3-2005

Entropy Generation as a Means of Examining Continuum Breakdown

Christopher R. Schrock

ENTROPY GENERATION AS A MEANS OF EXAMINING CONTINUUM BREAKDOWN

THESIS

Christopher R. Schrock, Civilian

AFIT/GAE/ENY/05-M20

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GAE/ENY/05-M20

ENTROPY GENERATION AS A MEANS OF EXAMINING CONTINUUM BREAKDOWN

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Christopher R. Schrock, B.S.E.

Civilian

March 2005

AFIT/GAE/ENY/05-M20

ENTROPY GENERATION AS A MEANS OF EXAMINING CONTINUUM BREAKDOWN

Christopher R. Schrock, B.S.E
Civilian

Approved:

/signed/	March 2005
Maj. Richard McMullan (Chairman)	Date
/signed/	March 2005
Dr. Jose Camberos (Member)	Date
/signed/	March 2005
Lt.Col. Raymond Maple (Member)	Date

AFIT/GAE/ENY/05-M20

Abstract

In an effort to provide a means to quantify the validity and breakdown of the continuum equations of fluid flow, the concept of entropy generation is examined. It is reasoned that since this quantity is fundamentally related to the physics of nonequilibrium it should provide a good tool for the analysis of such phenomena. Furthermore, since one may formulate this parameter utilizing statistical mechanics and kinetic theory, there are no inherent mathematical limitations necessary in its calculation.

An analysis based on statistical mechanics and kinetic theory which leads to a form of entropy generation rate in terms of energy distribution functions is presented. This analysis is applied to monatomic and diatomic molecules. A numerical procedure is presented which allows for computation of these values using the Direct Simulation Monte Carlo Method (DSMC). Normal shock wave flows in argon and nitrogen were simulated at Mach numbers ranging from 1.2 to 10. Results are compared to Navier-Stokes predictions for the same shocks. The increase in entropy production entering the shock is predicted later by the Navier-Stokes equations than by DSMC, indicating that virtually no nonequilibrium phenomena are observable in the Navier-Stokes data until after the shock region has already been entered. Because of this, breakdown parameters based on continuum data will fail to capture the initial nonequilibrium and will not provide good measures of continuum breakdown. At lower Mach numbers, entropy production is on the order of the scatter in the DSMC data, which speaks to the ability of this parameter to characterize continuum onset. Observable error in the flow variables is shown to be a strong function of entropy generation in the flows considered, suggesting that this parameter is a good indicator of continuum breakdown when computed using kinetic approaches.

Acknowledgements

This research would not have been possible were it not for the contribution of several individuals. First, I would like to thank my advisor, Maj. Richard McMullan whose guidance and support proved invaluable throughout the term of this research. I would also like to thank Dr. Jose Camberos (AFRL/VAAC) for lending his expertise in the area of gas dynamics and kinetic theory, as well as providing the Navier-Stokes solver used in this research. I would also like to thank Professor Iain Boyd of the University of Michigan for providing the DSMC code for this work, as well as Dr. Quanhua Sun for assisting in its usage. Finally, I would like to thank the Computational Sciences Branch of the Aeromechanics Division of the Air Vehicles Directorate of the Air Force Research Laboratory (AFRL/VAAC) for financially supporting this research under Doug Blake.

Christopher R. Schrock

Table of Contents

Page

List of Figures

List of Tables

List of Abbreviations

List of Symbols

Constants

$\hbar$ Normalized Planck Constant ($\hbar = h/2\pi = 1.054571596 \times 10^{-34} J/K$)

h Planck Constant ($6.62606873 \times 10^{-34} J/K$)

k Boltzmann Constant ($1.3806503 \times 10^{-23} J/K$)

Greek Symbols

α Function used in Defining Discrete Distributions; Matrix used in Least Squares Calculation

β Term Computed in Least Squares Calculation

ϵ Energy of a Particle, Quantum Level, or Group of Quantum Levels

Γ Chapman-Enskog Perturbation Function; Number of Quantum States below a Given Energy

γ Specific Heat Ratio; Matrix used in Least Squares Calculation

κ Coefficient of Thermal Conductivity

λ Mean Free Path; Second Coefficient of Viscosity

$\vec{\Lambda}$ Internal Components of Generalized Momentum

μ Coefficient of Viscosity; Reduced Mass of a Two Particle System $\mu = m_1 m_2 / (m_1 + m_2)$

ν Intermolecular Collision Rate

Ψ Wavefunction

ρ Fluid Density

σ Collisional Cross Section

τ Shear Stress Tensor

Θ_{rot} Characteristic Temperature of Rotation

Θ_{vib} Characteristic Temperature of Vibration

Ω Statistical Multiplicity or Total Degeneracy of a System

ω Variable Hard Sphere Viscosity Exponent

Roman Symbols

$\bar{c}$ Mean Molecular Speed

δc	Width of Velocity Distribution Subinterval
$\vec{C}$	Thermal (or peculiar) Velocity
$\vec{c}$	Molecular Velocity Vector
c_j	Number of Quantum States in the j^{th} Group
$\delta\epsilon$	Width of Energy Distribution Subinterval
E	Aggregate Energy of a System of Particles; Least Squares Error
e	Specific Internal Energy of the Gas; Error in Gradient Approximation by Least Squares Method
e_Q	Error Observed in Navier-Stokes Solution for Flow Variable Q
$\vec{F}$	External Force
f	Probability Distribution Function
$\vec{g}$	Relative Velocity Vector of a Collision Pair
g_k	Degeneracy of the k^{th} Energy Level
$\mathcal{H}$	Boltzmann's H-Function
H	Hamiltonian
Kn	Knudsen Number
L	Characteristic Length
M	Mach Number
m	Particle Mass
N	Number of Particles
n	Number Density
n_1, n_2, n_3	Translational Quantum Numbers
n_{int}	Number of Subintervals in Velocity Distribution
N_{PDF}	Number of Simulated Particles used to Create Distribution Function
n_{rl}	Number of Rotational Levels in Rotational Energy Distribution
n_{rot}	Rotational Quantum Number
n_{vib}	Vibrational Quantum Number
n_{vl}	Number of Vibrational Levels in Vibrational Energy Distribution
$\mathcal{P}$	Bird's Continuum Breakdown Parameter

$\vec{P}$	Generalized Momentum Vector
P	Momentum
p	Pressure
$\vec{Q}$	Generalized Coordinate Vector
$\vec{q}$	Heat Flux Vector
Q	Generic Flow Variable
$\vec{r}$	Particle Position Vector
R	Specific Gas Constant
r_e	Equilibrium Atomic Separation Distance
S	Entropy
s	Specific Entropy
$\tilde{s}$	Entropy Density $\tilde{s} = \rho s$
$\vec{\mathcal{T}}$	Entropy Density Flux Vector
T	Temperature
t_f	Characteristic Flow Time
t_s	Density Shock Thickness
V	Potential Energy
$\mathcal{W}$	Statistical Weight of Particle
W_i	Total Degeneracy of the i^{th} Macrostate
w_j	Degeneracy of the j^{th} Group of Quantum Levels; Weighting Term used in Least Squares Calculation

ENTROPY GENERATION AS A MEANS OF EXAMINING CONTINUUM BREAKDOWN

I. Introduction

Many of the interesting and open problems remaining in the field of fluid mechanics are concerned with the study of nonequilibrium phenomena. Nonequilibrium phenomena in fluids typically result from the propensity of the constituent molecules to undergo changes in their internal energy state during a collision with a surface or another molecule, as well as their ability to react with other molecules upon collision. These events occur at some finite rate in the fluid, and not every encounter results in such changes. If these events occur in such a way as to alter the macroscopic properties of the fluid over time, they may be regarded as nonequilibrium phenomena. These effects can be responsible for causing a number of fluid properties that might normally be treated as constants to vary at a finite rate. Among these are fluid composition, viscosity, thermal conductivity, specific heats, etc. The variable nature of these properties in regions of nonequilibrium changes the manner in which energy and momentum are transferred in the fluid, which alters the macroscopic thermodynamic and flow variables throughout the flow.

Nonequilibrium effects complicate the analysis of fluid flow. The traditional continuum equations of fluid mechanics do nothing in and of themselves to address these effects. In fact, it can be shown that these equations permit only small departures from equilibrium in the translational energy mode of the molecules, but must be augmented with additional externally derived equations to compensate for other forms of nonequilibrium. Integration of such models with the continuum equations in some cases is not well understood. Additionally, describing the dynamics of the phenomena itself may be nontrivial, complicating the development of augmenting models.

The branch of gas dynamics which is formally concerned with accounting for changes in the gas at the molecular level is kinetic theory. Kinetic theory attempts to track, at least statistically, the energy state, momenta and position of every particle in the gas as a function of time, and accounts for the variation of these properties due to collisions. This information can than be integrated over the collection of particles to obtain the macroscopic properties of the gas. No inherent assumptions regarding equilibrium are required and only the particle collision dynamics require modeling. The assumptions which go into such models typically do not exhibit restricted validity in regions of nonequilibrium. These properties make kinetic theory ideal for the study of nonequilibrium phenomena.

The present research utilizes the tools of kinetic theory to examine continuum breakdown/onset in regions of nonequilibrium. Entropy generation is formulated using statistical mechanics and kinetic theory. This parameter is examined because of its fundamental relation to nonequilibrium phenomena.

Unfortunately, the great flexibility afforded by kinetic theory is not without cost. The governing equation of kinetic theory, the Boltzmann Equation, is a nonlinear integro-differential equation for a probability distribution function which statistically describes the state of the particles as a function of time. This equation must be solved in a space with dimension equal to the number of position coordinates a particle may possess, plus the number of momentum components a particle may possess, plus, if the flow is unsteady, the additional dimension of time. Thus, for a steady flow of a monatomic gas in three physical dimensions, the equation must be solved in a space of dimension no less than six. For polyatomic molecules, which may possess several components of angular momentum and many vibrational degrees of freedom, the dimension of the space grows even higher. As a result, closed form solutions of the Boltzmann equation for flows of practical interest are difficult to obtain. Computational methods for the Boltzmann equation have been, and continue to be, developed. However, due to the inherent complexity of the equation itself, such methods are correspondingly complicated. Some of these methods are simplified by

assuming a certain form of the distribution function, but in doing so, they lose their ability to render accurate solutions in regions of nonequilibrium.

Fortunately, other kinetic approaches exist, which, though demanding by today's technology state, allow us to examine many practical flows of interest. Rather than trying to numerically solve the mathematics governing the fluid, these methods simulate the physics occurring in the fluid. That is, instead of solving for a function that describes statistically the evolution of the gas, these methods simulate the actual physics of the various particle interactions occurring in the gas which that function seeks to account. Accounting for the state of every single particle and every single molecular encounter would be quite difficult, if not impossible with present technology, unless the density of the gas is quite low or the solution domain is quite small. For this reason, the Direct Simulation Monte Carlo Method (DSMC) was developed by Bird[5].

DSMC stays true to the heart of kinetic theory without actually attempting to solve the Boltzmann equation. This method seeks to track changes due to molecular encounters by simulating only a fraction of the particles in the gas. Each particle is given a statistical weight $\mathcal{W}$, equal to the number of actual particles that it represents. During a given increment of time, representative collisions are simulated between these particles throughout the gas. Each of these collisions is then taken to represent $\mathcal{W}$ actual collisions which occurred in that time increment. When the instantaneous results are aggregated over many simulation steps, one obtains results which converge to the reality of what is actually happening in the gas.

Having a kinetic-based tool like DSMC allows one to examine the very phenomena which tend to invalidate the continuum equation sets. This invalidation has been termed *continuum breakdown* [9, 12, 39], but in many circumstances may be better characterized as *equilibrium breakdown*, as commonly it is the presence of significantly nonequilibrium effects which cause the continuum equations to fail. Furthermore, if it were possible to predict the regions of the flow field where the continuum equations

would fail, theoretically a hybrid method could be developed which combines the relative speed of traditional computational fluid dynamics (CFD) with the accuracy of DSMC in regions of nonequilibrium. This has been attempted in the past with varying degrees of success [23, 13, 25, 26, 35]. A major obstacle to applying this concept is the ability to determine a parameter which consistently identifies regions of nonequilibrium or continuum breakdown.

