has yet to be made available to the OpenFOAM community at large. Having parallel capability with hybridFoam could greatly improve computation time, and in turn provide a higher feasible limit to resolution of the DSMC solution, and speed future development efforts. In this same spirit, although efforts were made to install OpenFOAM on the Bishop computing cluster at Cal Poly, only the standard OpenFOAM libraries were made available. Any attempts to compile and run custom libraries were unsuccessful. Access to the computational power of Bishop would compound the gains in computation time nicely if the solver could be run in parallel. Additionally, as the cluster can be accessed remotely, compatibility with OpenFOAM code development makes access to Bishop even more attractive when the current COVID-19 pandemic is taken into consideration. Finally the results of this thesis could be adapted to improve existing computational tools developed at Cal Poly, namely the in-house DSMC solver SINATRA which was developed to model pulsed plasma thrusters [18]. Using the lessons learned through this effort to add hybrid capabilities to the solver would be a valuable addition to Cal Poly's computational capabilities.

6.0.2 HPC Enhancements

hybridFoam is a very simple solver, and lacks many of the features available to HPC codes. The following is a selection of potential upgrades available to hybridFoam if it sees further development.

1. **Dynamic HPC Interface:** Every fully fledged HPC solver attempts to predict continuum breakdown in order to efficiently place the HPC interface. A variety of breakdown parameters are available with the majority based on Knudsen number. The switch from a geometrically fixed HPC interface to a dynamic interface would make hybridFoam much more useful as an investigative tool and help to optimize computation time as well as accuracy if continuum breakdown is predicted correctly.

2. **Application of Chapman-Enskog Distribution:**The Chapman-Enskog distribution describes the velocity distribution of particles just as continuum breakdown occurs. As a result, it can be used to provide a point of translation between the fluid and particle domains in hybridFoam and provide a smoother transference of data at the HPC interface.

3. **Effective Dampening of Statistical Scatter:** Although linear interpolation and a relaxation coefficient are currently implemented at the HPC interface of hybridFoam, the statistical scatter from the particle solution has not been eliminated from the fluid region. As the fluid region is considered continuum, it would be ideal to achieve a negligible amount of statistical scatter.

4. **Improving Particle Injection:** The current particle injection method injects particles at interface with a velocity proportional to the temperature of the incoming fluid flow. Modifying how these particles are initialized in the bound-

ary cell could dampen the current inconsistencies at the interface. A Poisson distribution could be used to dictate the velocity of the incoming particles and the initial position of these particles within the cell could be randomized by propagating their velocities over a random fraction of Δt.

5. **Utilizing an NS Solver:** Currently hybridFoam assumes inviscid flow in the fluid domain. Although the fluid solver itself has the capability to model viscous flow, accommodations would need to be made to properly implement it in an HPC architecture. These include determining the thermal conductivity of flow entering the DSMC region and having a separate mesh size in the fluid region in order to capture temperature and momentum diffusion.

6. **2D and 3D Geometries:** Currently hybridFoam has only been developed with 1D geometries in mind. To improve its usefulness as a modeling and sim tool, it will be necessary to account for higher dimension geometries.

7. **Integration with Various Flow Solvers:** Adding the ability to model reactive flows, plasma flows, or other unique flow models to the DSMC and fluid subsets of hybridFoam would be useful. This however would most likely be outside the scope of Cal Poly's research interests and offer a learning experience to the developer rather than added value for the aerospace department.

Bibliography

[1] D. A. Anderson, J. C. Tannehill, and R. H. Pletcher. Governing Equations of Fluid Mechanics and Heat Transfer. *Computational Fluid Mechanics and Heat Transfer*, pages 181–188, 1984.

[2] R. Arslanbekov and V. Kolobov. Adaptive Kinetic-Fluid Models for Expanding Plasmas. *Journal of Physics: Conference Series*, 1031(1), 2018.

[3] L. M. Bermúdez, K. J. Barnhart, and C. W. Brunner. Modeling, simulation, and validation of plume impingement effects on the cygnus spacecraft. *Journal of Spacecraft and Rockets*, 55(2):427–436, 2018.

