

A STUDY INTO VALIDATING A COUPLED METHOD OF CHARACTERISTICS AND
DIRECT SIMULATION MONTE CARLO METHOD AGAINST EMPIRICAL DATA

by

ANDREW D. BROWN
B.S. University of Central Florida, 2012

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Sciences in Aerospace Engineering
in the Department of Mechanical and Aerospace Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2020

Major Professor: Kareem Ahmed

ABSTRACT

The following will outline the methodology and results of validating a coupled Method of Characteristics (MOC) and Direct Simulation Monte Carlo (DSMC) method. This research focused specifically on modeling plume impingement, induced by Reaction Control System (RCS) thrusters that flew on the National Aeronautics and Space Administration's (NASA's) space shuttle Discovery. For each simulation, the continuum portion of the RCS thruster was simulated using MOC for solving hyperbolic Partial Differential Equations (PDEs) and computed with the NASA code, Reacting and Multi-phase Program (RAMP). The solution was then implemented as a starting condition into the NASA DSMC code, Direct Simulation and Monte Carlo Analysis Code (DAC). Typically, DSMC models rely on code-to-code validation for fidelity.

The significance of this research is in its ability to validate its models against empirical data. Prior to computing solutions for these simulations, the mesh size and structure were optimized and many variants of DSMC input parameters were iterated on in order to acquire a reliable, mesh-independent, fully optimized numerical solution. This research will discuss the mathematical formulation of MOC for nozzle flow and DSMC for rarefied gases. Additionally, it will provide an explanation of how to implement these mathematical concepts into the two solvers: RAMP and DAC. Ultimately, this research will demonstrate that the overall process illustrated produces results in good agreement with empirical data. As a consequence, the methodology presented is granted an increased level of confidence and will greatly contribute to the aerospace industry and its effort in understanding and predicting rarefied flow fields.

To my wife.

Kristi Nelson-Brown

For her encouragement and unwavering support.

ACKNOWLEDGMENTS

The author is sincerely grateful that Dr. Kareem Ahmed, Dr. Alain J. Kassab and Dr. Ranajay Ghosh have taken the time to form a committee of advisors for this research. Their expertise and guidance has been integral in the completion of this research. The author would especially like to thank the personnel of NASA's Launch Services Program. Their support made this research possible

TABLE OF CONTENTS

LIST OF FIGURES	vii
LIST OF TABLES	ix
CHAPTER 1 INTRODUCTION	1
CHAPTER 2 LITERATURE REVIEW	6
2.1 DSMC Background.....	6
2.2 DSMC Overview	14
CHAPTER 3 MOC	18
3.1 MOC Numerical Formulation.....	18
3.2 MOC Numerical Implementation RAMP.....	22
CHAPTER 4 COUPLING MOC & DSMC.....	24
CHAPTER 5 THE DSMC CODE	28
5.1 DMSC Numerical Formulation.....	28
5.2 DSMC Numerical Implementation	32

CHAPTER 6 RESULTS	37
6.1 MOC Results.....	37
6.2 DSMC Results	39
CHAPTER 7 CONCLUSION.....	42
APPENDIX A: RAMP to DAC MATLAB CODE	44
Appendix B: Supplemental Figures	51
LIST OF REFERENCES	56

LIST OF FIGURES

Figure 1 SPIFEX Data Acquisition Plate Description	2
Figure 2 Computation Domain of MOC	24
Figure 3 Location of MOC DATA Selection	25
Figure 4 MOC Solution to DSMC Boundary Condition	27
Figure 5 Iteration Convergence Criteria Study	34
Figure 6 Left: 2.21E6 & Right: 3.41E8 Mesh Count.....	35
Figure 7 Mesh Independence Study.....	36
Figure 8 RAMP Results Velocity Mag (Left) Axial Velocity (Right)	37
Figure 9 RAMP Results Temperature (Left) Nd (Right).....	38
Figure 10 Overview of Simulation	39
Figure 11 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #1	40
Figure 12 Ramp Results Magnitude of Gradient of Density	51
Figure 13 Ramp Results Density	51

Figure 14 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #2	52
Figure 15 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #3.....	52
Figure 16 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #4	53
Figure 17 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #5	53
Figure 18 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #6	54
Figure 19 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #7	54
Figure 20 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #8	55
Figure 21 Steady State Measurement vs DSMC Results (Left) Contour of Pressure DSMC Results (Right) Position #9	55

LIST OF TABLES

Table 1. Flow Regime and Knudsen Number.....	9
Table 2 DAC Boundary Condition Types	16

CHAPTER 1

INTRODUCTION

A rarefied flow plume impingement model is created in DAC and the results are compared to experimental data. The experimental data was acquired in 1994 by NASA during the Shuttle Plume Impingement Flight Experiment (SPIFEX) which flew aboard Discovery on Space Transport System flight number 64 (STS-64). SPIFEX collected plume impingement data in the vicinity of Discovery's operating Reaction Control System (RCS) thrusters while on orbit. SPIFEX was essentially a 10m long robotic arm with a data acquisition plate [1]. The 6 degrees of freedom (DOF) of the SPIFEX boom allowed for the plate to sample the flow field at many different radii, angles of attack and azimuths relative to the running RCS thruster, and recorded both transient and steady-state data. Ultimately, the objective of this research is to simulate the pressure from the RCS thrusters as it impinges on the SPIFEX plate.

As an overview, a four step process was employed to simulate the SPIFEX plume impingement for comparison with the experimental data: First, determine the 2D solution of the flow in the nozzle responsible for the impingement on the SPIFEX plate, both within the nozzle and out of the nozzle to a boundary of sufficient rarefaction. Secondly, interpolate the 2D solution to a 3D plume and provide said solution as an input into DAC. Next, optimize the DAC simulation by determining mesh independence and convergence criteria. And lastly for validation of the methods presented in this research, compare the DAC results to experimental data.

One of the very first assumptions in attempting to model this experiment was that the boom itself would have a negligible effect on the resultant plume impingement onto the SPIFEX plate, and hence only the plate has been simulated and is illustrated below in figure 1.

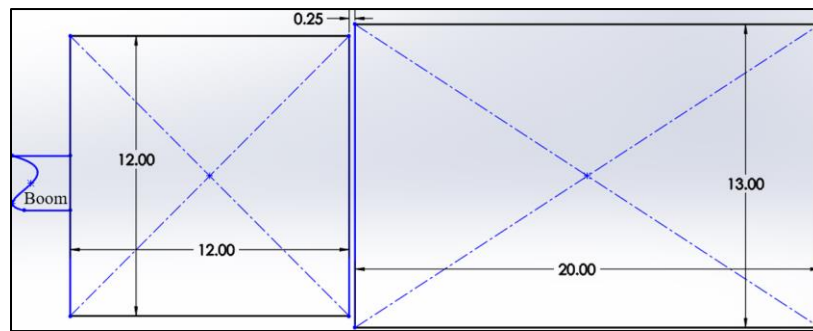


Figure 1 SPIFEX Data Acquisition Plate Description

The illustration in figure 1 is a sketch of a rendered down CAD model of the SPIFEX plate and was equipped with three separate means of data acquisition, which will be designated as measurement 1, 2 and 3. Measurement 1 was a thermistor type sensor and measurement 2 was an absolute capacitance manometer. The third pressure measurement device was a force sensing device which was normalized by the SPIFEX plate area to determine the impingement pressure. Throughout the experiment, these three pressure measuring devices acquired steady state pressure readings at 9 different positions relative to RCS thrusters, all of which were oriented normal to the impinging flow. These varying positions have been designated as Position 1, Position 2...Position n.

For this research, the continuum portion of the simulation will employ chemical kinetics, and seek to obtain chemical equilibrium. Conversely, the rarefied portion of the simulation will assume frozen chemical properties as well as assuming a single species with an average molecular weight. Lastly, grid independence and convergence criteria will be determined prior to executing any bulk computations. Grid independence will be attained in the DAC model by incrementing towards finer and finer grids while monitoring some output parameter. Once the output parameter is shown to be unchanging with incrementing grid size, the simulation results will be assumed to be independent of the grid size. Additionally, convergence will be determined by documenting the same output parameter as a function of physical time. Upon observing a constant output parameter, like pressure, the simulation will be said to have converged to a quasi-steady state solution.

Currently in DAC, there are two options for defining boundary conditions: One, define velocity, temperature, number density and gas mole fraction manually using DACs' Surface Tool Plus (STP) on all or one user defined node in the simulation. It's worth noting that the nature of DSMC theoretically allows for simply defining stagnation temperature, pressure and molecular weight of a nozzles' throat. This method would result in a number of molecules to track on the order of 10^{20} and would quickly become computationally prohibitive. Also, that method involves interacting with a Linux command line for each surface and can become cumbersome for even uniform boundary conditions on minimal or coarsely meshed surfaces. The second option is to solve the continuum portion of the computational domain using heritage tools like CFD or MOC, followed by converting the solution into a specific format readable by DAC using

some programming language. For this research, the approach will adopt the latter option and make use of RAMP [2] and MATLAB [3].

In converting continuum results into DAC, problems arise both in the interpolation scheme as well as the results which are being interpolated. For instance, in order to apply a starting condition into DAC via the boundary of a plume where the flow becomes sufficiently rarefied, the continuum portion of the source generating the plume must be solved for. The greatest mitigating factor in this method is in modeling the source, typically a nozzle, which requires it's freestream to be close to 0 absolute pressure. This freestream is required to be sufficiently rarefied, which both enables plume expansion consistent with nozzle flow expansion in a vacuum, and is a fundamental violation of most CFD programs. As a consequence, the NASA code RAMP was selected to solve the continuum portion of the simulation since RAMP uses MOC for nozzle flow solutions, and MOC essentially eliminates Navier-Stokes diffusive component and therefore eliminates the continuum requirement.

While RAMP is a well-validated MOC program, the interpolation scheme into DAC from RAMP poses some questions of validity. Specifically, and as a consequence of the nature of RAMP for nozzle flow, the solution will either be 2D or 2D axisymmetric. This solution will need to be interpolated to 3D. As a result, an assumption of negligible third-dimension variations was asserted, as well as an assumption that the solution is symmetric about the flow axis. In making these assumptions, and by following procedural steps 1-4 as previously outlined, the objective of simulating the SPIFEX plume impingement has a clear path forward. This research

will validate these assumptions and will greatly increase the working knowledge for coupling continuum solvers to DSMC.

In summary, the methodology for simulating rarefied flow plume impingement will begin with first solving the continuum portion using RAMP, interpolating to DAC boundary condition, obtaining a grid independent model, determining convergence criteria and comparing to experimental data. The following text will not only present a comprehensive derivation of the MOC and DSMC, but will also provide detailed procedural insights into implementing both associated RAMP and DAC codes, respectively. Lastly and in association with the primary objective of this research, a detailed explanation of the coupling of the two codes will be discussed along with a complete method for determining mesh independence, convergence criteria and comparison to experimental data.

CHAPTER 2 LITERATURE REVIEW

2.1 DSMC Background

The following will outline the most relative literature available regarding the process of developing a numerical model for rarefied flow fields. The purpose of this literature review is ultimately to augment and inform a thesis aimed at validating DAC against experimentally acquired rarefied flow data. Therefore, this report will contain background into rarefied flow field modeling. Also, the following will discuss some important quantitative metrics used in rarefied flow analysis and discuss their uses in detail. Next, this literature review will introduce the DSMC code DAC and provide informative explanations on its procedural use and implementation. Finally, three cases of peer reviewed journal publications that have had great success in modeling rarefied flow fields will be presented and discussed for reference.

Modeling fluids in low density hypersonic flows has been of great engineering importance since the conception of the space industry. The Navier-Stokes differential equations have been and continue to be a reliable source for predicting the behavior of fluid flows [4]. These differential equations have been and continue to be employed in a variety of commercial and open-source programs and are collectively referred to as Computational Fluid Dynamics (CFD) programs. Each individual CFD software package operates on the same principle of solving a partial differential equation at nodes in a grid while taking on some initial condition. [5]

The Naiver-Stokes equations break down though for low density fluid flows, which is evident in their derivation [6]. For brevity, implied in the derivation of the Naiver-Stokes equations is the assumption that the effects of the intermolecular collisions are outweighed by the effects of the fluids momentum and can hence be neglected [7, 8].