Identification of such a parameter allows one to do more than simply hybridize the two computational methods. It also allows one to study the fundamental physics which cause the equations to break down. Furthermore, it allows one to evaluate the performance of higher-order continuum equation sets such as the Burnett and Super-Burnett equations. Moreover, such a parameter should allow for the verification of augmented versions of the continuum equations which include models to attempt to account for other forms of nonequilibrium beyond translational. Before further discussing such parameters, it is beneficial to discuss phenomena associated with continuum breakdown.

1.1 A Discussion of Nonequilibrium Phenomena and Continuum Breakdown

The equilibrium state of a gas is one in which the distribution of molecular energy states or composition does not vary with time. Although intermolecular collisions continuously occur, and energy is continuously transferred through these collisions, the gas, when viewed as a whole, maintains the same number of particles of a given energy class in each of its possible energy modes. Likewise, if the gas has the opportunity to react, in equilibrium, the rate at which the forward reaction progresses is exactly equal to the rate at which the reverse reaction progresses, so that the composition of the gas does not vary over time.

Nonequilibrium is, in fact, a transient state. It is the process by which the gas settles or relaxes into its new equilibrium configuration. A common example of such phenomena is that of a shock wave. Upstream of the shock, the gas is in an equilibrium

state at some temperature and mean flow velocity. Far downstream of the shock, the gas is also in a state of equilibrium at a lower mean velocity and higher temperature. Although equilibrium prevails on either side of the shock, the distribution of molecular velocities is markedly different. Due to the change in mean flow velocity across the shock, the upstream distribution is centered at a higher value than the downstream distribution. The higher temperature causes the downstream distribution to be more spread out than the upstream distribution. Transition from the upstream distribution to the downstream distribution is achieved through molecular collisions.

The thickness of the shock wave is precisely the distance required for the flow to undergo enough molecular collisions to achieve the final equilibrium state. This is an example of translational nonequilibrium. Within the shock, the velocity distribution is not Maxwellian and is in fact almost bimodal as the two distributions blend together. This behavior can be seen in the experimental data of Pham-Van-Diep and Erwin[32]. It is precisely for this reason that the continuum equations fail within the shock, as they are derived assuming only small deviations from the Maxwellian. Additionally, if the gas is capable of storing energy internally, the increased temperature across the shock may cause the gas to redistribute energy over these modes. This causes a region of rotational and vibrational nonequilibrium to exist. Furthermore, if the gas is capable of reacting, the increased temperature will cause the reaction rate to change and a state of chemical nonequilibrium will appear.

The continuum equations are also invalidated in regions of very low density, that is when the gas is rarefied. When a relatively small number of particles occupy the flow field of interest, the continuum hypothesis is obviously invalidated as large regions of the flow field may contain no particles at all. This may occur naturally, or may be caused by rapid expansion of the flow. In this case, the collision rate drops drastically and the continual decrease in temperature predicted by the continuum equations cannot be maintained. This is because as the number of collisions the particles experience continues to decrease, the translational temperature eventually levels out or "freezes" as the free molecular regime is approached [3, 5, 9].

Alternatively, when the length scale of consideration is on the order of the mean free path in the gas, the flow may "appear" rarefied even at normal densities, and the continuum hypothesis will no longer hold. This case is of practical interest for the analysis of microscale aerodynamics and fluid flow through micro-electro-mechanical systems (MEMS) [16, 17, 36, 37].

As seen in the previous examples, the validity of the continuum equations is inherently tied to the collision rate in the gas. Letting ν represent the intermolecular collision rate and t_f represent the characteristic flow time, the Knudsen number, Kn may be defined as follows.

$$Kn = \frac{1}{\nu t_f} \tag{1}$$

Alternatively, letting $\bar{c}$ be the mean molecular speed, λ be the mean free path (the average distance traveled by a molecule before encountering a collision), and L represent a characteristic length scale, the above equation becomes,

$$Kn = \frac{1}{\nu t_f} = \frac{1}{(\bar{c}/\lambda)\,(L/\bar{c})} = \frac{\lambda}{L} \tag{2}$$

Traditionally the validity criterion for the continuum equations has been stated as $Kn << 1$. However, the choice of characteristic length scale may be somewhat ambiguous. Choosing a length scale based upon the geometry over which the flow is considered results in a parameter which may describe globally how well the continuum equations apply, however, it does nothing to address local flow physics which may occur. A better choice is to consider a local length scale related to the physics occurring in the flow field. This may be a quantity such as a shock thickness, boundary layer thickness, or some other length associated with the local flow physics. The validity of the various equation sets as dependent upon such a local Knudsen number is summarized in Figure 1.

The limits shown in Figure 1 are based upon experimental observations and should not be thought of as hard limits or having a rigorous mathematical basis.

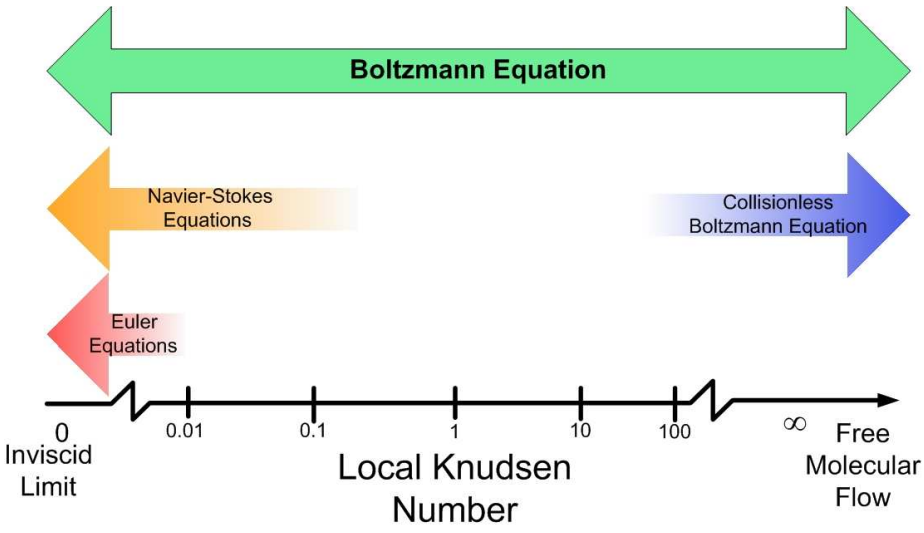


Figure 1: Knudsen Number Ranges for Various Equation Sets

Indeed this is the very reason it is desired to study continuum breakdown. The Euler equations are strictly based upon the Maxwellian velocity distribution, which is the prevailing distribution when the gas is in equilibrium. Furthermore, as they include no viscous terms, they are only valid at very low Knudsen number, or when the collision rate becomes quite large. When this occurs, the gas is able to redistribute its energy to accommodate varying conditions almost instantly, and the flow is practically in equilibrium everywhere.

The Navier-Stokes equations exhibit an extended Knudsen number validity over the Euler equations as seen in Figure 1. Pinpointing exactly where they become invalid is somewhat difficult, but it is generally accepted that this occurs at Knudsen numbers somewhere around 0.1 [5]. The Navier-Stokes equations can be derived by using the first Chapman-Enskog correction to the Maxwellian distribution function [14, 38]. This means that the Navier-Stokes equations are based upon a small departure from the equilibrium distribution function and will fail in regions of significant nonequilibrium.

The Boltzmann equation is seen to apply at all Knudsen numbers. This illustrates the great flexibility offered by kinetic theory, and shows why kinetic methods

are ideal for examining the breakdown of the continuum equations. At the opposite end of the spectrum, a simplified version of the Boltzmann equation, known as the collisionless Boltzmann equation, is seen to apply as the free molecular limit is approached. In this case, no information is transfered by collisions, and the integral terms defining collisional transfer in the Boltzmann equation may be taken to be identically zero and solution is greatly simplified.

Many flow scenarios contain several physical processes that exhibit local Knudsen numbers at many different locations on this spectrum. This is particularly the case in some hypersonic flows. Consider a typical blunt body in hypersonic flow. The physics involved with the associated strong shocks invalidate the continuum equations. Aft of the shock the continuum equations may hold. Close to the body, strong gradients exist across the boundary layer and the process of activating the internal modes of the gas may occur in nonequilibrium fashion. Further downstream the flow may expand around the body to the point at which the density is too low for the continuum equations to hold. Such a mixed flow field exemplifies why a hybrid method may be desirable, illustrates the importance of understanding where the continuum equations are and are not valid, and emphasizes the need for a good understanding of nonequilibrium phenomena.

1.2 Previously Examined Parameters for Quantifying Continuum Breakdown

In the past, several parameters have been examined in an attempt to quantify continuum breakdown. For obvious reasons, many of the initial investigations considered parameters in the form of a Knudsen number based upon the local flow physics. This approach is based on the observation that if flow variables change over relatively small distances, it is unlikely that enough collisions are experienced for the gas to adjust to the new conditions in an equilibrium manner.

Bird [3] was one of the first to examine the validity of such a parameter using DSMC. He did so in the context of nonequilibrium resulting from expansions of the

gas. In doing so, he suggested the following parameter as a means of quantifying equilibrium breakdown.

$$\mathcal{P} = \frac{1}{\nu}\left|\frac{D\left(\ln\rho\right)}{Dt}\right| \tag{3}$$

where ρ is the density of the gas. In the case of one dimensional steady flow, this becomes

$$\mathcal{P} = \frac{u}{\rho\nu}\left|\frac{d\rho}{dx}\right| = M\sqrt{\frac{\pi\gamma}{8}}\frac{\lambda}{\rho}\left|\frac{d\rho}{dx}\right| \tag{4}$$

where u is the streamwise fluid velocity and M is the Mach number in the gas. Bird found that a value of $\mathcal{P}$ greater than 0.02 tended to predict the onset of nonequilibrium well.

While this parameter seems to indicate continuum breakdown well in the specific case of gaseous expansions, it was later observed by Wang and Boyd [39] that this parameter did not predict nonequilibrium well in regions of low velocity. As seen in equation (4), the Mach number dependence will drive P to zero in regions of low velocity no matter what the degree of nonequilibrium. Boyd proposed examining a Knudsen number based upon the gradient local length (GLL) as a means of examining continuum breakdown [11].

$$Kn_{GLL} = \frac{\lambda}{Q}\left|\nabla Q\right| \tag{5}$$

where Q is some flow property of interest. The dependence of Bird's parameter on a gradient based local Knudsen number is readily seen in equation (6).

$$\mathcal{P} = M\sqrt{\frac{\pi\gamma}{8}}Kn_{\rho} \tag{6}$$

where $Kn_{\rho} = \frac{\lambda}{\rho}\left|\frac{d\rho}{dx}\right|$ is the Knudsen number based upon the gradient local length of the density.

Determining which flow variable is the best to use in equation (5) for examining continuum breakdown is nontrivial, and may vary from one flow scenario to another.

For this reason, Wang and Boyd proposed using the following parameter[35].

$$Kn_{max} = max\left(Kn_{\rho}, Kn_{T}, Kn_{V}\right) \tag{7}$$

where Kn_T is the Knudsen number based upon the gradient local length of the temperature, and Kn_V is the Knudsen number based upon the gradient local length of the fluid velocity. They proposed that significant nonequilibrium was indicated when this parameter became greater than about 0.05

Unfortunately, while such parameters are relatively easy to compute, they do not necessarily have firm theoretical grounding. Qualitatively, they make some sense, in that many forms of nonequilibrium are set into motion by the presence of strong gradients in the flow variables. Such parameters are also tied to the traditional understanding of a Knudsen number restriction on the continuum equations. However, the fact that multiple parameters must be computed suggests that there is something larger at work here. Furthermore, Boyd has observed that the Knudsen number parameters do not capture the onset of a shock front very well[9].

Other researchers have suggested examining parameters which quantify the magnitude of the perturbation terms in the velocity distribution assumed by the continuum equations. To that end, Boyd[9] has examined the perturbation terms of the Chapman-Enskog distribution function included in the Navier-Stokes equations as a measure of continuum breakdown. This parameter is given below.

$$\Gamma\left(\vec{C}\right) = 1 + q_i^* C_i \left(\frac{2}{5}C^2 - 1\right) - \tau_{ij}^* C_i C_j \tag{8}$$

where, $\vec{C}$ is the thermal (or peculiar) velocity, and $\vec{q}^*$ and τ^* are the normalized heat flux vector and shear stress tensor defined below.

$$\vec{q}^* = \frac{\vec{q}}{p}\sqrt{\frac{2m}{kT}} \tag{9}$$

$$\tau^* = \frac{\tau}{p} \tag{10}$$

where $\vec{q}$ and τ are the dimensional heat flux vector and shear stress tensor, respectively, p is the pressure, T is the temperature, m is the mass of a gas particle, and k is the Boltzmann constant. When Γ is perturbed much from unity, the main assumption of Chapman-Enskog theory breaks down, as the distribution is no longer a small perturbation of the Maxwellian. This occurs when significant nonequilibrium effects are present. This parameter is tied directly to the mathematical assumptions behind the derivation of the Navier-Stokes equations.