[4] G. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows - G. A. Bird*. Oxford University Press Inc., New York, 1st edition, 1994.

[5] J. Blazek. Unstructured Finite-Volume Schemes. *Computational Fluid Dynamics: Principles and Applications*, 5:121–166, 2015.

[6] I. D. Boyd and T. R. Deschenes. Hybrid Particle-Continuum Numerical Methods for Aerospace Applications. (1), 2012.

[7] D. B. Chrisey and G. K. Hubler. *Pulsed laser deposition of thin films*. J. Wiley, 1994.

[8] S. Colin. Single-Phase Gas Flow in Microchannels. *Heat Transfer and Fluid Flow in Minichannels and Microchannels*, pages 11–102, 2013.

[9] M. Gad-El-Hak. The Fluid Mechanics of Microdevices. *Journal of Fluids Engineering, Transactions of the ASME*, 121(1):5–33, mar 1999.

[10] M. Gad-el Hak. The {Fluid} {Mechanics} of {Microdevices}—{The} {Freeman} {Scholar} {Lecture}. *Journal of Fluids Engineering*, 121(1):5–33, 1999.

[11] A. L. Garcia and B. J. Alder. Generation of the Chapman-Enskog Distribution. *Journal of Computational Physics*, 140(1):66–70, 1998.

[12] K. Gott. *A HYBRID CFD-DSMC MODEL DESIGNED TO SIMULATE RAPIDLY RAREFYING FLOW FIELDS AND ITS APPLICATION TO PHYSICAL VAPOR DEPOSITION*. Phd, Pennsylvania State University, 2015.

[13] C. J. Greenshields. *OpenFOAM User Guide*. Number July. The OpenFOAM Foundation, 8th edition, 2020.

[14] C. J. Greenshields, H. G. Weller, L. Gasparini, and J. M. Reese. Implementation of semi-discrete, non-staggered central schemes in a colocated, polyhedral, finite volume framework, for high-speed viscous flows. *International Journal for Numerical Methods in Fluids*, 2009.

[15] E. Jun and I. D. Boyd. Assessment of the LD-DSMC hybrid method for hypersonic rarefied flow. *Computers and Fluids*, 166:123–138, 2018.

[16] V. I. Kolobov, R. R. Arslanbekov, V. V. Aristov, A. A. Frolova, and S. A. Zabelok. Unified solver for rarefied and continuum flows with adaptive mesh and algorithm refinement. *Journal of Computational Physics*, 223(2):589–608, may 2007.

[17] A. Kurganov and E. Tadmor. New High-Resolution Central Schemes for Nonlinear Conservation Laws and Convection-Diffusion Equations. *Journal of Computational Physics*, 160(1):241–282, 2000.

[18] D. Lunde. A HOMEGROWN DSMC-PIC MODEL FOR ELECTRIC PROPULSION PLUMES, 2019.

[19] G. B. Macpherson, N. Nordin, and H. G. Weller. Particle tracking in unstructured, arbitrary polyhedral meshes for use in CFD and molecular dynamics. *Communications in Numerical Methods in Engineering*, 25(3):263–273, 2009.

[20] T. Matsuda, H. Isaka, H. Murata, and H. Boffin. Application of the Molecular Hydrodynamics to Accretion Flows. *Numerical Modeling of Space Plasma Flows*, 359:270–275, 2006.

[21] F. peng Bai, Z. hua Yang, and W. gang Zhou. Study of total variation diminishing (TVD) slope limiters in dam-break flow simulation. *Water Science and Engineering*, 11(1):68–74, 2018.

[22] I. Sagert, W. Bauer, D. Colbry, J. Howell, R. Pickett, A. Staber, and T. Strother. Hydrodynamic shock wave studies within a kinetic Monte Carlo approach. *Journal of Computational Physics*, 266:191–213, 2014.

[23] T. E. Schwartzentruber. CONSISTENT CONTINUUM-PARTICLE MODELING OF HYPERSONIC FLOWS AND DEVELOPMENT OF HYBRID SIMULATION CAPABILITY AIR FORCE RESEARCH LABORATORY Space Vehicles Directorate 3550 Aberdeen Ave SE AIR FORCE MATERIEL COMMAND KIRTLAND AIR FORCE BASE, NM 87117-5776. Technical report, University of Minnesota, Minneapolis, 2016.