With rarefied gases however, simulating fluids of exceptionally low density with a continuum assumption in place violates the assumption implicated in the derivation of the Naiver-Stokes equations and hence cannot be used [9]. Alternatively, one remaining option for modeling low density fluid systems is to track each molecule and associated intermolecular collision. Of course, it's clear that a simulation tracking every particle, even in a relatively small computational domain of near zero pressure, would require an extraordinary amount of processing power.

It was ultimately Professor G.A. Bird who postulated the DSMC method for getting around the computational cost of tracking every particle in a flow field [10]. The DSMC method uses a probabilistic means to track particles while decoupling space and time [11].

Since tracking individual particles through space and time even in a low-density medium would be prohibitively computationally expensive, the DSMC method creates simulated particles which represent groups of real particles in the computational domain. This idea is the basic governing concept that makes the numerical analysis of rarefied gasses possible and was given the term F_{num} by Professor Bird [12].

When using the DSMC method to model a rarefied flow field, F_{num} given in equation 1 is a useful parameter and is the ratio of real to simulated particles. A typical DSMC simulation will commonly operate on $\sim 10^{18}$ real particles for every simulated particle [13]. With the number of molecules to keep track of now reduced by several orders of magnitude, the DSMC method can proceed to track the simulated particles.

$$F_{num} = \frac{\text{real particles}}{\text{simulated particles}} \quad (1)$$

Another great insight in the development of the DSMC method is in its ability to decouple space and time in the tracking of particles. This is done by optimizing f_{num} with grid size and time step just such that proceeding a particle collision at some point P_1 the new position of the particle at P_2 can be found by the product of the velocity vector and the time step, equation 2. Or in other words, a particle entering a mesh cell will remain unperturbed at least through the cell space.

$$P_2 = P_1 + \vec{V}\Delta t \quad (2)$$

Another useful parameter in rarefied gas dynamics is the ratio of mean free path to characteristic length. Fluids are said to be rarefied when the ratio of the mean free path (λ) to the representative length scale becomes sufficiently large. This ratio is known as Knudsen number [14], hence equation 3.

$$Kn = \frac{\lambda}{L} \quad (3)$$

Where the mean free path (λ) can be thought of as the average distance a molecule travels before being perturbed from its path, and the representative physical length scale would be dependent on the geometry within the flow domain; such as throat diameter for nozzle flow or planar area surface length in the flow direction for flow over an airfoil [15]. Table 1 below illustrates the ranges of rarefaction of gases based on Knudsen number [16].

Table 1. Flow Regime and Knudsen Number

Knudsen Number	Range of Kn	Flow regime	Solver
kn	> 1	<i>Free molecular flow</i>	DSMC
kn	$0.1 > 1$	<i>Transitional</i>	DSMC
kn	$0.1 > 0.01$	<i>Slip flow</i>	DSMC & CFD

When describing the intermolecular collisions between simulated particles, a molecular model must be defined and there are several widely accepted molecular models to choose from. They differ in large part by accounting for or neglecting the large range attractive forces between molecules [17]. There are many molecular models described in use today, but the most common models are the variable hard sphere (VHS) and hard sphere (HS) [18]: The text goes on to explain that these VHS and HS molecular models are common because they disregard high range attractive forces between molecules. Additionally, the viscosity and mean free path from the text are represented as a function of temperature and given by equation 4 and 5, respectively. Where $\alpha = \omega - 1/2$, and the HS molecular model is simply a special case of the VHS model where $\alpha = 0$.

$$\mu = \mu_{ref} \left(\frac{T}{T_{ref}} \right)^\omega \quad (4)$$

$$\lambda = \left[\sqrt{2} \pi n d_{ref}^2 \left(\frac{T_{ref}}{T} \right)^\alpha \right]^{-1} \quad (5)$$

The most common molecular model that accounts for the intermolecular force is the Lennard Jones potential (LJ) model [19]. The LJ potential model uses empirically derived correlations of Bird *et.al* [20] to describe the potential energy, U given in equation 6, in addition

to the viscosity term as a function of temperature given in equation 7, where V and $W^{(2)}$ are functions of $kT\varepsilon^{-1}$ tabulated in [21, 22] and $\varepsilon = 0.281eV$ [23].

$$U = 4\varepsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right] \quad (6)$$

$$\mu = \frac{5V}{8\sigma^2 W^{(2)}} \sqrt{\frac{mkT}{4\pi}} \quad (7)$$

With either of the molecular models employed, collisions and subsequent changes in momentum and position of the simulated particles with neighboring simulated particles are then calculated probabilistically via Boltzmann equation 8, which was formulated by Ludwig Boltzmann in 1872 for his method of describing low density gases from a kinematics perspective [24].

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \mathbf{F} \cdot \nabla_{\mathbf{v}} f = \left[\frac{\partial f}{\partial t} \right]_{coll} \quad (8)$$

In addition to the molecular model governing the behavior of particles during collision in general, collision types are further divided into two groups: Particle to surface and particle to

particle. Collisions of simulated particles with a solid surfaces are calculated directly. The nature of such collisions have a direct impact on the prediction of the boundary layer as well as surface-to-simulated particle kinetic energy transfer and subsequent surface heating and or change in flow energy [25]. In addition to changes in kinetic energy and boundary layer development, surface collision models directly affect the design of a vehicles' thermal protective system [26] and nozzle corrosion [27]. Collisions between particles however are calculated probabilistically as an average of the particles properties interacting with the average of some neighboring particles properties.

Number density is another useful parameter in DSMC analysis of rarefied gases because it provides insight onto the degree of rarefaction, or the diluteness of a fluid without considering the representative physical length scale [28] and has units of particles per volume. Hence equation 9:

$$n_d = \frac{\rho}{M_w} A_v \left[\frac{\text{molecules}}{m^3} \right] \quad (9)$$

In addition to the molecular model type, the surface catalysis efficiency is an important parameter in DSMC as it defines the likelihood of molecular recombination after a collision. Commonly, it is a percentage defined by DAC users and is derived by empirical data. The surface catalysis efficiency can potentially yield erroneous solutions if not properly defined.

There are real world consequences in hypersonic rarefied flows that occur as a direct result of the catalysis efficiency of the surface. For example, strong shockwaves are generated about the bow of a hypersonic aircraft. As a result, the fluid in the region between the strong shockwave and the surface of the aircraft experience a rapid rise in temperature and subsequently excite the vibrational degrees of freedom of the gas resulting in poly/di-atomic molecules dissociating into highly reactive atomic gases. If this were to occur at high altitudes where the ambient air is of low enough density, the newly dissociated, reactive gas species would diffuse through the boundary layer and possess the potential to chemically react with the vehicles' thermal protective layer [29]. In fact, the same publication cites that 30% of the heat load on reentry vehicles can be attributed to surface catalysis efficiency.

Lastly, the accommodation coefficient in DAC is a flag to be set to either diffuse or specular and has been shown to greatly affect the overall result of the simulation [30]. A noteworthy takeaway from existing literature regarding the reaction rates in a finite-rate surface catalysis model [31] was that in varying the accommodation coefficients from 0.5→1, the accommodation coefficient of 0.9 produced results in the simulation which were most agreeable to their empirical data gathered in the same study.

2.2 DSMC Overview

There are several DSMC codes available in both open-source; i.e. OpenFOAM, SPARTA and PIClas, as well as commercially available code; i.e. codes by Professor Bird like DS1V, DS2V and DS3V to name a few, but this research will focus on making use of Direct Simulation Monte Carlo Analysis Code 1997 or DAC97 officially, or DAC for short.

DAC is a rarefied gas simulation solver created by NASA's Johnson Space Flight Center [32]. Relative to other DSMC codes, DAC sets itself apart from the rest of the DSMC codes available today mostly by its preprocessor. DAC's preprocessor takes a lot of the guess work out of determining the optimal f_{num} and time step as a function of user defined mesh size and molecular flow properties. DAC makes use of a dual level Cartesian grid which makes the computational domain rectangular and is aligned with the coordinate axis. The surface geometry of the test article is then embedded within computational domain and is represented by a collection of unstructured tetrahedral cells. Cartesian divisions of the computational domain are specified by the user and make up the Level-I refinement and fill the rectangular bounding box. The second level grid refinement surrounds the embedded surface geometry. It is with the level II cell refinement that DAC has ability to restart a simulation with an adapted mesh according to a combination of the previous solution and user defined changes.

In DAC, it is up to the user to develop a text writing program to create the geometry input file. Fortunately, the DAC User's manual provides instructions on how to create such a code. The format of the DAC geometry input file follows the "right-hand-rule" for outward normal

definitions in association with triangle connectivity order. For instance, consider moving clockwise from point one to 4 on a square which is bisected by points 1 and 3, making two triangles. A point on the surface would be defined by equation 10, and the proper from of triangle connectivity would then be, if out of the page was the normal direction, expressed as in equation 11.

$$P_n = f(x_n, y_n, z_n) \quad (10)$$

$$\begin{bmatrix} Tri_1 \\ Tri_2 \end{bmatrix} = \begin{bmatrix} P_1 & P_2 & P_3 \\ P_2 & P_4 & P_3 \end{bmatrix} \quad (11)$$

When writing data to DAC geometry input files, there are several parameters to define in addition to the X,Y and Z points with their associated triangle connectivity. The first entry of this file, on row 1, from left to right is the number of nodes, number of triangles and number of gas species in the simulation. Following this, the XYZ geometric points are listed in arbitrary order, followed by the triangle connectivity ordered sets, keeping in mind the outward normal identifying convention.

The next group in the DAC geometry input file is the group number and boundary condition type, which are ordered in a sequence of rows such that the first row pertains to the first triangle listed in the 3rd entry. Group numbers are used to associate similar group types with each other, and 5 boundary condition types [33], which are described below in table 2.

Table 2 DAC Boundary Condition Types

Boundary Type	Description
Solid Wall	No molecules produced, none pass through
Outgassing Wall	No molecules pass through, molecules generated on flow side
In Boundary	Molecules disappear upon contact, molecules created on flow side
Out Boundary	Molecules disappear upon contact, no molecules generated
Freestream	Molecules disappear upon contact, no molecules generated

The 5th row of the DAC input file contains one of the following for each vertex in the computational domain: The accommodation coefficient, the surface catalysis efficiency, wall temperature, gas temperature, number density of the flow, three components of velocity and mole fraction.

In short, DAC is an incredibly capable DSMC code and is an invaluable DSMC tool both for analysis and design. For example aerial vehicles traveling through fluid mediums typically make use of moveable aerodynamic surfaces to control the attitude of the vehicle. However, in the case of a vehicle traveling through low density or otherwise highly rarefied mediums, the vehicle relies on the use of reaction control system (RCS) jets to control their position and attitude [34]. Understanding the performance and behavior of these thrusters is of crucial

importance in the design and operation of spacecraft meant to operate in rarefied flow fields and cannot be predicted without the use of DSMC solvers like DAC.