Rather than evaluate the full Chapman-Enskog term, others have simply examined the magnitude of the shear stress and heat flux terms to determine regions of nonequilibrium. Michaelis utilized the quantities τ^* and $\vec{q}/pa$ (effectively $\vec{q}^*$), where a is the speed of sound, to determine nonequilibrium in his hybrid method [29]. In fact, he was able to show that the Chapman-Enskog velocity distribution actually predicted negative probabilities over some portion of the velocity spectrum when these terms became of order one. Michaelis suggested nonequilibrium is signaled when either of these values became greater than 0.5, though Garcia has suggested a stricter criterion of $B = max\left(\|\vec{q}^*\|, \|\tau^*\|\right) < 0.2$ as the regime in which the continuum equations are valid[21].

It should be noted that the computation of the heat flux and shear stress terms involves the combination of a number of flow gradients. This partially speaks to why multiple gradient-based Knudsen parameters are required to obtain a reasonable predictor.

1.3 Entropy Generation as a Means of Quantifying Continuum Breakdown

While many of the previously discussed parameters attempted to quantify breakdown by examining the terms involved in the mathematical formulation of the continuum equations, it is possible to consider a parameter that is connected directly to

the physics which invalidates the continuum equations. From thermodynamics, we know that a system in equilibrium is characterized by zero entropy production. A system relaxing to a new equilibrium state is characterized by positive entropy production. Therefore, all nonequilibrium phenomena in a flow field of interest must, by definition, be entropy generators. For this reason, entropy generation should provide a good means for examining the physics of continuum breakdown.

In the big picture, entropy generation is the only quantity which has firm theoretical justification in examining the physics of equilibrium breakdown. Thermodynamics does not tell us specifically that large gradients will exist in regions of nonequilibrium, nor does it specifically state what will happen to the heat flux vector or shear stress tensor. It does, however, tell us exactly what happens to the entropy in regions of nonequilibrium, namely that it must be increasing.

Entropy production is the larger force at work in regions of nonequilibrium. It must include all of the effects which can contribute to nonequilibrium. In the continuum sense, it is possible to formulate expressions for entropy generation directly involving the heat flux and shear stress terms, which, in turn, can be related to various flow gradients. Therefore, entropy generation should include, in one parameter, all of the effects which the previously discussed parameters attempted to quantify. So when examining entropy production, one need not worry that any of the possible contributors to nonequilibrium have been ignored.

To that end, entropy generation was examined by Camberos, Chen and Boyd as a parameter for examining continuum breakdown [12, 15]. Several flow scenarios were investigated including a laminar boundary layer, normal shock, and hypersonic cylinder flare configuration. Unfortunately, in the flow scenarios examined, the entropy parameters were found to be no more effective in predicting continuum breakdown than the Knudsen parameters. All of the parameters were shown to fail in predicting continuum breakdown in the shock front. This is counterintuitive since the entropy generation must contain all of the nonequilibrium phenomena at work. The answer

as to why this occurred lies in the fact that their analysis was performed using data derived from the Navier-Stokes equations which already have inherent assumptions regarding equilibrium in them.

It is postulated that any parameter directly related to nonequilibrium, when calculated using continuum data with inherent equilibrium assumptions, will fail to realize its full, nonequilibrium potential. That is to say, because a small deviation from equilibrium is assumed in the derivation, the mathematics will attempt to limit the extent of nonequilibrium observable in the flow. For the most part, many of the previously discussed parameters were examined as switching parameters for hybrid methods, and, as such, have in many cases been investigated using continuum data.

For this reason, the present study undertakes the investigation of entropy generation as a means of predicting continuum breakdown utilizing kinetic methods which are free from equilibrium assumptions. Entropy is formulated via statistical mechanics to avoid any dependence on macroscopic quantities. These results are combined with kinetic theory to obtain the entropy generation rate in the gas. These derivations are performed in Chapter 2 for the cases of translational, rotational, and vibrational entropy generation.

The formulas derived in Chapter 2 are integrated into the DSMC code MONACO developed by Boyd at the University of Michigan [10]. The numerical methods employed in implementing these calculations are discussed in Chapter 3. Several simulations of the normal shock problem are performed, as this problem seems to represent a sticking point for many parameters computed from continuum data and provides an ideal case for examining nonequilibrium. A discussion of these simulations is also contained in Chapter 3. These simulations are performed for argon, in which only translational nonequilibrium was considered, and for nitrogen, in which the additional modes of rotational and vibrational nonequilibrium are considered. Although only these three forms of nonequilibrium are examined, the methods of analysis are

general enough to permit practically any form of nonequilibrium (e.g. electronic nonequilibrium, chemical nonequilibrium, etc.) to be considered.

These results are then compared against data obtained by numerically integrating the one-dimensional Navier-Stokes equations. A thorough comparison and discussion of these results is presented in Chapter 4. It is shown that the Navier-Stokes equations tend to delay the onset of nonequilibrium and also limit the extent of observable nonequilibrium in the normal shock problem.

Based upon these results several conclusions are summarized in Chapter 5 regarding the viability of entropy generation as a means of examining continuum breakdown in regions of nonequilibrium. It is postulated that due to the delayed and diminished nonequilibrium observed in the Navier-Stokes results, no parameter calculated using such data will be able to reliably quantify the onset of nonequilibrium. Chapter 5 also includes recommendations for future work in this area of research.

II. Theoretical Development

This chapter presents an analysis based on statistical mechanics and kinetic theory that leads to a formulation of entropy generation in terms of energy distribution functions. A short introduction to entropy viewed from a statistical perspective is given in which a relation for the entropy in terms of the statistical multiplicity of a system is given. The next few sections relate to how this statistical multiplicity can be determined using energy distribution functions and results from quantum mechanics. The entropy is then shown to be accessible through expectation values of the energy distribution functions, and a short discussion of Boltzmann's H-theorem is presented. The chapter concludes by combining these results with the Boltzmann transport equation to provide an expression for the entropy generation rate.

2.1 Entropy from a Statistical or Microscopic View

To study the phenomena of entropy generation without regard to macroscopic assumptions, one must draw upon the principles of statistical mechanics. One interpretation of entropy has traditionally been stated as "a measure of disorder". Other interpretations exist, but drawing upon this, consider a system of N particles with some aggregate energy E. Since E is an aggregate quantity, each particle is allowed to posses some amount of energy. It may store this energy in several ways, for example: it may translate, rotate, vibrate, become electronically excited, etc. Furthermore, there are a number of different ways to distribute the energy of the system to the N individual particles and still realize the aggregate energy E. This notion is known as *degeneracy* or *statistical multiplicity*.

Without a knowledge of quantum mechanics, one might conclude that there are an infinite number of ways to distribute the energy of the system to the individual particles and still realize the aggregate total. However, all of the various modes in which a particle can store energy are in fact, quantitized; there are only distinct values of energy that each mode is allowed to possess and the notion of a continuous energy spectrum is flawed. For this reason, a particle can manifest its energy in a finite

number of ways, and the number of ways in which the system of particles can achieve the aggregate total E is also finite. It is through this property that entropy can be associated with a measure of randomness or disorder.

Systems with a larger statistical multiplicity are more random than those of lower statistical multiplicity, as there are more ways for the system to realize its energy state. Since a system with a larger statistical multiplicity is more random, it will possess more entropy than its counterparts. Boltzmann's relation makes a direct connection between the concept of statistical degeneracy and entropy. This relation can be exhibited through the thought experiment outlined in Figure 2.

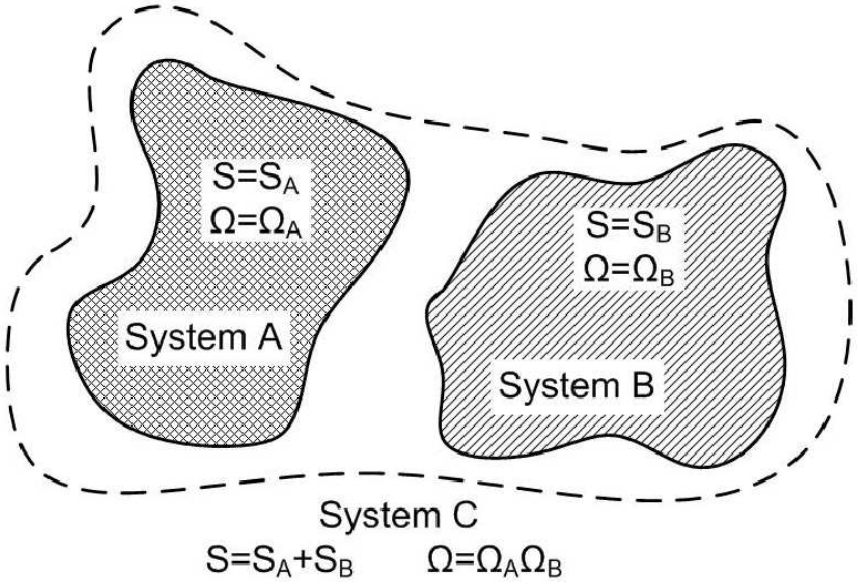


Figure 2: Illustration of Entropy-Multiplicity Dependence

Consider two isolated thermodynamic systems of particles. Denote the entropy of the systems by S_A and S_B respectively, and the statistical multiplicity of each by Ω_A and Ω_B. Consider a third thermodynamic system composed of the previous two, still isolated systems. Since entropy is additive, the entropy of the new system is given by $S_C = S_A + S_B$. Furthermore, since for every realizable energy state in System A, System B retains its entire multiplicity, the multiplicity of the new system is given by $\Omega_C = \Omega_A \Omega_B$. Therefore, if entropy is related to this idea of degeneracy, it must

be related through a function which translates products of degeneracy into sums of entropy; that is, it must be logarithmic [38: pp. 112-116].

$$S = k \ln \Omega \tag{11}$$

This equation is known as the Boltzmann relation for the entropy. It is one of the most important relations from statistical mechanics because in one simple relation it links our microscopic (or quantum) understanding of the gas to our macroscopic understanding of entropy and traditional thermodynamics. The proportionality constant is chosen to be the Boltzmann number, $k = 1.38065 \times 10^{-23}$ (J/K) [38].

If one can determine the statistical multiplicity of a system, that is, the number of different ways the energy can be distributed over the particles and maintain the aggregate sum, equation (11) can be used to determine its entropy rather than using expressions involving the macroscopic properties of the gas. Using concepts from statistical mechanics, expressions for the statistical multiplicity may be derived.

2.2 Determination of the Statistical Multiplicity or Counting the Number of Possible Microstates

In statistical mechanics any possible permutation of particle energy states which is consistent with the overall energy of the system is called a *microstate* of the system. Determining the statistical multiplicity of the system is equivalent to counting the number of feasible microstates.

Before beginning this discussion, a distinction must be made regarding the types of particles under consideration. The *Pauli Exclusion Principle*, states that particles such as protons, electrons, and neutrons, which have half odd integral spin cannot occupy the same quantum state as another such particle[7, 18, 40]. This means that if the particles of interest are composed of an odd number of these elementary units, only one particle may occupy a given energy state. Such particles are called *fermions*, and the mathematics of determining the microstates is known as *Fermi-*

Dirac Statistics [40]. On the other hand, if the particles we are interested in are composed of an even number of elementary units, there is no limitation on the number of particles occupying a given quantum state. Such particles are called *bosons*, and the mathematics associated with them is *Bose-Einstein Statistics* [40]. It will be shown that the two methodologies converge when the number of available quantum states is much larger than the number of particles under consideration (since the probability of any two particles occupying the same state is very small under such conditions).

The first method outlined here for counting the number of microstates follows the treatment given in Vincenti and Krueger [38]. This method allows one to avoid counting the particles in every individual quantum level. This is especially useful for energy modes where the quantum levels are very closely spaced.