[24] T. E. Schwartzentruber, L. C. Scalabrin, and I. D. Boyd. Multiscale particle-continuum simulations of hypersonic flow over a planetary probe. *Journal of Spacecraft and Rockets*, 45(6):1196–1206, 2008.

[25] Y. Shi, Q. Zhao, F. Fan, and G. Xu. A new single-blade based hybrid CFD method for hovering and forward-flight rotor computation. *Chinese Journal of Aeronautics*, 24(2):127–135, apr 2011.

[26] E. A. Silber, M. Boslough, W. K. Hocking, M. Gritsevich, and R. W. Whitaker. Physics of meteor generated shock waves in the Earth's atmosphere – A review, aug 2018.

[27] N. Singh and T. E. Schwartzentruber. Coupled Vibration-Rotation Dissociation Model for Nitrogen from Direct Molecular Simulations. In *47th AIAA Thermophysics Conference*, Reston, Virginia, jun 2017. American Institute of Aeronautics and Astronautics.

[28] R. Singh and R. K. Soni. Laser-Induced Heating Synthesis of Hybrid Nanoparticles. In *Noble Metal-Metal Oxide Hybrid Nanoparticles: Fundamentals and Applications*, pages 195–238. Elsevier, oct 2018.

[29] Q. Sun and I. D. Boyd. Evaluation of Macroscopic Properties in the Direct Simulation Monte Carlo Method. *Journal of Thermophysics and Heat Transfer*, 19(3):329–335, jul 2005.

[30] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Number October 2013. Springer-Verlag Berlin Heidelberg, New York, 2nd edition, 1999.

[31] G. D. Van Albada, B. Van Leer, and W. Roberts. A Comparative Study of Computational Methods in Cosmic Gas Dynamics. *Astronomy and Astrophysics*, (108):76–84, 1982.

[32] H. K. Versteeg and W. Malalasekera. *An Introduction to Parallel Computational Fluid Dynamics*. Pearson Education Limited, Essex, England, 2nd edition, 2007.

[33] M. D. Weinberg. Direct simulation Monte Carlo for astrophysical flows - I. Motivation and methodology. *Monthly Notices of the Royal Astronomical Society*, 438(4):2995–3006, 2014.

[34] C. White, M. K. Borg, T. J. Scanlon, S. M. Longshaw, B. John, D. R. Emerson, and J. M. Reese. dsmcFoam+: An OpenFOAM based direct simulation Monte Carlo solver. *Computer Physics Communications*, 224:22–43, 2018.

[35] M. J. Wright, D. K. Prabhu, and E. R. Martinez. Analysis of apollo command module afterbody heating part I: AS-202. *Journal of Thermophysics and Heat Transfer*, 20(1):16–30, 2006.

APPENDICES

Appendix A

RHOCENTRALFOAM ALGORITHM

The OpenFOAM fluid solver used to form hybridFoam follows a Kirganov and Tadmor solution scheme. An abbreviated version of the algorithm given by Greenshields et al. [14] is presented here.

1. $t = t + \Delta t$

2. Evaluate $\rho_{f\pm}$, $(\rho\vec{\mathbf{u}}_{f\pm})$, and $T_{f\pm}$ from cell centered ρ, $(\rho\vec{\mathbf{u}})$, and T using a flux limiting scheme. The van Leer flux limiter is given by

$$\beta(r) = \frac{r + |r|}{1 + r} \tag{A.1}$$

   where r is given by

$$r = 2\frac{\mathbf{d} \cdot (\Delta\Psi)_P}{(\Delta_{\mathbf{d}}\Psi)_f} - 1 \tag{A.2}$$

   where $\mathbf{d}$ is the vector connecting the center of the origin cell P to the neighbor cell center N.

3. Calculate: $\vec{\mathbf{u}}_{f\pm} = (\rho_{f\pm}\vec{\mathbf{u}}_{f\pm})/\rho_{f\pm}$, $p_{f\pm} = \rho_{f\pm}RT_{f\pm}$;$\phi_{f\pm} = \mathbf{S}_{f\pm} \cdot \vec{\mathbf{u}}_{f\pm}$; $c_{f\pm} = \sqrt{\gamma RT_{f\pm}}$.