CHAPTER 3

MOC

3.1 MOC Numerical Formulation

To begin with, equation 12 represents the velocity potential of a steady, 2D, irrotational flowfield in the form a partial differential equation [35].

$$0 = \left(1 - \frac{u^2}{a^2}\right) \frac{\partial^2 \phi}{\partial x^2} + \left(1 - \frac{v^2}{a^2}\right) \frac{\partial^2 \phi}{\partial y^2} - \frac{2uv}{a^2} \frac{\partial^2 \phi}{\partial x \partial y} \quad (12)$$

Then, for some flow field variable f , the change in f , df , is found to be related to small changes in 2d space, namely dx and dy , where $u = \partial\phi/\partial x$, and $v = \partial\phi/\partial y$ [36]. Therefore, if in letting the flow field variable f be u and v in accordance with the potential velocity definition then equation 12 can be expressed as equations 13 and 14, respectively.

$$du = \frac{\partial^2 \phi}{\partial x^2} dx + \frac{\partial^2 \phi}{\partial x \partial y} dy \quad (13)$$

$$dv = \frac{\partial^2 \phi}{\partial y^2} dy + \frac{\partial^2 \phi}{\partial x \partial y} dx \quad (14)$$

Following this, equations 12, 13 and 14 all involve second derivatives of ϕ and can be solved using Cramer's rule [37]. Hence, equation 15.

$$\frac{\partial^2 \phi}{\partial x \partial y} = \frac{(1-u^2/a^2)*du dy + (1-v^2/a^2)dv dx}{(1-u^2/a^2)dy^2 + (2uv/a^2)dxdy + (1-v^2/a^2)(dx)^2} \quad (15)$$

For any chosen direction, (dx and dy), equation 15 should provide a solution for $\partial^2 \phi / \partial x \partial y$. There exist however, particular directions for which $\partial^2 \phi / \partial x \partial y$ will be indeterminate. Or alternatively phrased, the denominator of equation 15 is equal to zero, equation 16.

$$0 = (1 - u^2/a^2)dy^2 + (2uv/a^2)dxdy + (1 - v^2/a^2)(dx)^2 \quad (16)$$

Hence, from solving equation 16 for dy/dx and in using the subscript '*indt*' to indicate where the differentials along the slope of the characteristic line are indeterminate, we arrive at equation 17.

$$\left(\frac{dy}{dx}\right)_{indet} = \frac{\frac{uv}{a^2} \pm \sqrt{\frac{u^2+v^2}{a} - 1}}{1 - \frac{u^2}{a^2}} \quad (17)$$

Finally, u and v are expressed in Cartesian coordinates where $u = V \cos(\theta)$ and $v = V \sin(\theta)$. Additionally, in making use of the fact that $V^2 = u^2 + v^2$ and $M = V/a$, then equation 17 can be represented as equation 18. Furthermore, by making use of the definition of local Mach angle [38], $\sqrt{M^2 - 1} = 1/\tan(\alpha)$, we arrive at equation 19, which can be manipulated algebraically into equation 20.

$$\left(\frac{dy}{dx}\right)_{indet} = \frac{-\cos\theta \sin\theta \pm \sqrt{M^2 - 1}}{1 - M^2 \cos^2\theta} \quad (18)$$

$$\left(\frac{dy}{dx}\right)_{indet} = \frac{\cos\theta \sin\theta \pm \cos\alpha \sin(\alpha)}{\sin^2\alpha - \cos^2\alpha} \quad (19)$$

$$\left(\frac{dy}{dx}\right)_{indet} = \tan(\theta \pm \alpha) \quad (20)$$

Next, from equation 20, the argument of tangent provides two answers for lines in the flow wherein which the derivatives are indeterminate, or so called *characteristic lines*, and make a Mach angle with the streamline. Additionally, the net angle between the characteristic lines and

the axis is equal to the sum of the angles made by the streamlines themselves with the flow axis and hence indicates that the Mach waves *are* the characteristic lines.

Next, in consideration of the governing equations for MOC, we consider an instance where equation 15 has a zero numerator. Hence, the numerator of equation 15 is set equal to zero, solved for dv/du and had the aforementioned velocity relations used in equation 12&13 substituted into it and finally further manipulated algebraically to become equation 21, which is the result of transforming equation 15 from a partial differential equation into an ordinary differential equation. Next, integrating equation 21 becomes the Prandtl-Myer function, equation 22. Finally, equation 23, which contains the constant of integration from equation 21, is the featured algebraic relationship required for the numerical scheme for characteristic lines in a 2D axi-symmetric nozzle [39].

$$d\theta = \pm \sqrt{M^2 - 1} \frac{dV}{V} \quad (21)$$

$$v(M) = \int_0^M \frac{\sqrt{M^2 - 1}}{[1 + (\gamma - 1)M^2/2]} \frac{dM}{M} \quad (22)$$

$$\theta \pm v = \text{constant} \quad (23)$$

3.2 MOC Numerical Implementation RAMP

As a general overview, the NASA code RAMP operates over three intervals, each of which are called via command prompt which will request an input file in text data form. For this research, a relatively simple batch files was created to allow for a single key stroke execution of a RAMP case which included post processing. This is not necessary for RAMP operation, but proved to be a valuable use of time for this case.

The batch file would first call the built in executable for formulating the Chemical Equilibrium Calculation (CES), followed by executing the MOC using the stagnation properties found from CES with the RAMP executable. Following this, RAMP relays the initial flow properties in the nozzle to the boundary layer calculation tool BLIMPJ, after which RAMP is called again using the boundary layer as a quasi-wall. This cycling between BLIMPJ and RAMP can potentially be executed multiple times, but 2-3 iterations have been found to be sufficient. Lastly the post processing tool is called and a solution file is written to the working directory with user defined flow parameters that can be parsed programmatically or read as TecPlot [40] or text data files.

RAMP's CEC program requires only a few input parameters which must be properly configured in a text documents in order for CEC to read it. For instance, the reactants are defined as a molecular ratio and are defined in their respective roles as either fuel or oxidizer on the first two lines. Secondly, the throat stagnation pressure is defined along with the range and increment of fuel to oxygen mixture ratios, typically ranging from 0.8 to 2.2 on the next two lines. Variable

units are also specified in CEC such as “ENG” or “SI” in the case of the latest CEC version. This instance highlights the importance in creating a batch file for this operation, although CEC, RAMP and BLIMPJ depend on each other, each program will clear the newly created text files from the directory. Hence, it becomes useful to automate the copying and storing of important interim files that may be worth keeping.

Following this, the RAMP code is called using the stagnation properties calculated through CEC. This code will require the reading of a text document containing specially formatted XY points of the nozzle, mesh size, and computational domain, 2D type, (axisymmetric or true 2D). This is also where a specific flag can be set to define the computational domain to be outside of the nozzle up and out to boundaries like Bird Breakout parameter or Knudsen number.

The boundary layer calculation does not require any specially formatted text documents from the user. All of the information for the boundary layer calculations (BLIMPJ) are taken from CEC and RAMP. Following this, the process can be repeated with a new RAMP calculation which takes into account the boundary layer as a pseudo nozzle wall. Lastly, the post processing executable is called and writes out the solution files containing user defined flow parameters such as Mach, Knudsen number or Bird number, to name a few.

CHAPTER 4

COUPLING MOC & DSMC

The process for coupling RAMP results to DSMC requires an interim code to both parse the output from RAMP as well as properly translating the flow properties from 2D-axisymmetric to 3D. For this task, it was decided to develop a MATLAB script that could seamlessly accomplish not only the parsing and the translation, but also write a completed and properly formatted text file for input into DSMC. Hence, the MATLAB script located in appendix A was created.

To begin with, the solution from MOC was parsed using MATLAB and had each of its flow properties stored at each X and Y nodal location illustrated in figure 2 below.

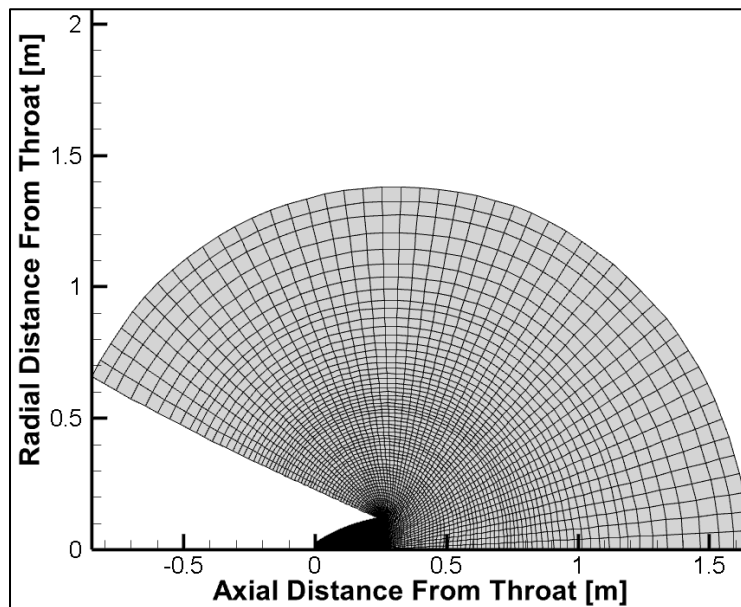


Figure 2 Computation Domain of MOC

Following this, the data obtained from the MOC solution was parsed to take the outer most boundary, as this is the furthest away from the throat and is assumed to be most rarefied. The figure below illustrates the boundary upon which data is selected for rotation. In order to ease computational cost, the boundary in the image below did not follow the data contour at approximately the position of $[-0.5 \ 1]$. Rather, the data extraction was halted at that point and terminated at the point $[-1.0 \ 0]$. The reason for this being an acceptable parsing of data is due to the nature of the plume boundary condition type, which is defined such that particles striking the boundary will disappear, and hence any fluid activity aft of the nozzle ($x < 0$) would not contribute to the forward plume impingement.

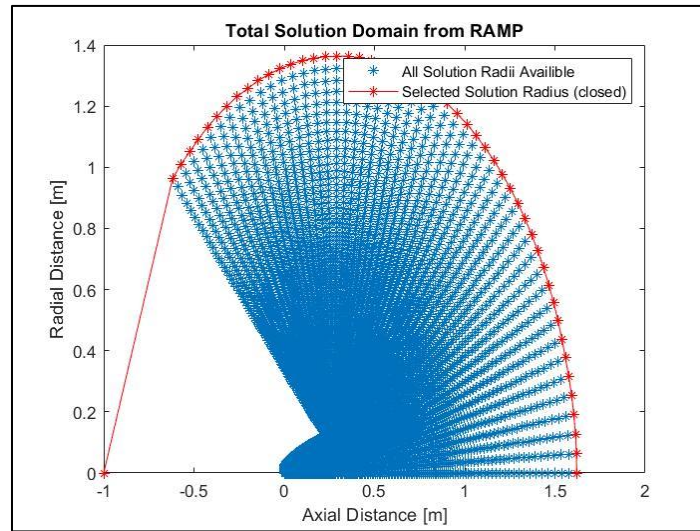


Figure 3 Location of MOC DATA Selection

Next, the assumption of negligible “Z” directional solution properties was asserted, i.e.

$V_k = 0, \omega_{radial} = 0, \omega_{axial} = 0$, and hence the nodal positions and velocity were translated

according to equation 24 and 25, while the other properties like number density were simply transcribed according to their axial position on the plume, equation 26.

$$\begin{matrix} X_{translation} & X_{2D} \\ Y_{translation} & = Y_{2D} \cos(\theta_n) \\ Z_{translation} & Y_{2D} \sin(\theta_n) \end{matrix} \quad (24)$$

$$\begin{matrix} U_{translation} & U_{2D} \\ V_{translation} & = Y_{2D} \cos(\theta_n) \\ W_{translation} & Y_{2D} \sin(\theta_n) \end{matrix} \quad (25)$$

$$N_{d\ i,j} = \rho_i \frac{A_v}{M_{w_i}} \quad (26)$$

Finally, the 2D axi-symmetric data is parsed and translated into a 3D solution, which is then written to a text file in accordance with the format illustrated in the DAC portion of the literature review. To solidify the methodology presented in this section, the “surf” function in MATLAB [41] has been used to plot the revolved 2D-axisymmetric MOC solution into a 3D surface with its representative nodal boundary conditions. Hence, figure 4 depicts a contour of axial velocity imposed on a bulb shape which is a result of rotating the axisymmetric RAMP solution.