Consider dividing an energy spectrum into several energy groupings, each containing several quantum energy levels. We associate a characteristic energy, ϵ_j, with the j^{th} group, and stipulate that the size of each group should be such that the energy associated with each of the quantum states it contains should be negligibly close to ϵ_j. The system may be examined by considering how particles can be distributed amongst the various groups and how they can be distributed within a group. Denoting the number of particles in the j^{th} group by N_j, any arrangement of particles across the groups must obey the following,

$$\sum_j N_j = N \tag{12}$$

$$\sum_j N_j \epsilon_j = E \tag{13}$$

where N is the total number of particles in the system and E is the total energy of the system.

Denote the number of quantum states in group j by c_j. Consider the Bose-Einstein (BE) case. Here, there is no limit as to how many particles can occupy any state in group j. The total number of ways to arrange N_j objects into c_j bins is

easily verified to be $(N_j + c_j - 1)!$. However, this expression must be corrected for two factors. First, the N_j particles are indistinguishable. Second, the various states within the group are also indistinguishable. Therefore, the previous expression must be divided by a factor of $N_j!(c_j - 1)!$ to correct for these items. The total number of ways to arrange N_j bosons into c_j states is then given by w_j in equation (14).

$$(w_j)_{BE} = \frac{(N_j + c_j - 1)!}{N_j!\,(c_j - 1)!} \tag{14}$$

Each arrangement of particles over the groups is called a *macrostate*. Denoting this macrostate by the index i, and considering the contribution from each group, the total number of microstates available in this macrostate is,

$$(W_i)_{BE} = \prod_j (w_j)_{BE} = \prod_j \frac{(N_j + c_j - 1)!}{N_j!\,(c_j - 1)!} \tag{15}$$

Now consider the Fermi-Dirac (FD) case. Here one is limited to one particle per quantum state. Clearly, a further constraint is $c_j \geq N_j$. In this case, there are $c_j!/(c_j - N_j)!$ possible ways to arrange the particles. Again, this expression must be corrected for the fact that the particles are indistinguishable. The total number of ways to arrange N_j fermions into c_j states is therefore,

$$(w_j)_{FD} = \frac{c_j!}{(c_j - N_j)!N_j!} \tag{16}$$

Denoting this macrostate by the index i and taking into account each group, the total number of microstates in this macrostate becomes

$$(W_i)_{FD} = \prod_j (w_j)_{FD} = \prod_j \frac{c_j!}{(c_j - N_j)!N_j!} \tag{17}$$

Allowing for the existence of multiple energy groupings and macrostates, the total statistical multiplicity is found by summing over all macrostates consistent with

equations (12) and (13).

$$\Omega = \sum_i W_i \text{ for all } i \text{ such that } \sum_j N_j = N \text{ and } \sum_j N_j \epsilon_j = E \tag{18}$$

Traditionally, rather than calculate the total number of possible microstates due to every possible macrostate, an assumption often used is known include only the contribution of the *most probable* macrostate in the count [19]. This assumes that the most probable macrostate dominates the counting of microstates. Then, one can approximate $\Omega \approx W_{max}$ and can determine how the particles must be distributed to achieve this. This leads to the equilibrium distribution for the various energy modes. For the present application it is better to assume that we know how the particles are distributed in the current macrostate and proceed to determine Ω using the relations above. As the system approaches thermodynamic equilibrium, the distribution of particles over the states will tend to the most probable macrostate. Denoting the current macrostate of the system by W we will simply use $\Omega = W$. Then, since the entropy is related to $\ln \Omega$, the following expression applies for bosons.

$$\ln (W) = \sum_j \left[\ln (N_j + c_j - 1)! - \ln (c_j - 1)! - \ln (N_j)! \right] \tag{19}$$

and for fermions,

$$\ln (W) = \sum_j \left[\ln (c_j)! - \ln (c_j - N_j)! - \ln (N_j)! \right] \tag{20}$$

For sufficiently large arguments of the factorial, the above logarithms may be approximated using Stirling's formula [38, 40].

$$\ln (z)! \approx z \ln z - z \tag{21}$$

Applying this approximation to equation (19) and (20) one obtains,

$$\ln(W) \approx \sum_j \left[\pm c_j \ln\left(1 \pm \frac{N_j}{c_j}\right) + N_j \ln\left(\frac{c_j}{N_j} \pm 1\right)\right] \tag{22}$$

where the $(+)$ applies in the case of bosons and the $(-)$ for the case of fermions.

Application of the assumption that there are many more quantum states in the group than there are particles in the group ($c_j >> N_j$) is known as the Boltzmann limit. The Boltzmann limit is a good assumption provided the energy mode under consideration has closely spaced levels and is at a large enough temperature that a significantly large number of these levels are active [38]. Under this assumption equation (22) becomes,

$$\ln(W) \approx \sum_j N_j \left[1 + \ln\left(\frac{c_j}{N_j}\right)\right] \tag{23}$$

Since now there are many more states than particles, the probability of any two particles occupying the same state is extremely small, hence, particle independence has been attained.

In many cases it is not possible to group several energy levels as the quantum spacing is too large. Returning to this case, denote an individual quantum level by the subscript k. Suppose there are N_k particles in the k^{th} state. To determine the statistical multiplicity of the system, the question becomes, "How many different ways are there to arrange N identical particles so that the first state contains N_1 particles, and the second contains N_2 particles, and so on?" The answer is found in the *multinomial coefficient* using the same reasoning as above [40]. The total number of microstates is then given by the following.

$$W = \frac{N!}{N_1!N_2!N_3!\cdots} = \frac{N!}{\prod_k N_k!} \tag{24}$$

This term is actually a generalization of the binomial coefficient from algebra. Considering the natural logarithm of this term as before,

$$\ln(W) = \ln\left(\frac{N!}{\prod_k N_k!}\right) = \ln(N!) - \sum_k \ln(N_k!) \tag{25}$$

Applying Sterling's approximation one obtains the following simplified expression.

$$\ln W = -N \sum_k \frac{N_k}{N} \ln\left(\frac{N_k}{N}\right) \tag{26}$$

In some cases an energy mode may be degenerate; that is, there may be multiple quantum states which have the same energy level. In this case, denoting the degeneracy of the k^{th} state by g_k, equation (26) becomes

$$\ln W = -N \sum_k g_k \frac{N_k}{N g_k} \ln\left(\frac{N_k}{N g_k}\right) = -N \sum_k \frac{N_k}{N} \ln\left(\frac{N_k}{N g_k}\right) \tag{27}$$

Equations (26) and (27) allow the examination of modes which have widely spaced levels. Equation (23) provides a simplification for modes with levels which are more closely spaced in that not every energy level need be tabulated. To utilize any of these equations, expressions which define how the particles are distributed across the energy spectrum must be determined, as well as expressions which define how the quantum states are distributed across the spectrum. The former is handled rather simply through the use of energy distribution functions. The latter is handled using quantum mechanics.

2.3 Description of the System using Energy Distribution Functions

One can describe how the particles are distributed across an energy spectrum through the use of energy distribution functions. Here discrete functions will be used for modes with more widely spaced levels, while those with more closely spaced levels utilize continuous distributions. In the continuous case, define the energy distribution

function, f, to be a normalized probability distribution function (PDF), such that $f(\epsilon)d\epsilon$ is the probability of a particle having energy in the range $[\epsilon, \epsilon + d\epsilon]$. The normalization condition is then,

$$\int_0^\infty f(\epsilon)d\epsilon = 1 \tag{28}$$

This distribution is defined in the usual sense of a probability distribution, namely we can define the expectation value of any property, Q, which is dependent upon the argument of the PDF as

$$\langle Q \rangle = \int_0^\infty Q(\epsilon) f(\epsilon) d\epsilon \tag{29}$$

In the discrete case, define $f_k = N_k/N$, where N_k is the number of particles in the k^{th} quantum state. Further define a function α as follows.

$$\alpha(\chi) = \begin{cases} 1 & \text{if } \chi = 0 \\ 0 & \text{if } \chi \neq 0 \end{cases} \tag{30}$$

Then any discrete energy distribution function can be defined as,

$$f(\epsilon) = f_1 \alpha(\epsilon - \epsilon_1) + f_2 \alpha(\epsilon - \epsilon_2) + \cdots = \sum_k f_k \alpha(\epsilon - \epsilon_k) \tag{31}$$

In the case of the discrete distribution, the probability of a particle having energy ϵ is given by $f(\epsilon)$. As in the continuous case one can define an expectation value in the discrete case.

$$\langle Q \rangle = \sum_k Q(\epsilon_k) f(\epsilon_k) = \sum_k Q(\epsilon_k) f_k \tag{32}$$

Acknowledging the presence of different modes of energy storage in the particle, one can define a total PDF for the system. Supposing there to be m continuous energy modes and n discrete energy modes, one can define the probability of a particle occupying the incremental element of energy space defined by the various energy

modes as

$$f\left(\epsilon_1, \epsilon_2, \ldots, \epsilon_m, \epsilon'_1, \epsilon'_2, \ldots, \epsilon'_n\right) \partial\epsilon_1 \partial\epsilon_2 \cdots \partial\epsilon_m \tag{33}$$

where the $(')$ notation denotes the discretized quantities. If the energy modes are independent of one another, an individual PDF can be formed for each mode, and the product of the individual PDFs will yield the overall PDF.

$$f\left(\epsilon_1, \epsilon_2, \ldots, \epsilon_m, \epsilon'_1, \epsilon'_2, \ldots, \epsilon'_n\right) = \prod_{i=1}^{m} f_i\left(\epsilon_i\right) \prod_{j=1}^{n} f_j\left(\epsilon'_i\right) \tag{34}$$

If the modes are not strictly independent, but it is desired to model them as such, the individual PDF for a continuous mode can be found by integrating over all of the other continuous modes and summing over all of the discrete modes.

$$\begin{aligned} f_j\left(\epsilon_j\right) = \int_{\epsilon_1} \cdots \int_{\epsilon_{j-1}} \int_{\epsilon_{j+1}} \cdots \int_{\epsilon_m} \sum_{\epsilon'_1} \cdots \sum_{\epsilon'_n} \\ f\left(\epsilon_1, \ldots, \epsilon_m, \epsilon'_1, \ldots, \epsilon'_n\right) \partial\epsilon_1 \cdots \partial\epsilon_{j-1} \partial\epsilon_{j+1} \cdots \partial\epsilon_m \end{aligned} \tag{35}$$

The same procedure can be applied to a discrete mode.

$$f_j\left(\epsilon_j\right) = \int_{\epsilon_1} \cdots \int_{\epsilon_m} \sum_{\epsilon'_1} \cdots \sum_{\epsilon'_{j-1}} \sum_{\epsilon'_{j+1}} \cdots \sum_{\epsilon'_n} f\left(\epsilon_1, \ldots, \epsilon_m, \epsilon'_1, \ldots, \epsilon'_n\right) \partial\epsilon_1 \cdots \partial\epsilon_m \tag{36}$$

It should be noted that the product of the separated modes will only represent the true PDF if the modes are statistically independent. This is not always strictly the case. If the energy modes are coupled, separation into component distributions will not necessarily be an accurate representation of the system. In the case of a diatomic molecule, this is true of the rotational and vibrational energy modes. The centripetal acceleration of rotation acts as a forcing function to the vibrational mode, and the variable displacement due to the vibrational mode causes the rotational inertia to vary. However, the coupling is fairly weak until the separation distance between the atoms becomes large. The present work assumes that these modes can be isolated.

This assumption is not strictly necessary and is made only to simplify the analysis applied to the DSMC data.