4. Calculate convective derivatives from $f\pm$ interpolations using the following equation:

$$\sum_f \phi_f \Psi = \sum_f [1/2\phi_{f+}\Psi_{f+} + 1/2\phi_{f-}\Psi + \omega_f(\Psi_{f-} - \Psi_{f+})] \tag{A.3}$$

where the weighting coefficient, ω, is defined by $\omega_f = \frac{1}{2}max(\psi_{f+}, \psi_{f-})$ and the $+$ and $-$ subscripts indicate flow into or out of the cell face owner. Additionally find ∇p via

$$\sum_f \mathbf{S}\Psi_f = \sum_f \left[\frac{1}{2}\mathbf{S}_f\Psi_{f+} + \frac{1}{2}\mathbf{S}_f\Psi_{f-}\right]. \tag{A.4}$$

Both equations typically use the van Leer limiter, however several slope limiter options are available.

5. Update the thermal transport terms $\mathbf{T}_{exp}$, μ and k where $\mathbf{T}_{exp}$ is the stress tensor given by

$$\mathbf{T}_{exp} = \mu\left[(\nabla\vec{\mathbf{u}})^T - \frac{2}{3}tr(\nabla\vec{\mathbf{u}})\mathbf{I}\right] \tag{A.5}$$

6. Solve the conservation of mass equation for ρ

$$\frac{\partial\rho}{\partial t} + \nabla\cdot[\rho\vec{\mathbf{u}}] = 0 \tag{A.6}$$

7. Solve the momentum equation for $\rho\vec{\mathbf{u}}$

$$\left(\frac{\partial(\rho\vec{\mathbf{u}})}{\partial t}\right)_I + \nabla\cdot[\vec{\mathbf{u}}(\rho\vec{\mathbf{u}})] + \nabla p = 0 \tag{A.7}$$

8. Update $\vec{\mathbf{u}}$ from $(\rho\vec{\mathbf{u}})$ and ρ.

9. Solve for $\vec{\mathbf{u}}$ from

$$\left(\frac{\partial(\rho\vec{\mathbf{u}})}{\partial t}\right)_V - \nabla\cdot(\mu\nabla\vec{\mathbf{u}}) - \nabla\cdot(\mathbf{T}_{exp}) = 0 \tag{A.8}$$

10. Solve the energy equation by first solving for total energy density, (ρE), from

$$\left(\frac{\partial(\rho E)}{\partial t}\right)_I + \nabla \cdot [\vec{\mathbf{u}}((\rho E) + p)] + \nabla \cdot (\mathbf{T} \cdot \vec{\mathbf{u}}) = 0 \tag{A.9}$$

11. Update T with the following

$$T = \frac{1}{c_v}\left(\frac{(\rho E)}{\rho} - \frac{|\vec{\mathbf{u}}|^2}{2}\right) \tag{A.10}$$

    while using the updated values for ρ, (ρE), and $\vec{\mathbf{u}}$

12. Solve for T from

$$\left(\frac{\partial(\rho c_v T)}{\partial t}\right)_V - \nabla \cdot (k \nabla T) = 0 \tag{A.11}$$

13. Update p via $p = \rho R T$

14. If $t = t_{end}$ stop, else repeat.

It is important to note that steps 7 and 10 are predictor steps while steps 9 and 12 are correction steps.