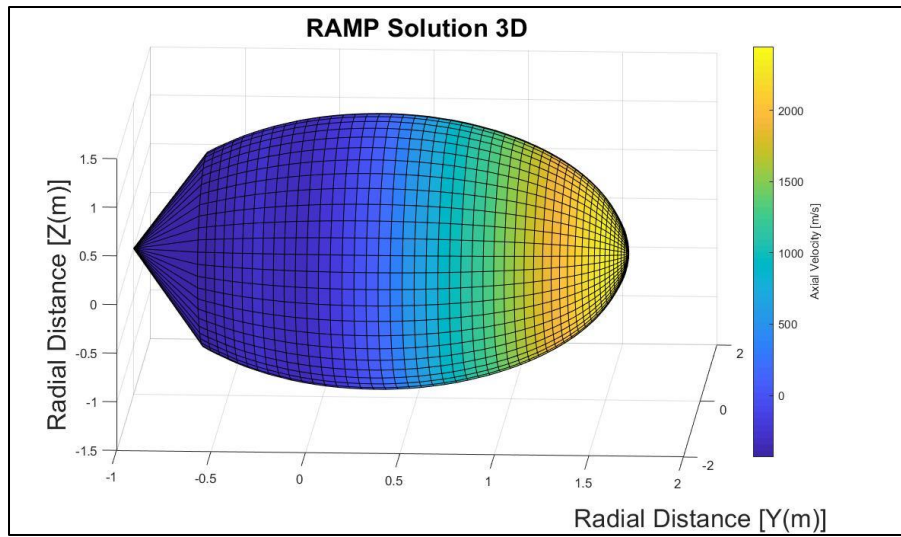


Figure 4 MOC Solution to DSMC Boundary Condition

CHAPTER 5 THE DSMC CODE

5.1 DMSC Numerical Formulation

The Monte Carlo method is a type of computational algorithm that rely on repeated sampling to obtain numerical results. In the case of rarefied gas dynamics, the method is applied to the Boltzmann equation in order to describe the statistical behavior of thermodynamic systems. Boltzmann along with Maxwell recognized early on in their study of rarefied gas dynamics that the practical calculation of the behavior of gas particles would be computationally impractical as it would require a number of individual variables for each particle, which number on the order of 10^{20} [42]. In lieu of expressing the exact dynamics of N particles, the probability distribution function $P^{(1)}(x, \vec{V}, t)$ was proposed by Boltzmann.[43]

Therefore, the following will describe the evolution and assumptions in solving for the probability $P^{(1)}$ for a particle being at position x and time t having velocity $\vec{V}$, where $P^{(1)}$ satisfies equation 27.

$$\frac{\partial P^{(1)}}{\partial t} + \vec{V}_1 * \frac{\partial P^{(1)}}{\partial x_1} = G - L \quad (27)$$

To begin with, the evolution of the probability density of $P^{(1)}$ requires a new variable $P^{(2)}$ which would then be a function of the $P^{(1)}$ and its new time and velocity, $P^{(2)}(x_1, \vec{V}_1, t, x_2, \vec{V}_2)$. Or, put another way the new probability distribution as a function of the old probability distribution can be expressed in equation 28, below. Where equation 28 represents the assumption of *molecular chaos* [44], which implies that the collision of two particles are selected at random, then its probability distribution is a product of finding the two particles with different characteristics.

$$P^{(2)}(x_1, \vec{V}_1, t, x_2, \vec{V}_2) = P^{(1)}(x_1, \vec{V}_1, t)P^{(1)}(x_2, \vec{V}_2, t) \quad (28)$$

Next, for a cylindrical volume with a base area expressed as $dS = \sigma^2 d\mathbf{n}$, equation 28 can be expressed in the form illustrated below in equation 29, Where $(\vec{V}_2 - \vec{V}_1)\mathbf{n} < 0$

$$P^{(2)}(x_1, \vec{V}_1, x_1 + \sigma\mathbf{n}, x_2, t) = P^{(1)}(x_1, \vec{V}_1, t)P^{(1)}(x_2, \vec{V}_2, t) \quad (29)$$

Next, in order to determine the number of collisions of some hypothetical particle with another, at any point on the spherical surface of the particle, it is necessary to integrate over the

surface for all possibilities of particle properties to obtain an expression for both the gain (G from equation 27) and the losses (L from equation 27), to obtain equation 30 and 31 below.

$$L = (N - 1)\sigma^2 \int_{R^3} \int_{B^-} P^{(2)}(x_1, \vec{V}_1, x_1 + \sigma \mathbf{n}, x_2, \vec{V}_2, t) |(\vec{V}_2 - \vec{V}_1) \mathbf{n}| d\vec{V}_2 d\mathbf{n} \quad (30)$$

$$G = (N - 1)\sigma^2 \int_{R^3} \int_{B^+} P^{(2)}(x_1, \vec{V}_1, x_1 + \sigma \mathbf{n}, x_2, \vec{V}_2, t) |(\vec{V}_2 - \vec{V}_1) \mathbf{n}| d\vec{V}_2 d\mathbf{n} \quad (31)$$

Finally, by inserting equations 30 and 31 into equation 27, we have an expression for the Boltzmann equation, equation 32.

$$\begin{aligned} \frac{\partial P^{(1)}}{\partial t} + \vec{V}_1 * \frac{\partial P^{(1)}}{\partial X_1} = N\sigma^2 \int_{R^3} \int_{B^-} [P^{(1)}(x_1, \vec{V}'_1, t) P^{(1)}(x_1, \vec{V}'_2, t)] - \\ P^{(1)}(x_1, \vec{V}_1, t) P^{(1)}(x_1, \vec{V}_2, t) |(\vec{V}_2 - \vec{V}_1) \mathbf{n}| d\vec{V}_2 d\mathbf{n} \end{aligned} \quad (32)$$

The Boltzmann equation represented in equation 32 has been expressed with properties of a few assumptions, the most significant of all being the assumption of *molecular chaos*, and

hence requires some discussion. Molecular chaos is a product of randomness, and hence equation 27 cannot be argued as a property of dynamics [45]. However, it is clear that molecular chaos is present at the initiation of molecular disturbances, but such randomness must depreciate over time. If the molecular randomness was forever extant everywhere, the right hand side of equation 27 would be zero and hence no effect would result in the collisions with $P^{(1)}$ over time.

The numerical remedy for this overarching problem is in the capacity to impose molecular randomness for particles that are about to collide, rather than imposing the molecular chaos assumption to the entire domain, for the entire simulation. To that end, the numerical implementation of the Boltzmann equation represented in equation 32 has been modified with “nearest-neighbor” molecular models within the computational domain. Nearest neighbor molecular models have been presented in the Literature Review chapter and are a common remedy for the molecular chaos assumption.

5.2 DSMC Numerical Implementation

The first step required for DSMC numerical implementation is to execute DAC's STP executable [42], which is one of the great functions of this particular DSMC code. For instance, provided that a properly formatted geometry file is fed to STP in accordance with the steps and format discussed in the introduction of this research, STP can perform several valuable functions. STP has the ability to perform geometric input file diagnostics, boundary condition assignment, geometric transformations and much more.

The next step in implementing this DSMC program is to run DAC's highly valuable PREDAC function [46]. Generally speaking, with few exceptions, a successful execution of STP will populate a properly configured version of the geometry file and perform an initialization of the flow field. PREDAC uses this initialization of the flow field to set optimal values for particle to cell ratio by evaluating the number density of the boundary condition in conjunction with the velocity magnitude of the particles. PREDAC does this by increasing or decreasing the FNUM and the time step. This is a highly valuable function as the premise of this DSMC solver is that a particle entering a computational cell can be translated to the other side of the cell by way of its new position being a sum of its original position plus the product of its velocity vector and global time step. Logically, this requires the fluid particle to be unperturbed during inter-cell travel and hence requires an appropriate FNUM and time step.

Lastly, the DAC input file was generated from the PREDAC function and contains the appropriate global time step and FNUM along with all other user defined functions asserted in

the PREDAC file. This DAC input file comes from PREDAC nearly ready to be fed into DAC for computation to commence, it only requires the user to define the number of steps to steady state and the number of time steps to execute. When executing DAC with the DAC input file of simulation instructions and boundary conditions, the user then has the option to compute with parallel or serial processors. The parallel version of DAC is called DDAC and uses a Message Passing Interface (MPI) and can be called from a Linux window with the syntax `MPRUN -np # ddac >ddac.dat`. The MPI is an exceptionally efficient portable message passing standard [43] and it is the fundamental facet of DAC that allows its software such high performance when evaluating rarefied flows.

Upon executing DAC with some arbitrary mesh density, the program was allowed to run while monitoring some flow field property. The chosen property was arbitrary, but in for this case, pressure was monitored on the SPIFEX plate. The reason for this pre-run analysis was to determine the amount of physical computational time required for the simulation to reach a quasi-steady state. The amount of physical computational time required to converge to this pseudo-steady state was found to be approximately ~ 240 [ms], which was found to be in good agreement with experimental data. A plot of the impingement pressure on the SPIFEX plate vs physical computational time is illustrated below, in figure 5.

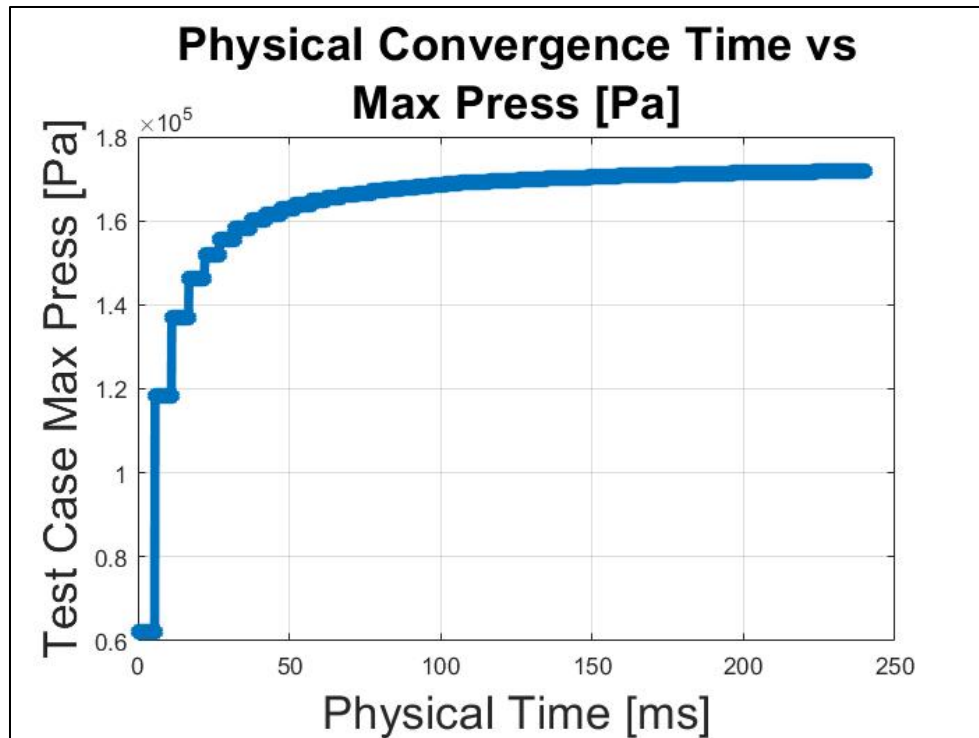


Figure 5 Iteration Convergence Criteria Study

With the required amount of physical computational time determined in figure 5, it was now necessary to determine a proper mesh size of the computational domain. For this effort, the same flow parameter, (pressure impinging on the SPIFEX plate) was recorded, this time at the end of ~ 240 [ms] physical computational time, and plotted against mesh density.

The first iteration began at an arbitrarily sized, yet exceptionally course mesh to establish a baseline, and the mesh was increased from there. During this grid independence study, it was necessary to hold all other variables constant as to not mistake a change in output parameter with some change in mesh size.

Recall that the Shuttle Discovery itself was not simulated in DSMC, and that it is only there for visual aid. Given that, figure 6 below depicts a base line course mesh containing $2.21\text{E}6$ cells. The processing time for this simulation was exceptionally fast. However, the plot in figure 7 indicates that the simulation is not grid independent. Also, the left hand side of figure 6 reveals a poorly structured, blotchy mesh. Figure 6 on the right hand side illustrates the same simulation with two orders of magnitude more cells, $\sim 3.41\text{E}8$. Also, from evaluating figure 7, it is clear that this is the order of magnitude required for mesh independence.