Having an individual distribution function defined is exactly equivalent to knowing the distribution of the particles over the energy spectrum. N_j in equation (23) can be expressed as

$$N_j = Nf(\epsilon)d\epsilon \tag{37}$$

Further, N_k in equation (26) is given by,

$$N_k = Nf\left(\epsilon_k'\right) = Nf_k \tag{38}$$

2.4 Distribution of the Quantum Energy Levels

An important postulate from quantum mechanics is that there exists a wavefunction, Ψ, which describes each energy mode a particle may posses. Another postulate states that for every classical quantity (momentum, position, energy, etc.), there exists an operator that when applied to the wavefunction yields the respective classical quantity [7]. The Hamiltonian of classical mechanics describes the energy content of a system [28] and is given in equation (39) below.

$$H = \frac{P^2}{2m} + V \tag{39}$$

where p is the momenta and V is the potential energy. If one replaces the terms of the Hamiltonian with their respective quantum mechanical operators and applies the resulting operator to the wavefunction, one obtains the Schrödinger equation, shown in its time independent form below [7].

$$-\frac{\hbar^2}{2m}\nabla^2\Psi\left(\vec{r}\right) + V\left(\vec{r}\right)\Psi\left(\vec{r}\right) = \epsilon\Psi\left(\vec{r}\right) \tag{40}$$

Here V is the potential field in which the particle exists, and $\hbar = h/2\pi$ where h is the Planck constant. This represents a partial differential equation in three spatial vari-

ables for the wavefunction Ψ. Depending upon the form of the potential, this equation may or may not be separable, and the solution process becomes complicated. A case where separation of variables is possible is the case of a particle simply translating in a box. In this case, $V = 0$ and a solution of equation (40) is readily obtained. It is also desirable to examine the rotational and vibrational behavior of the particles. In the rotational case, a change of reference frame to spherical polar coordinates is applied, and the solution process is somewhat laborious. Likewise, the existence of a spring-like potential complicates the vibrational case. Solution for the energy distributions in these cases is possible, though a full treatment requires a more rigorous discussion of quantum mechanics than is warranted here. For this reason, only the translational case is presented in detail. Results for the rotational and vibrational case will follow this derivation, along with a discussion of limitations for the models used.

Consider a particle whose only energy mode is translation moving in a cube of length L. The particle cannot exist outside of the cube, hence Ψ is zero on the surfaces of the cube. This problem may be solved using separation of variables. Let

$$\Psi(x, y, z) = \Psi_1(x)\,\Psi_2(y)\,\Psi_3(z) \tag{41}$$

Substituting into equation (40) and performing some intermediate steps, the partial differential equation reduces to the following three ordinary differential equations.

$$\begin{aligned}
\frac{\hbar^2}{2m}\frac{d^2\Psi_1}{dx^2} + \epsilon_1\Psi_1 &= 0 \\
\frac{\hbar^2}{2m}\frac{d^2\Psi_2}{dy^2} + \epsilon_2\Psi_2 &= 0 \\
\frac{\hbar^2}{2m}\frac{d^2\Psi_3}{dz^2} + \epsilon_3\Psi_3 &= 0
\end{aligned} \tag{42}$$

Since all three have identical form, consider the solution of the ordinary differential equation,

$$f''(\alpha) + \frac{2m\epsilon}{\hbar^2} f(\alpha) = 0 \tag{43}$$

subject to,

$$\begin{aligned} f(0) &= 0 \\ f(L) &= 0 \end{aligned} \tag{44}$$

It is easily verified that the family of solutions which satisfies equation (43) is given by,

$$f(\alpha) = \mathcal{A} \sin\left(\sqrt{\frac{2m\epsilon}{\hbar^2}}\alpha\right) + \mathcal{B} \cos\left(\sqrt{\frac{2m\epsilon}{\hbar^2}}\alpha\right) \tag{45}$$

To satisfy the first boundary condition, choose $\mathcal{B} = 0$. Then, since f should be non-trivial, ϵ must be chosen to satisfy the second boundary condition. This yields,

$$\epsilon = \frac{n^2 h^2}{8mL^2}, \quad n = 1, 2, \ldots \tag{46}$$

and hence,

$$f(\alpha) = \mathcal{A} \sin\left(\frac{n\pi\alpha}{L}\right) \tag{47}$$

Equation (41) then becomes

$$\Psi_{tr}(x, y, z) = \mathcal{A} \sin\left(\frac{n_1\pi x}{L}\right) \sin\left(\frac{n_2\pi y}{L}\right) \sin\left(\frac{n_3\pi z}{L}\right) \tag{48}$$

and the total translational energy is given by

$$\epsilon_{tr} = \epsilon_1 + \epsilon_2 + \epsilon_3 = (n_1^2 + n_2^2 + n_3^2)\,\tfrac{h^2}{8mL^2}, \quad n_1, n_2, n_3 = 1, 2, \ldots \tag{49}$$

If the precise form of the wavefunction is needed, one can determine the multiplicative constant via the normalization condition [7],

$$\int_{-\infty}^{\infty} \Psi^* \Psi d\vec{r} = 1 \tag{50}$$

where the asterisk denotes the complex conjugate. For present purposes only a knowledge of the energy levels is required. This was obtained in terms of three quantum numbers in equation (49).

In the present research only monatomic and diatomic molecules were examined. For the diatomic case, various quantum models for a two particle system can be used to describe the rotational and vibrational motion. For the rotational mode, the two atoms of the molecule can be treated as if they were connected by a rigid link, and rotational motion about the system center of mass is considered. This is known as the rigid rotator model. Under this assumption, expressions for the quantitized angular momentum and energy can be derived using equation (40) and applying various other tools from quantum mechanics. In this case, the energy levels are described by equation (51) [7, 18, 38].

$$\epsilon_{rot} = \frac{\hbar^2}{2\mu r_e^2} n_{rot} \left(n_{rot} + 1\right), \quad n_{rot} = 0, 1, 2, \ldots \tag{51}$$

where r_e is the equilibrium separation distance of the atoms and μ is the reduced mass of the system. It should be noted that the rotational energy levels are degenerate. The degeneracy of the k^{th} rotational level is given by $g_k = 2k + 1$ [38].

The vibrational mode may be handled by assuming that the atoms oscillate along their line of centers harmonically. This means that the potential operator discussed above is of the form $V(r) = \dfrac{m\nu^2}{8\pi^2} r^2$, where ν is the oscillating frequency of the molecule. This is known as the harmonic oscillator model. A one-dimensional form of equation (40) can be solved along a line connecting the two particles. In this case, the energy levels are given by the following. [7, 38].

$$\epsilon_{vib} = h\nu \left(n_{vib} + \tfrac{1}{2}\right), \quad n_{vib} = 0, 1, 2, \ldots \tag{52}$$

Strictly, neither the rotational or vibrational models discussed here truly represent the full dynamics involved in a diatomic molecule. In reality, the two modes are

coupled, as was discussed previously. However, until the vibrational mode becomes significantly excited and the internuclear displacement becomes fairly large, capturing the coupling effect is not crucial. Another inaccuracy arises in the modeling of the potential as parabolic (harmonic) in the vibrational case. This can be seen in Figure 3 below. The actual potential experienced by the particles is not symmetric as modeled, but rather diminishes as the particles move out past their equilibrium spacing. Over the lower portion of the potential well the parabola provides a decent fit to reality. However, a greater harm is done by the fact that the parabolic model leads to a prediction of equally spaced energy levels as seen in equation (52). In reality the first few levels are relatively of the same size, but the higher energy levels tend to mass together as the dissociation energy level is approached. This will lead to some inaccuracy in determining the energy level of a molecule which has a relatively large vibrational energy.

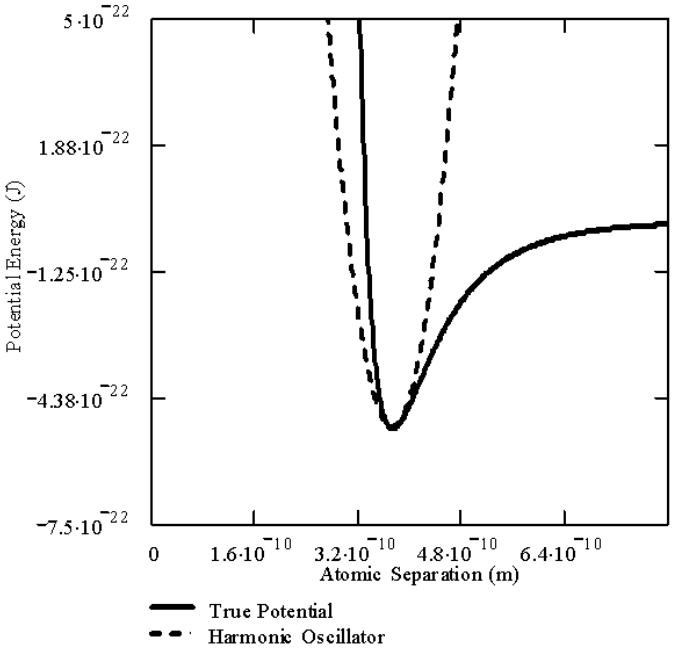


Figure 3: Diatomic Potential Functions

It is possible to examine both the coupling effect and the asymmetric potential through the use of a model known as the nonrigid rotator, anharmonic oscillator. This

model is more accurate, but also slightly more complex. For the basic research presented here, the rigid rotator and harmonic oscillator models should provide adequate results, so long as the vibrational mode is not highly excited.

2.5 Entropy Formulation in Terms of Expectation Quantities

Using the results of the previous two sections, it is possible to determine $\ln \Omega$ from equation (23) or (27). However, the case where energy states were grouped requires specification of how the groups are to be chosen. Specifically, the size of the groups needs to be specified so that the number of states and particles in the groups may be calculated from the previous results.

A natural choice for the groups is the interval $[\epsilon, \epsilon + d\epsilon]$, in reference to the energy distribution functions. Equation (37) immediately yields the number of particles in this interval. The number of quantum states in this interval must also be determined. The approach taken here follows Vincenti and Krueger [38: pp. 97, 125].

Denote the number of states below some energy level ϵ by $\Gamma(\epsilon)$. Then the number of states contained in the interval can be determined via the following expression.

$$c_j = \frac{d\Gamma(\epsilon)}{d\epsilon} d\epsilon \tag{53}$$

To use equation (53), an expression for Γ is required. This will follow directly from the energy levels previously derived. Begin by considering the number of translational states below ϵ. From equation (49),

$$\left(n_1^2 + n_2^2 + n_3^2\right) \frac{h^2}{8mL^2} \leq \epsilon \tag{54}$$

Equation (54) is the expression for the first octant of a sphere of radius $R = \frac{2L}{h}\sqrt{2m\epsilon}$ in the space defined by the quantum numbers n_1, n_2, n_3. The number of states below

ϵ can be found as the volume of that octant.

$$\Gamma_{tr}(\epsilon) = \frac{4\pi V}{3h^3}(8m\epsilon)^{3/2} \tag{55}$$

Hence, the number of states contained in $[\epsilon, \epsilon + d\epsilon]$ is given by,

$$(c_j)_{tr} = \frac{d\Gamma_{tr}}{d\epsilon}d\epsilon = \frac{4\pi V}{h^3}\sqrt{2m^3\epsilon}d\epsilon \tag{56}$$

Equation (56), along with equation (37), may be used in equations (23) and (11) to determine the appropriate expression for the translational entropy contribution. Having expressed the energy groups in terms of the infinitesimal $d\epsilon$, the former summation is now taken in the limit to be the integral over the entire spectrum. This yields

$$S_{tr} = kN\int_0^\infty f_{tr}(\epsilon)\left[1 - \ln\left(\frac{h^3 n f_{tr}(\epsilon)}{4\pi\sqrt{2m^3\epsilon}}\right)\right]d\epsilon \tag{57}$$

Alternatively, we can express the translational entropy in terms of the velocity distribution function, rather than the translational energy distribution function by using the following relation [38: p. 340].

$$\sqrt{\epsilon}d\epsilon = \frac{1}{2\pi}\left(\frac{m}{2}\right)^{3/2}\partial\vec{c} \tag{58}$$

In which case the entropy is given by

$$S_{tr} = kN\int_{-\infty}^{\infty} f(\vec{c})\left[1 - \ln\left(\frac{h^3 n f(\vec{c})}{m^3}\right)\right]\partial\vec{c} \tag{59}$$

where the $\partial\vec{c}$ denotes that the integration is performed on all three components of velocity over the entire velocity space.