Appendix B

VALIDATION OF SPLITMESHREGIONS

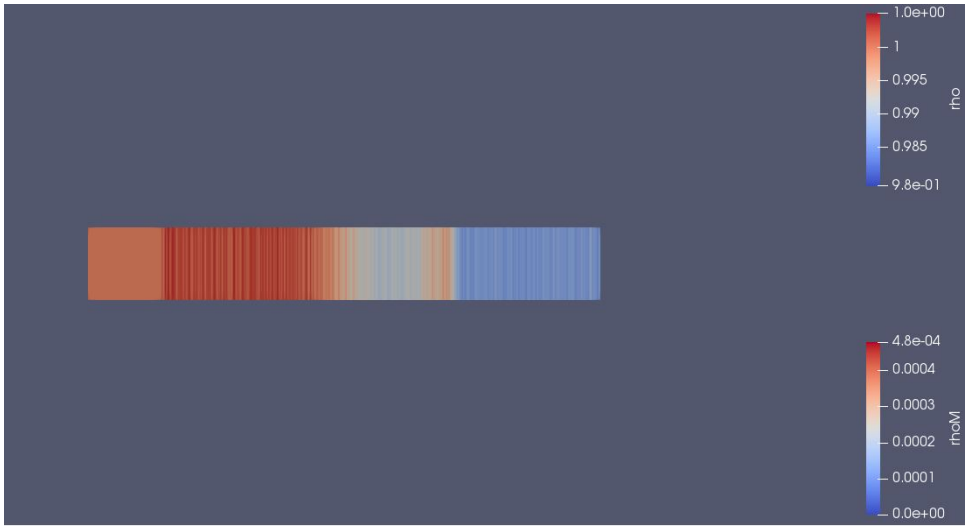
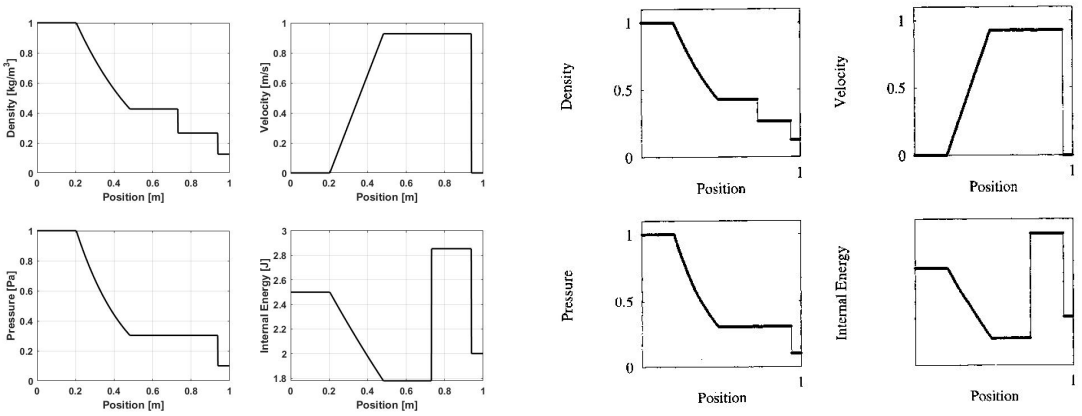


Figure B.1: Paraview color map of splitMeshRegions validation. The left CFD domain had homogeneous conditions held throughout the runtime while the right DSMC domain had a shocktube simulation. Zero gradient boundary conditions were used on all faces. Expected results were achieved.

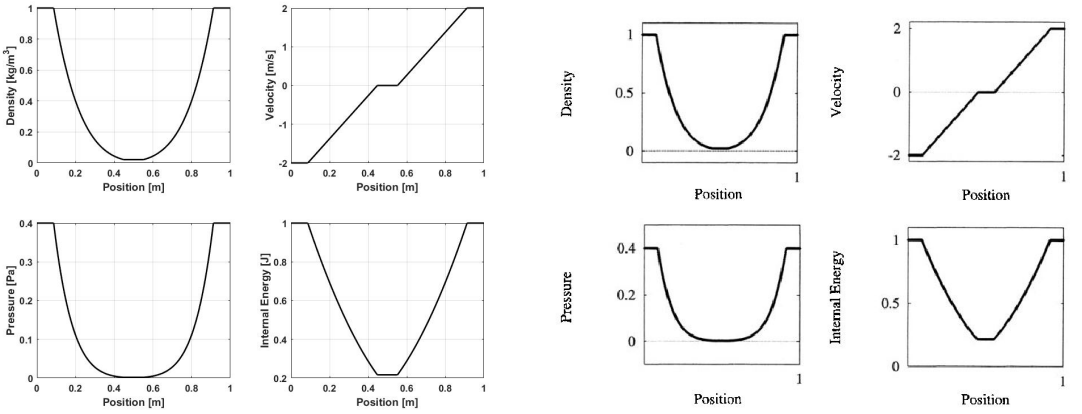
Appendix C

VALIDATION OF IDEAL SHOCKTUBE SOLVER MATLAB CODE



(a) (b)