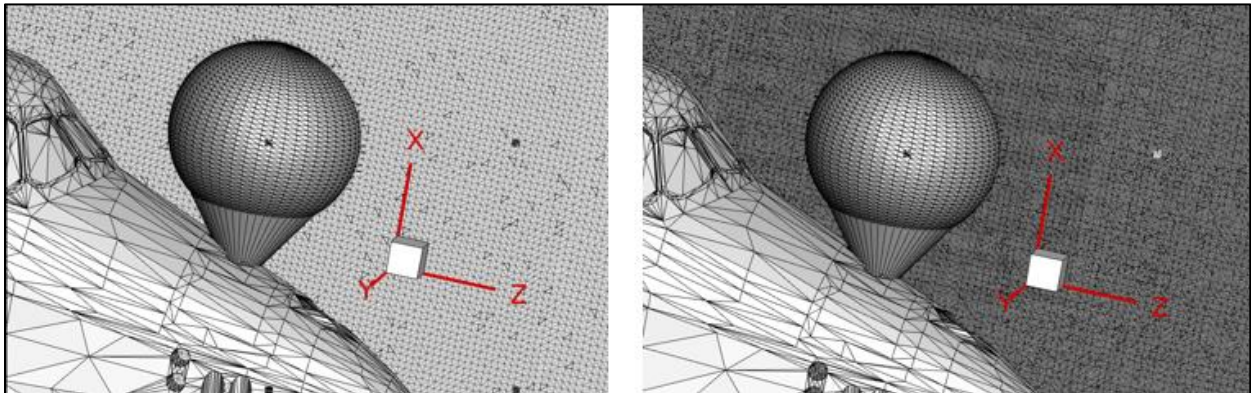


Figure 6 Left: $2.21\text{E}6$ & Right: $3.41\text{E}8$ Mesh Count

Lastly, a plot of the change in impingement pressure on the SPIFEX plate while increasing cell count was monitored using a constant sampling & plotting MATLAB routine. The results of which are plotted below in figure 7. The plot indicates that the resultant pressure on the SPIFEX plate converged asymptotically at around $2.75\text{E}8$ cells per computational domain.

Which in the end, made it clear that the all simulations (#1-#9) should be evaluated with cells numbered on the order of $\sim 3E8$ per computational domain, and should run for no less then ~ 240 [ms].

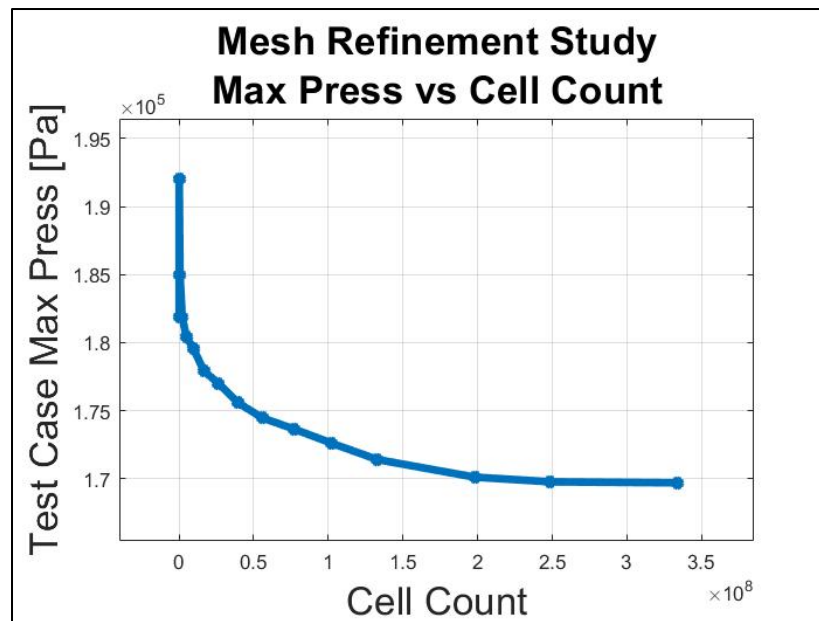


Figure 7 Mesh Independence Study

CHAPTER 6 RESULTS

6.1 MOC Results

To begin with, figure 8 below shows scalar contours of the flow velocity results of the RAMP simulation for flow through and outside of the nozzle of the RCS thruster with a 2D-axisymmetrical assumption implied. The contour plot on the left of figure 8 depicts the velocity magnitude. From the velocity magnitude plot, there exists stark gradients propagating away from the nozzle exit plane. These sharp gradients in velocity magnitude are Mach lines emanating away from the nozzle exit and indicate the presence of shock waves. Additionally illustrated in figure 8 is a contour of axial velocity, (Right). This plot provides greater insight into the stark changes in velocity magnitude and reveal that a large portion of the change in velocity is along the axial direction.

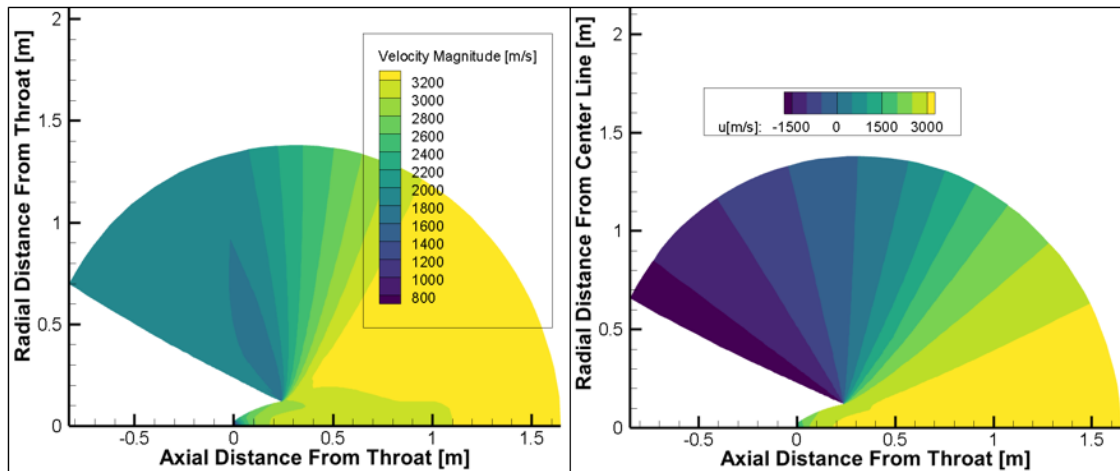


Figure 8 RAMP Results Velocity Mag (Left) Axial Velocity (Right)

Next, figure 9 depicts scalar contours of both the temperature distribution as well as the number density in and out of the nozzle. The plot on the left of figure 9 depicts temperature, and there are sharp gradients near the exit plane of the nozzle. An interesting observation of the contour of temperature is the rapid diffusion of temperature from the throat of the nozzle relative to the outer bands of the computational domain.

Additionally, the right side of figure 9 is a plot of number density distribution in and out of the nozzle wall boundary. There are sharp gradients of number density and appear to propagate in concert with the axial flow velocity contour. Figures 12 and 13 in Appendix B illustrate additional contour plots of the RAMP solution for both the magnitude of the density gradient and the density distribution for the same computational domain as figures 8 and 9.

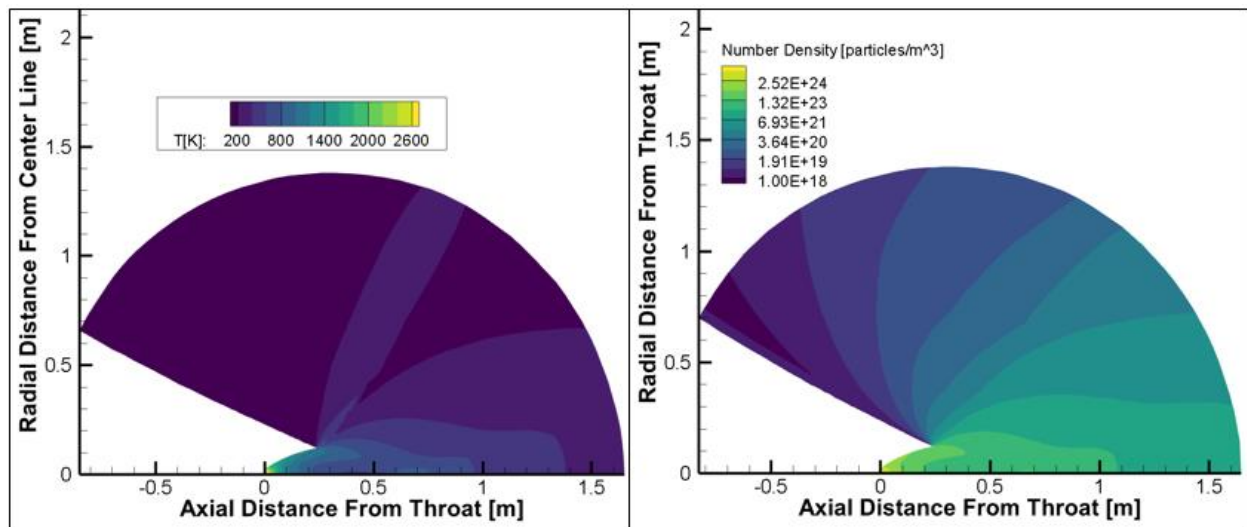


Figure 9 RAMP Results Temperature (Left) Nd (Right)

6.2 DSMC Results

The figure below stands as an overview of the results of running the DSMC simulation. The figure is a scalar contour of X-direction velocity and contains a “cut-plane” in order to evaluate the flow field defined by the normal coordinates, $[0, 1, 0]$ at position $[0,0,0]$. Also, the nose of the shuttle contains the continuum portion of the flow field which was solved for using RAMP and imposed as a starting condition in the DSMC simulation at the location where the RCS thruster was for Discovery.

Lastly in the figure, the black arrow is pointing at the simulated SPIFEX data acquisition plate, and is clearly being impinged upon. Since the rotation and translation of the MOC solution and subsequent integration into the DSMC solver was all done programmatically via MATLAB, figure10 serves as an assurance that the MATLAB program has executed without error and has performed as intended.

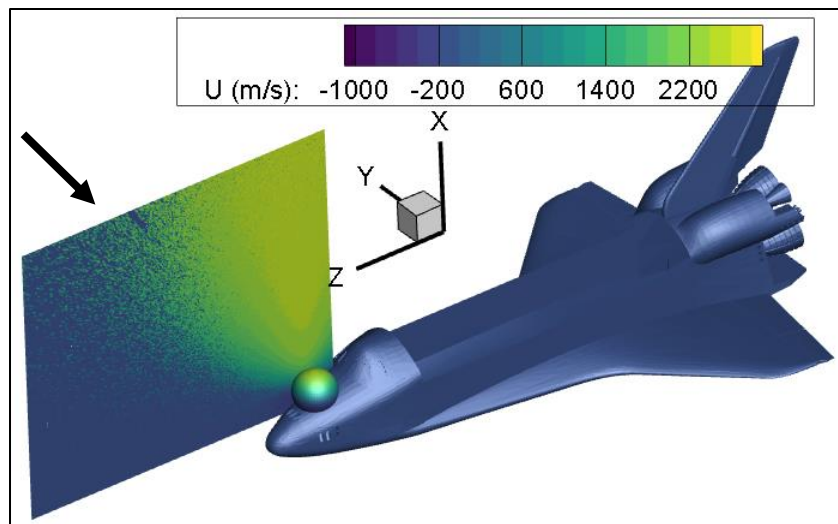


Figure 10 Overview of Simulation

Next, the results of the DSMC simulation for position 1 of the SPIFEX data acquisition plate is illustrated below. Figure 11 below illustrates the steady state experimental pressure plotted against the simulated data from $t=0$ to $t=240$ [ms]. Measurements 1, 2 and 3 are plotted in red, orange and purple, respectfully and an average of the three measurements is also plotted in green.