In the present study, only the translational mode will be examined using the closely spaced energy level formulation. The Boltzmann limit typically does not hold when the vibrational mode is examined by itself, as for example, in nitrogen

there are less than forty harmonic oscillator energy states before the dissociation level is reached. Only a few of these states are active until very high temperatures are attained. The rotational mode possesses many more levels than the vibrational mode, but not as many as the translational mode. For these reasons, both the rotational and vibrational modes were treated using the discrete formulations developed in the previous sections. Recalling equations (26) and (11) the entropy contribution from these modes becomes,

$$S_{rot} = -kN \sum_i f_{rot,i} \ln \left(\frac{f_{rot,i}}{g_{rot,i}} \right) \tag{60}$$

$$S_{vib} = -kN \sum_i f_{vib,i} \ln \left(f_{vib,i} \right) \tag{61}$$

Recalling the definition of an expectation value from equations (29) and (32) these entropy formulas can be written more succinctly in terms of expectation values of various quantities involving the distribution functions. Specifically,

$$\begin{aligned} S_{tr} &= \left\langle kN \left[1 - \ln \left(\frac{h^3 n f_{tr}(\epsilon)}{4\pi\sqrt{2m^3\epsilon}} \right) \right] \right\rangle = \left\langle kN \left[1 - \ln \left(\frac{h^3 n f(\vec{c})}{m^3} \right) \right] \right\rangle \\ S_{rot} &= \left\langle -kN \ln \left[\frac{f_{rot}(\epsilon)}{g_{rot}(\epsilon)} \right] \right\rangle \\ S_{vib} &= \left\langle -kN \ln f_{vib}(\epsilon) \right\rangle \end{aligned} \tag{62}$$

Or, using the definition of the specific gas constant, $R = k/m$, we have on a per mass basis,

$$\begin{aligned} s_{tr} &= \left\langle R \left[1 - \ln \left(\frac{h^3 n f_{tr}(\epsilon)}{4\pi\sqrt{2m^3\epsilon}} \right) \right] \right\rangle = \left\langle R \left[1 - \ln \left(\frac{h^3 n f(\vec{c})}{m^3} \right) \right] \right\rangle \\ s_{rot} &= \left\langle -R \ln \left[\frac{f_{rot}(\epsilon)}{g_{rot}(\epsilon)} \right] \right\rangle \\ s_{vib} &= \left\langle -R \ln f_{vib}(\epsilon) \right\rangle \end{aligned} \tag{63}$$

Again, the expectation quantities for the rotational and vibrational modes are defined in the discrete formulation given by equation (32), whereas the expectation quantity in the translational case is defined in the continuous formulation given by equation (29).

The translational entropy defined above is related to a concept known as Boltzmann's H-Theorem. Boltzmann defined a function $\mathcal{H}$ as the following [8: pp.49-62][14: p.67].

$$\mathcal{H}(t) = \int_{-\infty}^{\infty} f\left(\vec{c}, t\right) \ln\left(f\left(\vec{c}, t\right)\right) \partial\vec{c} \tag{64}$$

Boltzmann was able to show that this function is always monotonically decreasing. This is known as Boltzmann's H-Theorem. He reasoned that this quantity had to be related to the entropy in the gas. This is seen to be true from the above expression for the translational entropy. Namely, the entropy is an affine function of $\mathcal{H}$. In many derivations in which only the change in entropy is important, the entropy may be defined as $S = -k\mathcal{H}$. Since the derivation above is directly related to the H-theorem, it is consistent with the second law of thermodynamics, namely that entropy is a monotonically increasing quantity in an isolated system.

2.6 Calculating the Entropy Generation Rate Using Kinetic Theory

The Boltzmann equation from kinetic theory governs the evolution of the velocity distribution function of a gas by tracking particle motions into and out of the six-dimensional phase space defined by $\partial\vec{r}\partial\vec{c} = \partial x \partial y \partial z \partial u_x \partial u_y \partial u_z$ and is given as follows.

$$\frac{\partial(nf)}{\partial t} + \vec{c} \cdot \nabla\left(nf\right) + \vec{F} \cdot \frac{\partial(nf)}{\partial\vec{c}} = \int_{-\infty}^{\infty} \int_{0}^{4\pi} n^2 \left(f\left(z'\right) f\left(c'\right) - f(z) f(c)\right) \vec{g}\sigma \partial\Omega\partial\vec{c} \tag{65}$$

where n is the number density of the gas, and $\vec{F}$ is an external force acting on the particles. The right hand side of this equation is termed the *collision integral*. It accounts for particles being knocked into or out of the phase space by binary collisions with other particles. The Boltzmann equation as traditionally formulated

makes provision only for binary collisions, as tertiary and higher order collisions are quite rare. $\vec{c}$ and $\vec{z}$ are the molecular velocities of two particles involved in a collision, and the ($'$) denotes a post collisional quantity. The relative velocity of the two particles is denoted by $\vec{g}$, and $\sigma\partial\Omega$ is the collisional cross section. The first term on the left side of equation (65) represents the accumulation of particles in the phase space. The second term represents the change in the number of particles in the phase space due to convection into or out of $\partial\vec{r}$ by $\vec{c}$. The third term accounts for changes in the number of particles in the phase space due to acceleration into or out of $\partial\vec{c}$ by the external force.

Equation (65) holds generally for simple particles where transfer to the internal energy modes is not important. When the particles possess internal energy, this equation can be generalized to account for energy transfer to these modes. Let $\vec{Q}$ represent a vector of some number of generalized coordinates and let $\vec{P}$ represent a vector of some number of generalized momenta. Then the generalized Boltzmann equation which describes particle motions into and out of the phase space $\partial\vec{Q}\partial\vec{P}$ is given as follows [14: pp. 199-201].

$$\frac{\partial(nf)}{\partial t} + \dot{\vec{Q}} \cdot \frac{\partial\,(nf)}{\partial \vec{Q}} + \dot{\vec{P}} \cdot \frac{\partial(nf)}{\partial \vec{P}} = \int_{\vec{\Lambda}} \int_{-\infty}^{\infty} \int_{0}^{4\pi} n^2 \left(f_1' f' - f_1 f\right) \vec{g}\sigma\partial\Omega\partial\vec{c}\partial\vec{\Lambda} \qquad (66)$$

where $\vec{\Lambda}$ represents all of the internal components of $\vec{Q}$. f and f_1 now represent the entire distribution functions for two particles involved in a collision. The second term on the left side of the equation now represents the convection of particles into or out of $\partial\vec{Q}$ by $\dot{\vec{Q}}$. The third term on the left is analogous to the forcing term in equation (65) in that it represents the acceleration of particles into or out of $\partial\vec{P}$ by $\dot{\vec{P}}$.

The energy modes of interest in the current study are translation, rotation, and vibration. Each have associated momentum. For the rotational mode, three orientation angles could be considered as three extra generalized coordinates. For the vibrational mode, the internuclear separation could be considered as another generalized coordinate, bringing the total number of generalized coordinates for a

diatomic molecule to seven. Likewise, allowing for three components of rotational momentum and one for the motion along the line of centers from vibration, the number of generalized momentum components is also seven. This brings the total dimension of the phase space $\partial\vec{P}\partial\vec{Q}$ to fourteen. This illustrates the difficulty associated with a purely mathematical approach. Fortunately, it is possible to simplify the situation to fit present purposes.

In consideration of entropy, no knowledge of the internal generalized coordinates is required. Therefore $\vec{Q}$ can be replaced with $\vec{r}$ in equation (66) realizing that knowledge of the internal coordinates has been lost. For present purposes, there is no external forcing which will selectively act upon the internal modes; further, assume there is no external force such that $\dot{\vec{P}} = \vec{F} = 0$. Then the generalized Boltzmann equation reduces to the following.

$$\frac{\partial(nf)}{\partial t} + \vec{c}\cdot\nabla(nf) = \int_{\vec{\Lambda}}\int_{-\infty}^{\infty}\int_{0}^{4\pi} n^2\left(f(z')f(c') - f(z)f(c)\right)\vec{g}\sigma\partial\Omega\partial\vec{c}\partial\vec{\Lambda} \qquad (67)$$

It is readily seen that the only difference between this equation and the regular Boltzmann equation (65) is the collision integral, which now accounts for the effect of collisions on the internal energy modes. This is to be expected, as in the absence of external forcing the only mechanism which affects the internal modes is collisions with other particles.

Collision integrals are, in general, extremely difficult to evaluate and are responsible for the majority of the difficulty associated with solving the Boltzmann equation. This problem is amplified when the internal modes are brought into the picture. Fortunately, evaluation of such terms is not required when DSMC is employed. The collision physics simulated in the DSMC process allows one to examine the collisional contributions without actually using the collision integral.

Taking the moment of equation (67) with respect to Q/n where Q is some function of the arguments of the distribution function yields the Boltzmann transport

equation.

$$\begin{aligned}\frac{\partial}{\partial t}\left[\int_{\vec{\Lambda}}\int_{-\infty}^{\infty} n\frac{Q}{n}\partial\vec{\Lambda}\partial\vec{c}\right]+\int_{\vec{\Lambda}}\int_{-\infty}^{\infty}\frac{Q}{n}\vec{c}\cdot\nabla\left(nf\right)\partial\vec{\Lambda}\partial\vec{c}\\ =\int_{\Lambda}\int_{-\infty}^{\infty}\int_{0}^{4\pi}\frac{Q}{n}n^2\left(f_1'f'-f_1f\right)\vec{g}\sigma\partial\Omega\partial\vec{c}\partial\vec{\Lambda}\end{aligned} \tag{68}$$

This expression can be written more succinctly by recalling the definition of an expectation value. Simplifying, an expression governing the transport of $\langle Q\rangle$ throughout the fluid is obtained.

$$\frac{\partial}{\partial t}\langle Q\rangle+\nabla\cdot\langle\vec{c}Q\rangle=\Delta\left[Q\right] \tag{69}$$

where $\Delta[Q]$ represents the net change in Q due to collisions.

Recall that entropy can be written in terms of expectation values. Substituting appropriately for Q, equation (69) becomes an expression defining the entropy generation in the gas. Specifically, consider the generation of entropy density, $\tilde{s}=\rho s$.

$$\frac{\partial\tilde{s}}{\partial t}+\nabla\cdot\vec{\mathcal{T}}=\Delta\left[\tilde{s}\right]\equiv\dot{\tilde{s}} \tag{70}$$

Where $\vec{\mathcal{T}}$ is the entropy density flux vector. This expression can be applied to the various entropy forms discussed in the previous section to determine the entropy generation rate of each of the individual modes. Presuming the modes to be separable as previously discussed, the various entropy density and flux vectors are given in terms of expectation values below.

$$\begin{aligned}\tilde{s}_{tr}&=\left\langle kn\left[1-\ln\left(\tfrac{h^3nf(\vec{c})}{m^3}\right)\right]\right\rangle\\ \tilde{s}_{rot}&=\left\langle -kn\ln\left[\frac{f_{rot}\left(\epsilon\right)}{g_{rot,i}\left(\epsilon\right)}\right]\right\rangle\\ \tilde{s}_{vib}&=\left\langle -kn\ln f_{vib}\left(\epsilon\right)\right\rangle\end{aligned} \tag{71}$$

$$\vec{\mathcal{T}}_{tr} = \begin{bmatrix} \left\langle c_x kn \left[1 - \ln\left(\frac{h^3 nf(\vec{c})}{m^3}\right)\right]\right\rangle \\ \left\langle c_y kn \left[1 - \ln\left(\frac{h^3 nf(\vec{c})}{m^3}\right)\right]\right\rangle \\ \left\langle c_z kn \left[1 - \ln\left(\frac{h^3 nf(\vec{c})}{m^3}\right)\right]\right\rangle \end{bmatrix} \quad \vec{\mathcal{T}}_{rot} = \begin{bmatrix} u_x \tilde{s}_{rot} \\ u_y \tilde{s}_{rot} \\ u_z \tilde{s}_{rot} \end{bmatrix} \quad \vec{\mathcal{T}}_{vib} = \begin{bmatrix} u_x \tilde{s}_{vib} \\ u_y \tilde{s}_{vib} \\ u_z \tilde{s}_{vib} \end{bmatrix} \tag{72}$$

These quantities, when substituted into equation (70), will yield the net entropy generation due to nonequilibrium occurring in each mode. Since the expressions for the entropy of the internal modes exhibit no explicit dependence on the molecular velocity, the corresponding flux vectors are those traditionally expected. The translational mode does contain an explicit dependence on the molecular velocity and must be left in terms of expectation quantities.

This chapter has outlined a means of computing the entropy generation throughout the gas without using any macroscopic quantities or making any inherent equilibrium assumptions. This is precisely what is needed to examine nonequilibrium phenomena. To obtain these relations several ideas from statistical mechanics, quantum mechanics and kinetic theory were tied together; but, in the end, the expressions remain in terms of readily computable quantities from DSMC. The next chapter will deal with how to compute these quantities.