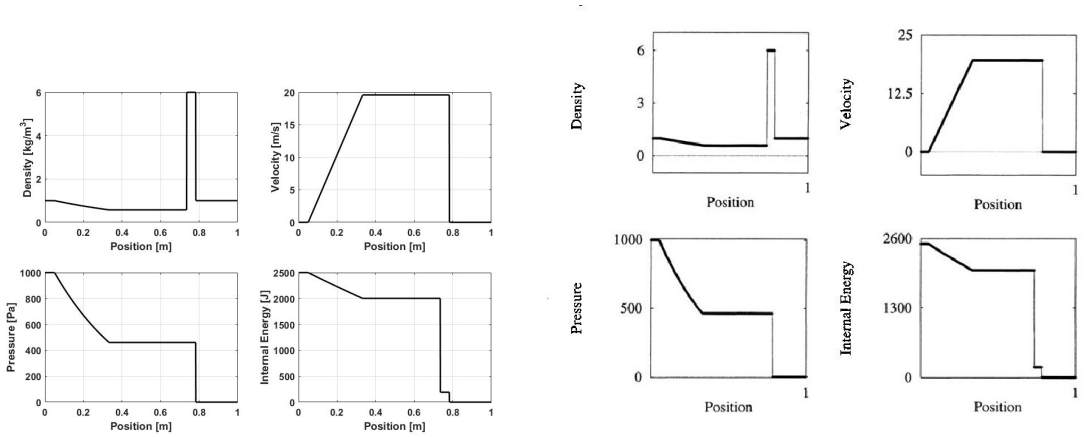
Figure C.1: Validation test case 1. Plot (a) is the adapted Matlab code, plot (b) is the ideal solution from Torro [30]



(a) (b)

Figure C.2: Validation test case 2. Plot (a) is the adapted Matlab code, plot (b) is the ideal solution from Torro [30]

In order to validate the Matlab code adapted from E.F. Torro's [30] three test cases were ran. Each case was a variation of the sod shocktube test case with pressure,



(a) (b)

Figure C.3: Validation test case 3. Plot (a) is the adapted Matlab code, plot (b) is the ideal solution from Torro [30]

Table C.1: Initial conditions for validation tests of the ideal shocktube solver

Test	ρ_L	u_L	p_L	ρ_R	u_R	p_R	Δt
1	1.0	0.0	1.0	0.125	0.0	0.1	0.25
2	1.0	-2.0	0.4	1.0	2.0	0.4	0.15
3	1.0	0.0	1000.0	1.0	0.0	0.01	0.012

density, and velocity initial conditions divided into left and right domains. The initial conditions are given in table C.1.