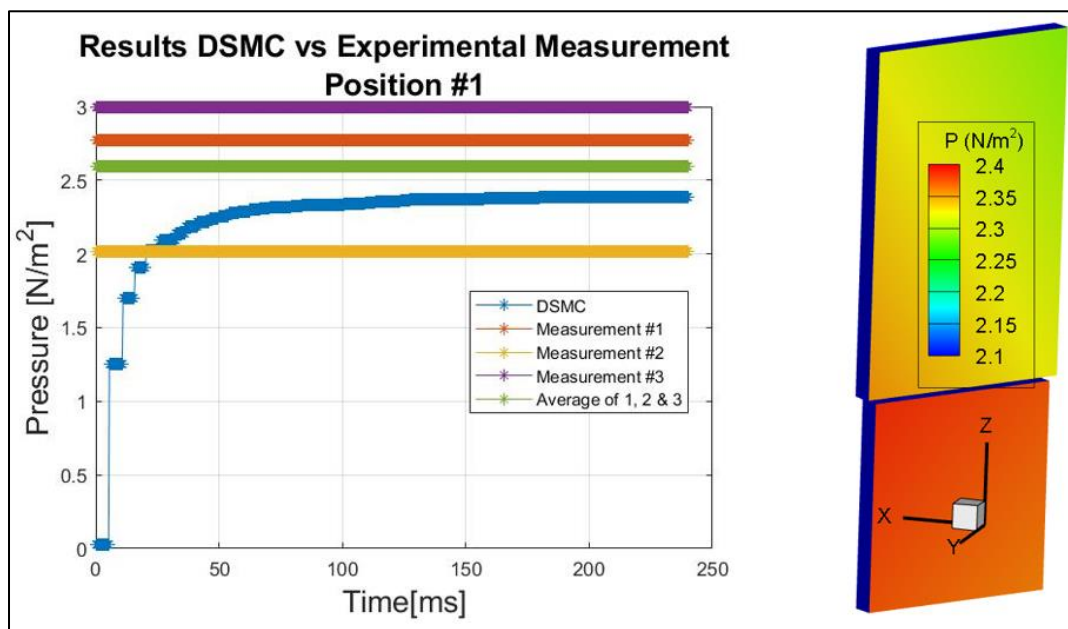


Figure 11 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #1

Additionally, figure 11, contains a scalar contour of the pressure induced on the SPIFEX plate from the simulation of the RCS thruster. One interesting take away from the scalar contour is that the pressure is not uniformly distributed. Although the plate was oriented normal to the RCS thruster, the difference in size of the plate has resulted in non-uniform pressure. Figures 14-

21 in Appendix B show the results of positions 2-9 in the same fashion, each with similar agreement to experimental data.

CHAPTER 7

CONCLUSION

This text has presented a detailed review of research into validating a coupled MOC and DSMC method for simulating plume impingement in a rarefied flow field. The data that was used to ground these simulations was acquired during the NASA mission STS-64, where a 6 DOF articulating arm on board NASA's Discovery Shuttle positioned a data acquisition plate with three different pressure sensing devices, at 9 different locations in the RCS plume relative to Discovery's RCS thrusters. Additionally, the numerical formulation for MOC and DSMC has been discussed in detail along with a procedural description of the implementation of their associated solvers, RAMP and DAC. Following this, the coupling of the solution from RAMP as a starting condition to DAC was explained, and the MATLAB code for carrying out such an interpolation has been placed in Appendix A.

Furthermore, the results of the RAMP solution were presented and discussed by illustrating contours of velocity, temperature and number density, as well as placing contours of the magnitude of the gradient of density along with density in Appendix B. To begin with introducing the DSMC simulation results, a holistic presentation of the DSMC results has been illustrated in figure 10, and discussed in detail. Solution files of the DSMC simulation at position 1 were then illustrated in the form of pressure plots against experimental data readings along with pressure contours on the SPIFEX data acquisition plate. Lastly, similar plots and contour

images of the DSMC simulation have been produced and cataloged in Appendix B for SPIFEX plate positions 2-9.

Upon examining the results from the DSMC simulation in positions 1-9, it is evident that the methodology presented in chapters 3 and 4 produce results in good agreement with experimental data. Also, as a result of the good agreement between the DSMC simulation and the experimental data, the method for coupling the RAMP solution to DAC has been granted an increased level of confidence, and will certainly be employed for future plume impingement analysis efforts.

As a final note, there are several components of this study that should be expounded upon in order to further augment the working knowledge of rarefied flow in the aerospace industry. First, in using the proven methods presented in this study, it would be greatly beneficial to expound upon the effect of constant azimuth and AoA with varying radii on plume impingement. Secondly, figure 8 indicates that a substantial portion of the flow from the RCS thruster nozzle changes direction and flows back towards the throat once it leave the nozzle. Hence, it seems pertinent to evaluate plume impingement at the base of the nozzle and immediate vicinity. Lastly, since this research focused on the pressure induced by the RCS plume, it did not include chemical kinetics outside of the nozzle, and hence could not ascertain any heating data as a result of plume impingement. Future work on this subject could greatly benefit rarefied flow analysis efforts by simulating the chemical kinetics throughout the computational domain, and consequently evaluate heating rates.

APPENDIX A: RAMP to DAC MATLAB CODE

```

clc
clear all
close all
A=importdata('New_Ramp_Solution.dat');
part_name='pos34_3'; % STL with no flow data this will be the name of the DAC input file as
well
cat=0.0; % catalytic effecency

%% parsing data from "A"
for i=1:length(A.textdata)
    line=A.textdata(i);
    if line{1}(2)=='T' && line{1}(3)=='='
        break
    end
end
b=regexp(line, '\d+(\.)?(\d+)?', 'match');
num=str2double([b{:}]);
numX=num(1);
numY=num(2);
numZ=num(3);
data=reshape(A.data', [numX*numY*7, 1]);
Xs= data(1:numX*numY)*1609/5280; %[m]
Ys= data(numX*numY+1:2*numX*numY)*1609/5280; %[m]
Vels= data(2*numX*numY+1:3*numX*numY)*1609/5280; %[m/s]
sigs= data(3*numX*numY+1:4*numX*numY); %[deg]
rhos= data(4*numX*numY+1:5*numX*numY)*515.378818; %[kg/m^3]
Press=data(5*numX*numY+1:6*numX*numY)*47.88; %[Pa]
Temps= data(6*numX*numY+1:end)*0.555556; %[K]
Nd=(6.0221409e+23)/8.3145*Press./Temps;
Mw=mean(rhos.*Temps*(8314)./Press);
disp(['Define ' num2str(Mw) ' g/mol for Mw in DAC Gas File'])
ringNum=1; % "1" is outter most
start=length(Xs)-ringNum*numX+1;
stop=start+numX-1;
x=vertcat(Xs(start:stop), -1); % ring selected for revolve
y=vertcat(Ys(start:stop), 0); % ring selected for revolve
x1=Xs;
y1=Ys;

```

```

x2=x;
y2=y;
z=y;
u=vertcat(Vels(start:stop).*cosd(sigs(start:stop)),0);
v=vertcat(Vels(start:stop).*sind(sigs(start:stop)),0);
phi=linspace(0,2*pi,length(x));
T=vertcat(Temps(start:stop),Temps(end));
Nd=vertcat(Nd(start:stop),Nd(end));
rhos=vertcat(rhos(start:stop),rhos(end));
Xs=zeros(length(phi),length(x));
Ys=zeros(length(phi),length(x));
Zs=zeros(length(phi),length(x));
us=zeros(length(phi),length(x));
vs=zeros(length(phi),length(x));
ws=zeros(length(phi),length(x));
Ts=zeros(length(phi),length(x));
Nds=zeros(length(phi),length(x));
Rhos=zeros(length(phi),length(x));
for i=1:length(phi)
    Xs(i,:)=x;
    Ys(i,:)=y*cos(phi(i));
    Zs(i,:)=y*sin(phi(i));
    us(i,:)=u;
    vs(i,:)=v*cos(phi(i));
    ws(i,:)=v*sin(phi(i));
    Ts(i,:)=T;
    Nds(i,:)=Nd;
    Rhos(i,:)=rhos;
end
%% plotting
plot(x1,y1,'*')
xlabel('Axial Distance [m]')
ylabel('Radial Distance [m]')
title('Total Solution Domain from RAMP')
figure
plot(x1,y1,'*')
hold on
plot(x2,y2,'r-*')
xlabel('Axial Distance [m]')
ylabel('Radial Distance [m]')
title('Total Solution Domain from RAMP')
legend('All Solution Radii Available','Selected Solution Radius (closed)')

```

```

figure
Surface=surface(Xs,Ys,Zs,us);
xlabel('Axial Distance [X(m)]')
ylabel('Radial Distance [Y(m)]')
zlabel('Radial Distance [Z(m)]')
title('Selected Solution Radius Revolved Around X-Axis')
grid on
cc=colorbar;
cc.Label.String = 'Axial Velocity [m/s]';
view([-10 30])
%% reading in plate
fv = stlread([part_name '.stl']);
faces=fv.faces;
verts=fv.vertices;
[vert,ia,ib]=unique(verts,'rows');%%removal of duplicate verticies
faces=ib(faces);
%% converting surf in figure 3 to stl
fv1=surf2tri(Xs,Ys,Zs);
Temp_nds=Nds;
Temp_Ts=Ts;
Temp_us=us;
Temp_vs=vs;
Temp_ws=ws;
Temp_rho=Rhos;
clear Nds
clear Ts
clear us
clear vs
clear ws
clear Rhos;
ck=1;
for i=1:length(Xs)
    for j=1:length(Xs)
        Nds(ck)=Temp_nds(i,j);
        Ts(ck)=Temp_Ts(i,j);
        us(ck)=Temp_us(i,j);
        vs(ck)=Temp_vs(i,j);
        ws(ck)=Temp_ws(i,j);
        rho(ck)=Temp_rho(i,j);
        ck=ck+1;
    end
end
end

```

```

Nds=Nds';
Ts=Ts';
us=us';
vs=vs';
ws=ws';
faceP=fv1.faces;
vertP=fv1.vertices;
% [vertP,ia,ib]=unique(vertP,'rows');%%removal of duplicate verticies
% faceP=ib(faceP);
% ck=1;
% for i=1:length(faceP)
%   if faceP(i,2)~=faceP(i,3) && faceP(i,1)~=faceP(i,3) && faceP(i,1)~= faceP(i,2)
%       tempFace(ck,:)=faceP(i,:);
%       ck=ck+1;
%   end
% end
% faceP=tempFace;
faceP=faceP+length(vert);
big_face=vertcat(faces,faceP);
big_vert=vertcat(vert,vertP);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% define DAC specific variables pg 18, DAC97 User Guide
NUMSP=length(big_vert);% number of surface noeds
NUMST=length(big_face);%number of surface TRIANGLES
NSPEC=1;%number of species

%% first record
fileID = fopen([part_name '.txt'], 'w');
Num_nodes=NUMSP;
triangles=NUMST;
species=NSPEC;

fprintf(fileID,' %12d %11d %11d\r',Num_nodes,triangles,species);
%% second record body XYZ points
for i=1:length(vert)
    fprintf(fileID,'%16.9f', vert(i,:));
    fprintf(fileID,'\r');
end
%% second record plume XYZ points
for i=1:length(vertP)
    fprintf(fileID,'%16.9f', vertP(i,:));

```

```

        fprintf(fileID, '\r');
end
%% third record body triangle connectivity
for i=1:length(faces)
    fprintf(fileID, '%15d', (faces(i,:)));
    fprintf(fileID, '\r');
end
%% third record plume triangle connectivity
for i=1:length(faceP)
    fprintf(fileID, '%15d', (faceP(i,:)));
    fprintf(fileID, '\r');
end
%% fourth record body Boundary conditions
%first number is group number
% second number is BC (1-5)
%1 solid wall
%2 outgassing wall
%3 inflow boundary
%4 outflow boundary
%5 free stream
for i=1:length(faces)
    fprintf(fileID, '%d %d', 1, 1);
    fprintf(fileID, '\r');
end
%% fourth record plume
%first number is group number
% second number is BC (1-5)
%1 solid wall
%2 outgassing wall
%3 inflow boundary
%4 outflow boundary
%5 free stream
for i=1:length(faceP)
    fprintf(fileID, '%d %d', 2, 3);
    fprintf(fileID, '\r');
end
%% 5th record body
for i=1:length(faces)%need exactly 27 inputs per face where Each vertex has 9 values
    %   vertex values
    %   1= accomodation coefficient
    %   2= surface catalysis
    %   3= wall temp