III. Numerical Methods and Implementation

This chapter will focus on the numerical methods used to implement the entropy and entropy generation calculations discussed in the previous chapter. A detailed discussion of the numerics will be provided, along with short descriptions of the actual coding used and modifications made to the original program. The normal shock problem is introduced and a description of the simulation details is given. A brief overview is also presented detailing how the one-dimensional Navier-Stokes equations were integrated to obtain results for this problem. Before beginning this discussion, it is useful to include a brief introduction to the DSMC code which was used.

3.1 Overview of MONACO

MONACO is a two-dimensional/axisymmetric DSMC code developed by the research group of Iain Boyd at the University of Michigan. MONACO permits the use of both structured and unstructured grids, and was written in a manner to provide for efficient parallelization of the DSMC algorithm. The default particle collision model used is the Variable Hard Sphere (VHS), although alternatively the Variable Soft Sphere (VSS) is available. MONACO includes routines to simulate energy transfer to internal energy modes, and also has the means of including chemical reactions in the simulation.[24, 10]

3.2 Formation and Calculation of Probability Distributions

Perhaps the most important assumption of the preceding chapter was the availability of distribution functions describing the energy state of the system. Such functions are calculable based upon DSMC data and a discussion of how such functions were calculated is presented here.

Although, in general, a distribution function may vary in space and time (as well as with the distribution variable), it has been assumed that within any given cell the distribution functions do not vary spatially. This introduces a dependence on the grid size as to how accurately changes in the distribution function are modeled. In

DSMC the cell size is typically chosen as a fraction of the mean free path, so that this assumption is fairly good. Furthermore, the temporal accuracy of the distribution function will depend on the time step used. The larger issue of accuracy in the distribution function comes from the number of particles used to generate it. If every particle in the flow could be simulated, the distribution function calculated in a given cell would be very close to the actual distribution. Unfortunately, only a fraction of the actual particles may be simulated, each of which is assigned a weight or number of actual particles it represents. Therefore, when the properties of a simulated particle are recorded for the generation of a distribution function, these values are taken to represent the properties of some larger number of particles. For this reason, as the statistical weight of the particles is reduced, the distribution function will become more representative of reality.

Alternatively, in steady flows the distribution functions do not change with time. This allows one to incorporate data from multiple (even several thousand) time steps in the creation of the probability distribution. Over time as more and more particles enter and leave a cell, each time contributing information to the creation of the distribution function. This has the same effect of reducing the statistical weight. This method is employed exclusively to the macroscopic variables in the DSMC process, which are in fact averaged over several thousand time steps to reduce the statistical scatter in the data. Without doing this, the level of scatter in an instantaneous DSMC result would prohibit it from being very useful at all. Like the distribution functions, this scatter may be reduced if more particles are simulated. The present study employed the method of sampling over many time steps in creating the distribution function since the standing normal shock is a steady state problem.

Although this is the case, the initial phases of the DSMC process are not steady. There are a number of transients associated the simulation process and the evolution of the flow to a steady state. For this reason, data should not be sampled for use in distribution functions until well after the flow has developed. This too is standard

practice in DSMC, and typically several thousand simulations may proceed before data is sampled for use.

3.2.1 Calculation of Velocity Distribution Functions. To calculate the translational contribution to the entropy, a valid velocity or translational energy distribution function is required. Here, the velocity distribution was chosen because of the availability of the computed data. In the DSMC process, the velocity components of each simulated particle are calculated at each simulation step. This data was used to create a distribution function of the following form.

$$f(\vec{c}) = f_x(c_x) f_y(c_y) f_z(c_z) \tag{73}$$

That is, it was assumed that the velocity distribution could be separated into a product of the component distributions.

Since, in the computational sense, it is impossible to compute and store the value of the distribution function over the entire velocity spectrum, it is required that the domain over which the function is computed be restricted. That is, upper and lower bounds on velocity for which the function is actually calculated must be defined. Outside of these bounds, the function is presumed to take on a value of zero. When the functions are generated instantaneously, these bounds may be set by the maximum and minimum velocity of particles in the cell. If data from multiple simulation steps are used, the bounds should be set before sampling in order to provide a consistent data structure. However, the maximum and minimum velocities that will be observed in the cell cannot be determined *a priori*. Therefore, rather than specify upper and lower bounds, the user chooses the number of standard deviations away from the mean to be included in the distribution. Both the mean and mean square velocities are sampled in the DSMC process. After the flow has reached steady state, these values may be used to set up the distribution function limits.

The distribution functions calculated for each of the velocity components are discrete. The user is allowed to specify the number of subintervals to include in the distribution. These subintervals are set up with constant width given by,

$$\delta c_i = \frac{c_{i,max} - c_{i,min}}{n_{int}} \tag{74}$$

where $c_{i,max}$ and $c_{i,min}$ represent the upper and lower limits of the distribution, and n_{int} represents the number of subintervals. Within each of these subintervals the distribution function is assumed to be constant. One would think that as the number of subintervals increases, or equivalently as the width of the subintervals decreases, a better representation of the distribution function would be obtained. This is, in fact, only partially true. If every particle could be tracked continuously over time this would be the case. Unfortunately, because the DSMC process only simulates a fraction of the particles over a finite number of time steps, the interval size may be chosen too small so as to cause the the distribution function to experience large jumps in value between consecutive subintervals. This is because the probability of observing a particle with velocity in a smaller subinterval is smaller than for a larger subinterval. If one could sample for a very long time these fluctuations would eventually die out, but this is not reasonable to expect. These fluctuations are nonphysical, and since entropy is an integral function of the distribution value it would be fairly sensitive to these fluctuations. Furthermore, the computational time is a function of the number of intervals included, and the entropy calculations require multiple distribution functions to be computed in each cell. It was found that a good balance of these competing factors was achieved when the number of subintervals per standard deviation was around 10-20.

At each simulation step where a velocity distribution is to be computed, the number of particles having a velocity in each subinterval is counted. The value of the

distribution function in the j^{th} subinterval is then given by equation (75).

$$f_j = \frac{N_j \mathcal{W}}{N \mathcal{W} \delta c} = \frac{N_j}{N \delta c} \tag{75}$$

It is readily verified that the normalization condition on f, namely that the zeroth moment be unity, is satisfied when f is defined as above.

$$\int_{-\infty}^{\infty} f(c) dc = \sum_{j=1}^{n_{int}} f_j \delta c = \sum_{j=1}^{n_{int}} \frac{N_j}{N \delta c} \delta c = \frac{N}{N} = 1 \tag{76}$$

When data from multiple time steps are used, it is required to keep a running total of the number of particles that have been used in creating the distribution. Denoting this quantity by N_{PDF}, the function value is computed on an update step as follows.

$$\begin{aligned} f_j^m &= \frac{f_j^{m-1} N_{PDF}^{m-1} + N_j}{N_{PDF}^{m-1} + N} \\ N_{PDF}^m &= N_{PDF}^{m-1} + N \end{aligned} \tag{77}$$

where the superscript m is used to denote values at the present time step.

3.2.2 Calculation of Rotational and Vibrational Energy Distribution Functions.

As stated in the precious chapter, a different approach to the calculation of the rotational and vibrational distributions was taken. Namely, the contribution of each quantum level was computed individually. In this case, the number of levels to include is specified by the user. Like the velocity distribution, these distributions are assumed to have a value of zero outside of this domain.

The energy of each level is calculated through the use of the rigid rotator and harmonic oscillator models. One cannot expect the DSMC process to exactly reproduce the energy calculated by these models, so these energy states are taken to be the midpoints of connected subintervals on the energy spectrum. For the rotational mode, the levels are unequally spaced so the upper and lower limits of the j^{th} level

were computed as the following.

$$\epsilon_{j,max} = \frac{\epsilon_j + \epsilon_{j+1}}{2} \tag{78}$$

$$\epsilon_{j,min} = \frac{\epsilon_{j-1} + \epsilon j}{2} \tag{79}$$

where ϵ_j is the j^{th} rigid rotator energy level. Then the width of each interval varies and is given by,

$$\delta\epsilon_j = \frac{\epsilon_{j+1} - \epsilon_{j-1}}{2} \tag{80}$$

The energy levels of the vibrational mode are equally spaced with $\delta\epsilon = h\nu$.

Like the velocity distribution, the number of particles having energy in each bin are computed. The probability of a particle having energy in the j^{th} quantum level is then simply calculated as

$$f_j = \frac{N_j}{N} \tag{81}$$

Like the velocity distribution, data from multiple time steps are incorporated. The update step for these modes is identical to that of equation (77).

The actual source code listing used to setup and compute the various distribution functions is given in Appendix A. Of particular interest are the subroutines 'pdfsetup.c' and 'pdfcalc.c'.

3.3 Calculation of Entropy and Entropy Flux

Having calculated distribution functions for the various entropy forms as above, it is then possible to compute the entropy of the various energy modes. The purpose of this section is to illustrate how this was accomplished.

The translational specific entropy in the current cell from equation (63) is given in terms of the separable distribution function below.

$$\begin{aligned} s_{tr} &= R - R\int_{-\infty}^{\infty} f\left(\vec{c}\right) \ln\left(\frac{h^3 n f(\vec{c})}{m^3}\right) \partial\vec{c} \\ &= R - R\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_x\left(c_x\right) f_y\left(c_y\right) f_z\left(c_z\right) \ln\left(\frac{h^3 n f_x(c_x) f_y(c_y) f_z(c_z)}{m^3}\right) \partial c_x \partial c_y \partial c_z \\ &= R - R\int_{-\infty}^{\infty} f_x\left(c_x\right) \ln\left(\frac{h^3 n^{1/3} f_x(c_x)}{m^3}\right) \partial c_x - \int_{-\infty}^{\infty} f_y\left(c_y\right) \ln\left(\frac{h^3 n^{1/3} f_y(c_y)}{m^3}\right) \partial c_y \\ &\quad - \int_{-\infty}^{\infty} f_z\left(c_z\right) \ln\left(\frac{h^3 n^{1/3} f_z(c_z)}{m^3}\right) \partial c_z \end{aligned} \tag{82}$$

In terms of the discrete functions calculated through DSMC this can be written,

$$\begin{aligned} s_{tr} = R - R\sum_{j=1}^{n_{int}} & f_{x,j} \ln\left(\frac{h^3 n^{1/3} f_{x,j}}{m^3}\right) \delta c_x - f_{y,j} \ln\left(\frac{h^3 n^{1/3} f_{y,j}}{m^3}\right) \delta c_y \\ & - f_{z,j} \ln\left(\frac{h^3 n^{1/3} f_{z,j}}{m^3}\right) \delta c_z \end{aligned} \tag{83}$$

Likewise, performing similar operations on the translational entropy flux vector of equation (72) one obtains,

$$\begin{aligned} \vec{\mathcal{T}_{tr,1}} &= u_x kn - \\ & kn\sum_{j=1}^{n_{int}} \left[c_{x,j} f_{x,j} \ln\left(\frac{hn^{1/3} f_{x,j}}{m}\right) \delta c_x - u_x f_{y,j} \ln\left(\frac{hn^{1/3} f_{y,j}}{m}\right) \delta c_y - u_x f_{z,j} \ln\left(\frac{hn^{1/3} f_{z,j}}{m}\right) \delta c_z \right] \\ \vec{\mathcal{T}_{tr,2}} &= u_y kn - \\ & kn\sum_{j=1}^{n_{int}} \left[u_y f_{x,j} \ln\left(\frac{hn^{1/3} f_{x,j}}{m}\right) \delta c_x - c_{y,j} f_{y,j} \ln\left(\frac{hn^{1/3} f_{y,j}}{m}\right) \delta c_y - u_y f_{z,j} \ln\left(\frac{hn^{1/3} f_{z,i}}{m}\right) \delta c_z \right] \\ \vec{\mathcal{T}_{tr,3}} &= u_z kn - \\ & kn\sum_{j=1}^{n_{int}} \left[u_z f_{x,j} \ln\left(\frac{hn^{1/3} f_{x,j}}{m}\right) \delta c_x - u_z f_{y,j} \ln\left(\frac{hn^{1/3} f_{y,j}}{m}\right) \delta c_y - c_{z,j} f_{z,j} \ln\left(\frac{hn^{1/3} f_{z,j}}{m}\right) \delta c_z \right] \end{aligned} \tag{84}$$

The rotational and vibrational specific entropy of equation (63) is computed using

$$s_{rot} = -R \sum_{j=1}^{n_{rs}} f_{rot,j} \ln \left(\frac{f_{rot,j}}{2j+1} \right) \tag{85}$$

$$s_{vib} = -R \sum_{j=1}^{n_{vs}} f_{vib,j} \ln f_{vib,j} \tag{86}$$

where n_{rl}, and n_{vl} are the number of rotational and vibrational levels included in the distributions, respectively. The flux terms are simply calculated using equation (72). The subroutines used to calculate these properties are 'getentropy.c' and 'entropflux.c' which are found in Appendix A. It should be said that like the other macroscopic quantities calculated using DSMC both the entropy and entropy fluxes are averaged over many time steps to reduce the level of statistical scatter.