Appendix D

RESULTS OF THE ONE DIRECTIONAL FLOW ASSUMPTION

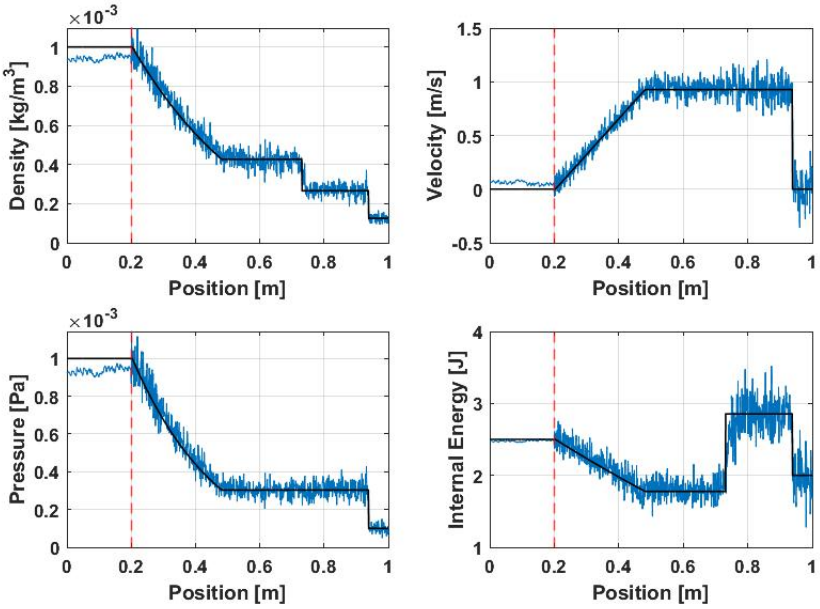


Figure D.1: Shocktube test case with one directional flow boundary conditions

Appendix E

ADDITIONAL MULTI-SHOCK TEST DATA

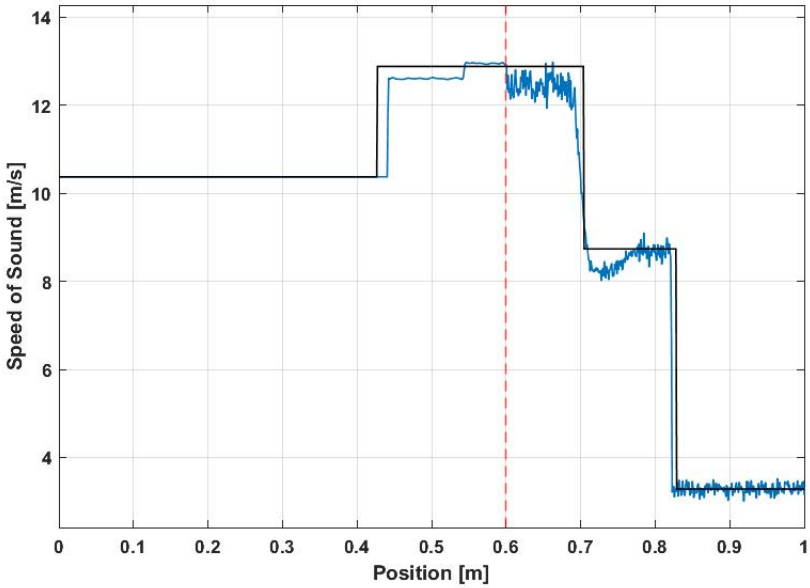


Figure E.1: Speed of sound results for Multi-shock test case: $\Delta t = 0.035s$

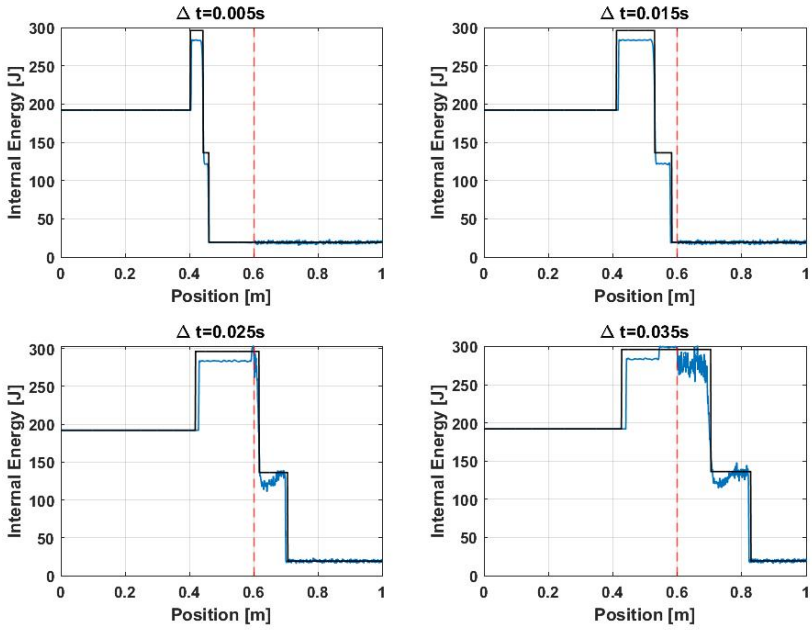


Figure E.2: Internal energy results for Multi-shock test at various time steps