```

```

% 4= number density
% 5= Vx
% 6= Vy
% 7= Vz
% 8= flow temperature
% 9= mole fraction
fprintf(fileID,'%15.7E', 1,1,1);%accomodation coef vert 1, 2 and 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E',cat,cat,cat);%surface cat efficiency v1 2 and 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 270,270,270);%wall temp v1 v2 v3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 1e22,1e22,1e22);%number density 1, 2 and 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 0,0,0);%U velocity vert1 2 and 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 0,0,0);%V velocity vert1 2 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 0,0,0);%W velocity vert1 2 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 300,300,300);%flow temp vert 1 2 3
fprintf(fileID,'\r');
fprintf(fileID,'%15.7E', 1,1,1);%mole fraction vert1 2 3
fprintf(fileID,'\r');
end
%% 5th record plume
faceP=faceP-length(vert);
for i=1:length(faceP)%need exactly 27 inputs per face where Each vertex has 9 values
    fprintf(fileID,'%15.7E', 1,1,1);%accomodation coef vert 1, 2 and 3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', cat,cat,cat);%surface cat efficiency v1 2 and 3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', Ts(faceP(i,:)));%wall temp v1 v2 v3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', Nds(faceP(i,:)));%number density 1, 2 and 3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', us(faceP(i,:)));%U velocity vert1 2 and 3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', vs(faceP(i,:)));%V velocity vert1 2 3
    fprintf(fileID,'\r');
    fprintf(fileID,'%15.7E', ws(faceP(i,:)));%W velocity vert1 2 3
    fprintf(fileID,'\r');

```

```

        fprintf(fileID,'%15.7E', Ts(faceP(i,:)));%flow temp vert 1 2 3
        fprintf(fileID,'\r');
        fprintf(fileID,'%15.7E', 1,1,1);%mole fraction vert1 2 3
        fprintf(fileID,'\r');
end
fprintf(fileID,'\r');
test_int='123456';
fprintf(fileID,'%13s',test_int);
fprintf(fileID,'\r');
fclose(fileID);

```

Appendix B: Supplemental Figures

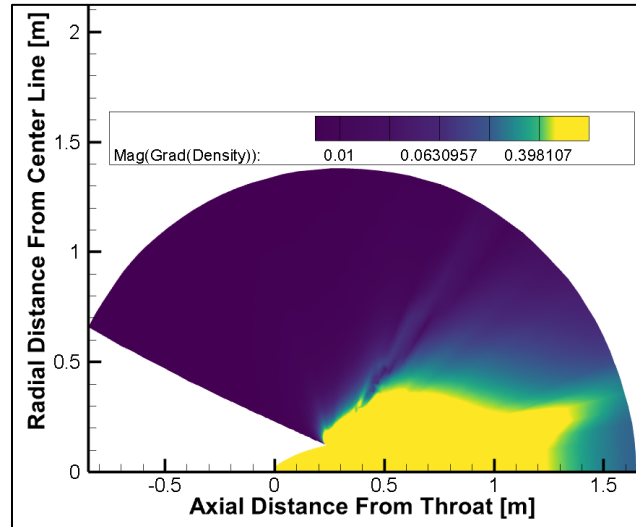


Figure 12 Ramp Results Magnitude of Gradient of Density

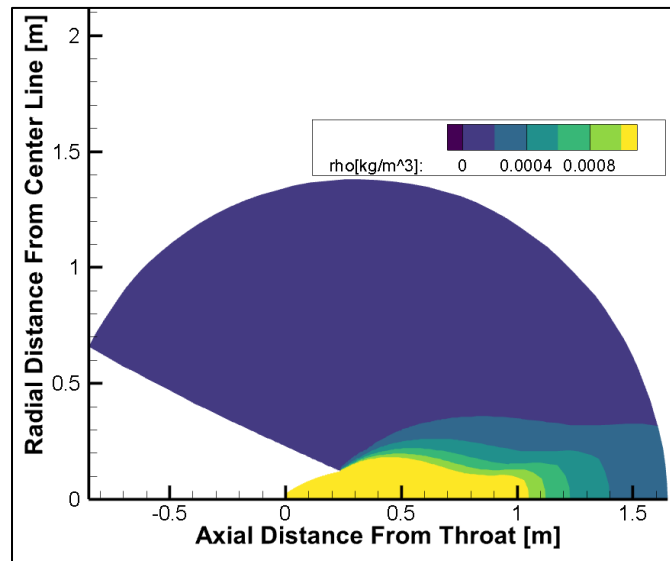


Figure 13 Ramp Results Density

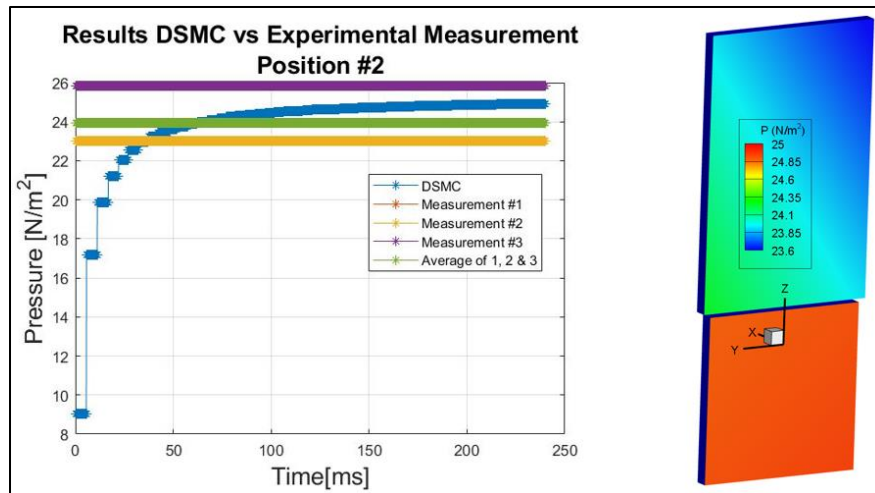


Figure 14 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #2

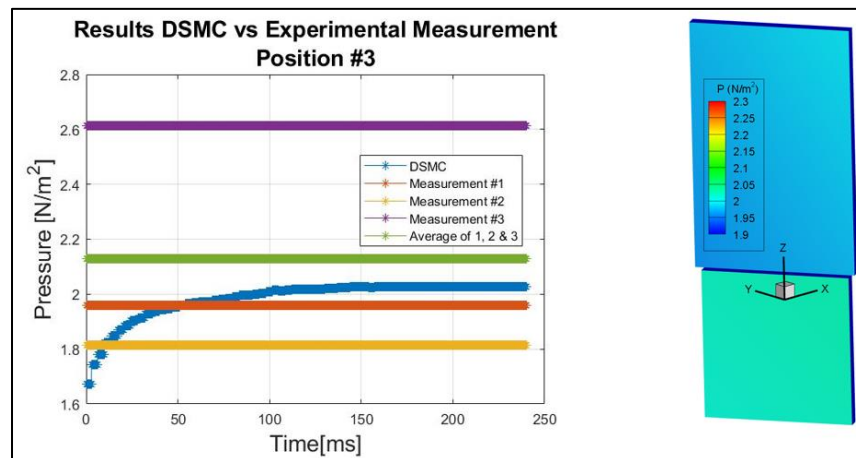


Figure 15 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) P[osition #3

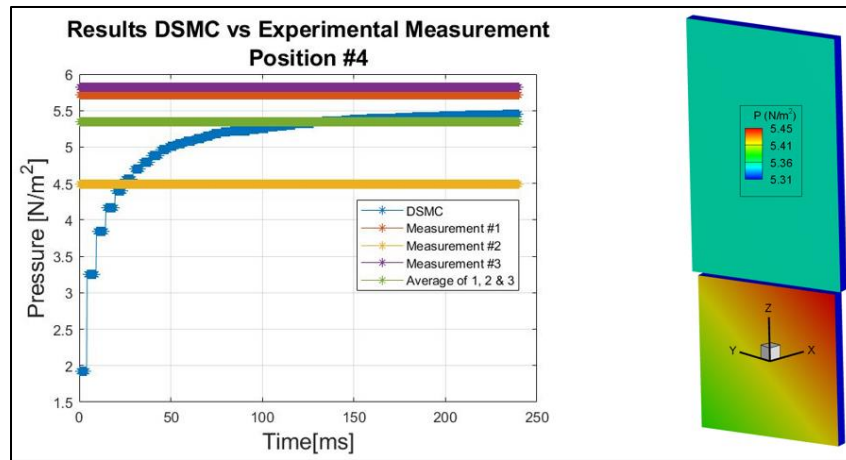


Figure 16 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #4

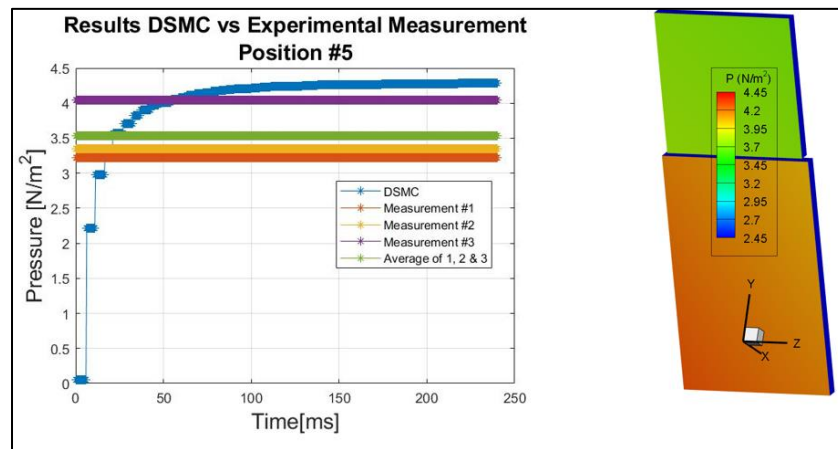


Figure 17 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #5

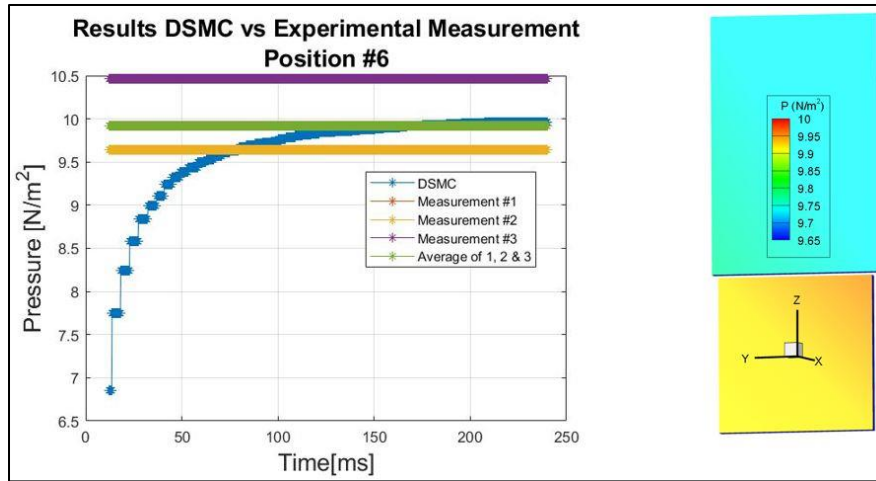


Figure 18 Steady State Measurement vs DSMC Results (Left),
Contour of Pressure DSMC Results (Right) Position #6

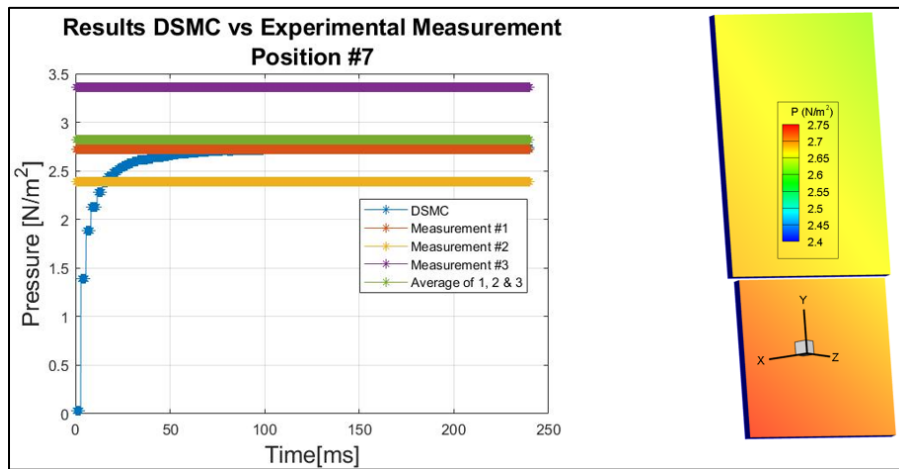


Figure 19 Steady State Measurement vs DSMC Results (Left),
Contour of Pressure DSMC Results (Right) Position #7

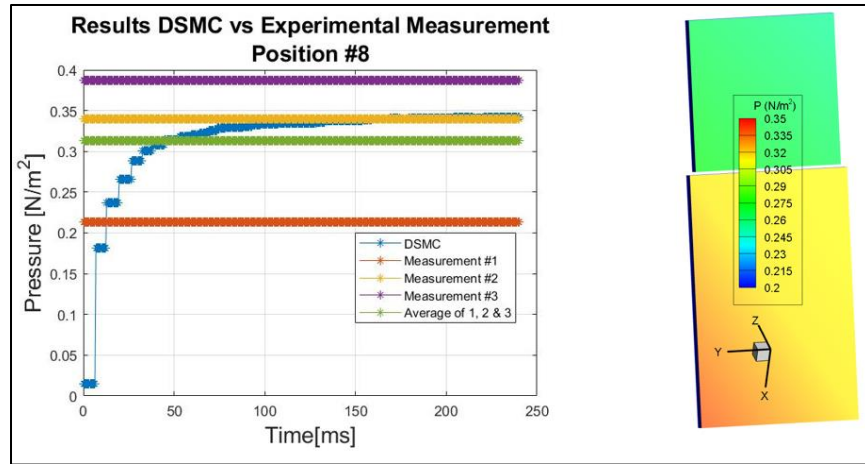


Figure 20 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #8

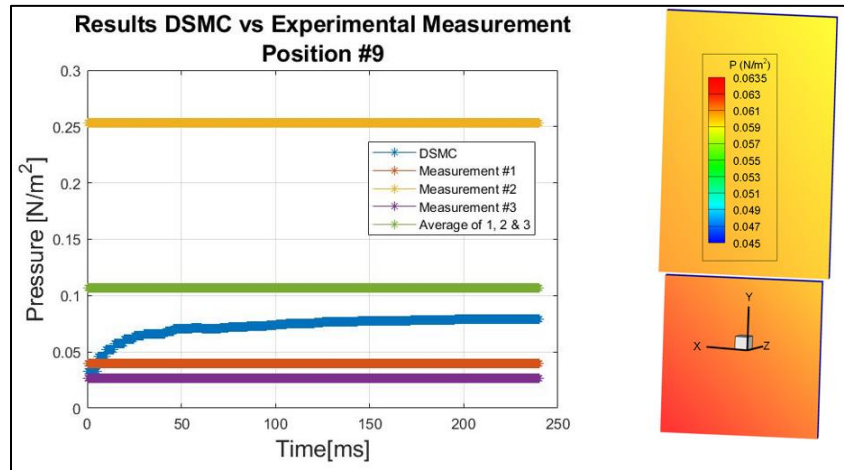


Figure 21 Steady State Measurement vs DSMC Results (Left), Contour of Pressure DSMC Results (Right) Position #9

LIST OF REFERENCES

- [1] “DSMC simulations of the Shuttle Plume Impingement Flight EXperiment(SPIFEX),” NASA Available: <https://ntrs.nasa.gov/search.jsp?R=20170007509>. [1] Hill, P. G., & Peterson, C. R. (2010). Mechanics and thermodynamics of propulsion (2nd ed.). S.I.: Pearson Education.
- [2] “MATLAB,” MathWorks Available: <https://www.mathworks.com/products/matlab.html>.
- [3] D, S. S., “High altitude chemically reacting gas particle mixtures. Volume 2: Program manual for RAMP2. [rocket nozzle and orbital plume flow fields],” NASA Available: <https://ntrs.nasa.gov/search.jsp?R=19850007493>. section 5-64
- [4] Anderson, J. D. (2010). Fundamentals of aerodynamics (In SI Units). New Delhi: Tata McGraw Hill Education Private Limited.
- [5] John, J. E. A., & Keith, T. G. (2006). Gas dynamics. Upper Saddle River, NJ: Pearson Prentice Hall.
- [6] Hash, D., & Hassan, H. (1995). A hybrid DSMC/Navier-Stokes solver. 33rd Aerospace Sciences Meeting and Exhibit. doi: 10.2514/6.1995-410
- [7] Iaccarino, G. (2009, February). ME469A. ME469A. Stanford.
- [8] Hadjiconstantinou, N. G., “The limits of Navier-Stokes theory and kinetic extensions for describing small-scale gaseous hydrodynamics,” Physics of Fluids, vol. 18, Jun. 2006, pp. 16–18.
- [9] Christou, C., and Dadzie, S. K., “On the numerical simulation of rarefied gas flows in micro-channels,” Journal of Physics Communications, vol. 2, 2018, pp. 8–9.
- [10] Bird, G. A. (1963). Approach to Translational Equilibrium in a Rigid Sphere Gas. Physics of Fluids, 6(10), 1518. doi: 10.1063/1.1710976
- [11] Bird, G. A., The DSMC method, North Charleston, SC: CreateSpace Independent Publishing Platform, 2013.
- [12] Niimi, T. (n.d.). Basic 11 High Knudsen Number Flows. Basic 11 High Knudsen Number Flows.
- [13] Tien, C. L., & Lienhard, J. H. (1979). Statistical thermodynamics. Washington: Hemisphere.
- [14] Sone, Y., Kinetic theory and fluid dynamics, Boston, MA: Birkhäuser, 2002.

- [15] Sturnfield, J., “COSMOL Conference,” COSMOL Conference.
- [16] Wilmoth, R., Carlson, A., and Lebeau, G., “DSMC grid methodologies for computing low-density, hypersonic flows about reusable launch vehicles,” 31st Thermophysics Conference, 1996, pp. 3–3.
- [17] Venkattraman, A., and Alexeenko, A. A., “Molecular Models for DSMC Simulations of Metal Vapor Deposition,” Molecular Models for DSMC Simulations of Metal Vapor Deposition, 2010, pp. 2–2.
- [18] G.A.Bird, Molecular Gas Dynamics and the Direct Simulation of Gas Flows, Oxford University Press, New York, 1994.
- [19] M. Kogan, Rarefied Gas Dynamics, Plenum Press, New York, 1969.
- [20] R.B.Bird, W.E.Stewart, and E.N.Lightfoot, Transport Phenomena, Wiley, New York, 2001.
- [21]. J.O.Hirschfelder, R.B.Bird, and E.L.Spotz, Journal of Chemical Physics 16(10), 968–981 (1948).
- [22] J.O.Hirschfelder, C.F.Curtiss, and R.B.Bird, Molecular Theory of Gases and Liquids, Wiley, New York, 1954
- [23] Venkattraman, A and Alexeenko, Alina A., "Molecular Models for DSMC Simulations of Metal Vapor Deposition" (2010). School of Aeronautics and Astronautics Faculty Publications. Paper 52 pp 3-3 para 2
- [24] Succi, S. (2001). The lattice Boltzmann equation for fluids dynamics and beyond. Oxford: Clarendon Press.
- [25] Venkattraman, A and Alexeenko, Alina A., "Molecular Models for DSMC Simulations of Metal Vapor Deposition" (2010). School of Aeronautics and Astronautics Faculty Publications. Paper 52 pp 3-3 para 2
- [26] Moss, J., and Bird, G., “DSMC Simulations of Hypersonic Flows with Shock Interactions and Validation with Experiments,” 37th AIAA Thermophysics Conference, 2004, pp. 2–2.
- [27] Mehta, N. A., and Levin, D. A., “Molecular-Dynamics-Derived Gas–Surface Models for Use in Direct-Simulation Monte Carlo,” Journal of Thermophysics and Heat Transfer, vol. 31, 2017, pp. 757–771.

- [28] Sorensen, C., Valentini, P., and Schwartzentruber, T. E., "Uncertainty Analysis of Reaction Rates in a Finite-Rate Surface-Catalysis Model," *Journal of Thermophysics and Heat Transfer*, vol. 26, 2012, pp. 407–416.
- [29] Padilla, J. F., and Boyd, I. D., "Assessment of Gas-Surface Interaction Models for Computation of Rarefied Hypersonic Flow," *Journal of Thermophysics and Heat Transfer*, vol. 23, 2009, pp. 97–97.
- [30] Allegre, J., Rafin, M., Caressa, J. P., "Experimental Investigation of Transverse Jet Effects Related to Hypersonic Space Vehicles," *Proceedings of the First European Symposium ESTEC*, Noordwijk, Netherlands, May 1991
- [31] Tartabini, P., Wilmoth, R., and Rault, D., "A systems approach to a DSMC calculation of a control jet interaction experiment," *28th Thermophysics Conference*, Jun. 1993, pp. 3–3.
- [32] Dunbar, B., "DSMC Code Simulates Rarefied Gas Dynamic Environments," NASA Available: <https://www.nasa.gov/centers/johnson/techtransfer/technology/MS-C-23445-1-dsmc-dac.html>.
- [33] Padilla, J. F., "NASA Technical Reports Server (NTRS)," NASA Available: [https://ntrs.nasa.gov/search.jsp?N=0&Ntk=All&Ntt=Comparison of DAC and MONACO DSMC Codes With Flat Plate Simulation&Ntx=mode matchallpartial&Nm=123|Collection|NASA STI|17|Collection|NACA](https://ntrs.nasa.gov/search.jsp?N=0&Ntk=All&Ntt=Comparison%20of%20DAC%20and%20MONACO%20DSMC%20Codes%20With%20Flat%20Plate%20Simulation&Ntx=mode+matchallpartial&Nm=123|Collection|NASA%20STI|17|Collection|NACA). Para 4 Pg. 2
- [34] Corbett, J., "Space Shuttle Program," NASA Available: https://www.nasa.gov/centers/wstf/about_us/our_history/space_shuttle_program.html
- [35] Anderson, J. D., *Fundamentals of aerodynamics (in SI units)*, New Delhi: McGraw-Hill, 2011. Pg. 169
- [36] Anderson, J. D., *Fundamentals of aerodynamics (in SI units)*, New Delhi: McGraw-Hill, 2011. Pg. 185
- [37] "Cramer's Rule," from Wolfram MathWorld Available: <https://mathworld.wolfram.com/CramersRule.html>.
- [38] Hall, N., "Mach Angle," NASA Available: <https://www.grc.nasa.gov/WWW/k-12/airplane/machang.html>.
- [39] KHAN, A. K. H. T. A. R. M. D., "Design of Minimum Length Supersonic Nozzle using the Method of Characteristics," *International Journal of Innovative Technology and Exploring Engineering Regular Issue*, vol. 2, Nov. 2013, pp. 1370–1374.

- [40] “Data Visualization & CFD post-processing software,” Tecplot Available:
<https://www.tecplot.com/>.
- [41] “Surf,” Surface plot - MATLAB Available:
<https://www.mathworks.com/help/matlab/ref/surf.html>.
- [42] Cercignani, C., Rarefied gas dynamics: from basic concepts to actual computations,
Cambridge: Cambridge University Press, 2000.pg 2 para 1
- [43] C. Cercignani, Mathematical Methods in Kinetic Theory, Plenum Press, New York (1969:
revised edition 1990)
- [44] C. Cercignani, Mathematical Methods in Kinetic Theory, Plenum Press, New York (1969:
revised edition 1990)
- [45] Luding, S., “On the relevance of ‘molecular chaos’ for granular flows,” ZAMM - Journal of
Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und
Mechanik, vol. 80, Oct. 1997, pp. 9–12.
- [46] L. Arkeryd. "On the Boltzmann equation. Part II The full initial value problem." Arch. Rat.
Mech. Anal 45. 17-34 (1972)