3.4 Calculation of the Entropy Generation Rate

With the entropy and entropy fluxes calculated, it is possible to determine the entropy generation rate by employing equation (70). To utilize this equation, both temporal and spatial gradients are required. The methods employed to calculate these quantities are discussed here.

3.4.1 Temporal Gradient Approximation. A number of methods exist for calculating the temporal derivative in equation (70). Likely, the simplest of these is a simple first order backward difference.

$$\left[\frac{\partial(\tilde{s})}{\partial t} \right]^m = \frac{[\tilde{s}]^m - [\tilde{s}]^{m-1}}{\Delta t} + O(\Delta t) \tag{87}$$

Where the index m is used to denote the current time step. Higher order approximations can be used, but with the small time step associated with the DSMC process it is likely that a first order difference is adequate. This method was implemented in 'timederiv.c' listed in Appendix A.

It should be noted that although this calculation has been programmed, no unsteady flows were examined in the current study which required its use. Significant fluctuations associated with the DSMC process may occur between consecutive time steps. Although upon average these fluctuations will die out in a steady flow, it was thought best not to introduce any additional statistical scatter in the entropy generation results. Therefore in the flows examined, the temporal derivative was assumed to be zero and was not actually calculated in this manner.

3.4.2 Spatial Gradient Approximation. Since MONACO is set up to handle unstructured grids, a method of calculating spatial gradients that is independent of grid topology is desirable. One such method is the least squares approach. The idea behind this approach is to approximate the gradients by minimizing the sum of the squared error arrived at via a first order Taylor approximation at adjacent cell centers.

Consider a cell i at location $\vec{r_i}$ with C adjacent cell centers. Then for some property Q which varies between cell i and some adjacent cell, j, one can write to first order,

$$Q_j \approx Q_i + \vec{\nabla Q} \cdot \Delta \vec{r_{i,j}} = Q_i + \frac{\partial Q}{\partial x}\Delta x_{i,j} + \frac{\partial Q}{\partial y}\Delta y_{i,j} + \frac{\partial Q}{\partial z}\Delta z_{i,j} \tag{88}$$

Equation (88) represents C linear equations in three unknowns. This over-constrained system can be written in matrix form as

$$\begin{bmatrix} \Delta x_{i,1} & \Delta y_{i,1} & \Delta z_{i,1} \\ \Delta x_{i,2} & \Delta y_{i,2} & \Delta z_{i,2} \\ & \vdots & \\ \Delta x_{i,C} & \Delta y_{i,C} & \Delta z_{i,C} \end{bmatrix} \begin{bmatrix} \frac{\partial Q}{\partial x} \\ \frac{\partial Q}{\partial y} \\ \frac{\partial Q}{\partial z} \end{bmatrix} = \begin{bmatrix} Q_i - Q_1 \\ Q_i - Q_2 \\ \vdots \\ Q_i - Q_C \end{bmatrix} \tag{89}$$

The least squares process is then applied to find $\frac{\partial Q}{\partial x}, \frac{\partial Q}{\partial y}, \frac{\partial Q}{\partial z}$ which minimize the least square error, E

$$E = \sum_{j=1}^{C} w_j^2 e_j^2 \tag{90}$$

where e_j is the error in Q which occurs at cell center j due to the approximation, and w_j is a weight function. Namely,

$$e_j = Q_j - \left[\frac{\partial Q}{\partial x} \Delta x_{i,j} + \frac{\partial Q}{\partial y} \Delta y_{i,j} + \frac{\partial Q}{\partial z} \Delta z_{i,j} + Q_i \right] \tag{91}$$

Typically w_j is set to unity. However, if it is desired to weight closer cells more heavily we can define the weight function to be

$$w_j^2 = \frac{1}{\parallel \Delta \vec{r}_{i,j} \parallel} \tag{92}$$

The process of solving this minimization problem is rather complex involving several manipulations of the above equations and application of the Graham-Schmidt process. This is outlined in Blazek [6: pp. 162-165] and will not be discussed here. Blazek gives the solution to be

$$\nabla Q = \sum_{j=1}^{c} \gamma_{ij} \left(Q_j - Q_i \right) \tag{93}$$

where,

$$\gamma_{ij} = \begin{bmatrix} \alpha_{ij,1} - \frac{R_{12}}{R_{11}} \alpha_{ij,2} + \beta \alpha_{ij,3} \\ \alpha_{ij,2} - \frac{R_{23}}{R_{22}} \alpha_{ij,3} \\ \alpha_{ij,3} \end{bmatrix} \tag{94}$$

$$\begin{aligned} \alpha_{ij,1} &= \frac{\Delta x_{ij}}{R_{11}^2} \\ \alpha_{ij,2} &= \frac{1}{R_{22}^2} \left(\Delta y_{ij} - \frac{R_{12}}{R_{11}} \Delta x_{ij} \right) \\ \alpha_{ij,3} &= \frac{1}{R_{33}^2} \left(\Delta z_{ij} - \frac{R_{23}}{R_{22}} \Delta y_{ij} + \beta \Delta x_{ij} \right) \end{aligned} \tag{95}$$

$$\beta = \frac{R_{12}R_{23} - R_{13}R_{22}}{R_{11}R_{22}} \tag{96}$$

The R_{pq} terms result from an upper triangular matrix that occurs in the solution process and are given as follows.

$$\begin{aligned}
R_{11} &= \sqrt{\textstyle\sum_{j=1}^{C} \Delta x_{ij}^2} \\
R_{12} &= \tfrac{1}{R_{11}} \textstyle\sum_{j=1}^{C} \Delta x_{ij} \Delta y_{ij} \\
R_{13} &= \tfrac{1}{R_{11}} \textstyle\sum_{j=1}^{C} \Delta x_{ij} \Delta z_{ij} \\
R_{22} &= \sqrt{\textstyle\sum_{j=1}^{C} \Delta y_{ij}^2 - R_{12}^2} \\
R_{23} &= \tfrac{1}{R_{22}} \textstyle\sum_{j=1}^{C} \Delta y_{ij} \Delta z_{ij} - \tfrac{R_{12}}{R_{11}} \textstyle\sum_{j=1}^{C} \Delta x_{ij} \Delta z_{ij} \\
R_{33} &= \sqrt{\textstyle\sum_{j=1}^{C} \Delta z_{ij}^2 - (R_{13}^2 + R_{23}^2)}
\end{aligned} \tag{97}$$

Blazek states that this approach is first order accurate and consistent, regardless of element type, and has comparable computational cost to the Green-Gauss approach. Further, the R, α, β, and γ values are all precomputable since they depend only on the grid geometry. This leads to a fairly compact and efficient method of calculating the spatial gradients which was used in the current study. The source code used to implement these calculations is also found in Appendix A and in subroutine 'spatialgrad.c'.

3.5 Other Required Modifications

In addition to the previously discussed calculations, a number of modifications internal to MONACO were required. These modifications included: appropriate calls to the aforementioned subroutines within the DSMC process; sampling and updating of the distribution functions and entropy results; output of appropriate data; reading in of appropriate data and control parameters; provision of data storage consistent with the existing MONACO data structure; provision for split domain initialization; amongst others.

The majority of additions which directly control the calculations described above are contained in the 'calc_cells.c' subroutine of the MONACO program. The additions to this routine are also given in Appendix A, although only the specific additions are listed rather than the entire subroutine. Smaller, less instructive modifications will not be discussed here.

3.6 Problem Setup: The Normal Shock Problem

The normal shock problem provides a relatively simple means of observing the breakdown of the Navier-Stokes equations. It has been known for some time that the Navier-Stokes equations break down in shock waves somewhere around a Mach number of two. Beyond this range, shock profiles are significantly thinner than experiment. For these reasons, along with the availability of experimental data, the normal shock problem provides an ideal test case for examining continuum breakdown in terms of entropy generation. To this end, shocks were simulated in both argon and nitrogen and DSMC results were compared with those obtained by numerically integrating the one-dimensional Navier-Stokes equations.

The upstream conditions for these runs were chosen to match those of the experimental data of Alsmeyer [2]. Namely, the upstream temperature was set to 300 K, and the upstream pressure was set to 6.668 Pa. Data was taken at upstream Mach numbers of 1.2, 1.4, 1.75, 2.0, 2.5, 3.0, 4.0, 6.0, and 8.0 in both gasses. The argon case also included additional runs at Mach 1.55, 5.0, and 10.0. The remainder of this chapter will deal with how these results were obtained.

3.6.1 Monte-Carlo Simulation of the Normal Shock. The normal shock problem is fundamentally a one-dimensional flow problem; MONACO, however, is a two-dimensional solver. To take advantage of the dimensionality of the problem, the grid height normal to the flow direction was restricted to a value of approximately 1.5 upstream mean free paths. This allowed the flow to be computed without spending too much computational time on the transverse direction. The length of the grid was

one hundred upstream mean free paths to ensure that the entire shock profile was captured and sufficient length was provided to meet the boundary conditions. The grid employed can be seen in Figure 4 below.

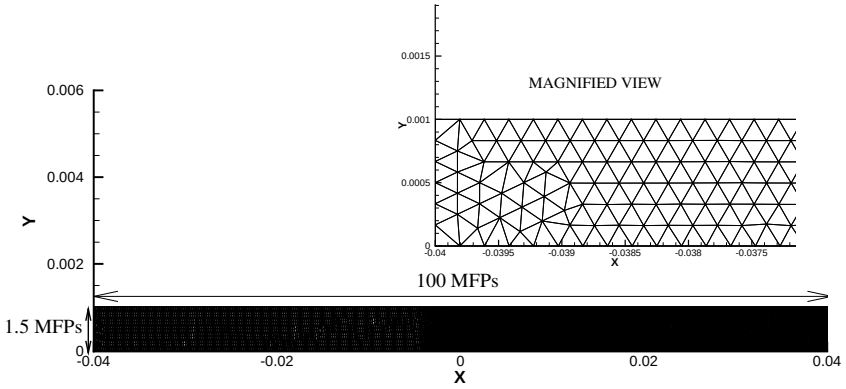


Figure 4: Grid used in DSMC Normal Shock Simulations

It can be seen that the cell dimensions used are quite small. The maximum cell size of the grid was first set to one-quarter of the upstream mean free path. After a run was completed, the grid was adaptively refined so that each cell was no larger than one quarter of the local mean free path. This refinement is clearly seen in Figure 4 on the downstream half of the grid. The simulations were performed once more on these refined grids to produce the final results.

The upper and lower boundaries employed symmetry boundary conditions to avoid any wall effects in such a narrow domain. The upstream and downstream boundaries utilized stream type boundary conditions, with the downstream conditions being set by the normal shock relations. The ambient condition generator subroutine in MONACO was modified to allow the two halves of the domain to be initialized to the upstream and downstream conditions respectively. Bird has stated that the usage of such boundary conditions is suboptimal for the normal shock problem as the number of particles entering and leaving the domain fluctuates at each boundary. This fluctuation causes the shock to execute a random walk and leads to a smearing

of the time averaged data [5]. He has suggested employing a specialized moving downstream boundary condition along with a stabilization routine to alleviate this effect, however, implementation of these items was not readily achievable in the case of the two-dimensional, unstructured code utilized here.

The simulation was allowed to evolve over 50,000 simulation steps before any data was sampled. The simulation time step was chosen in each case so that this iteration represented the time it took the flow to move twenty shock widths as suggested by Bird[5]. This gave a time step several orders of magnitude lower than the mean collision time. It should be said that there are no inherent stability limitations to the DSMC process as there are to continuum CFD methods, however, greater accuracy is attained as both the time step and cell size tend to zero, and as the statistical particle weight tends to unity.

Between 220,000 and 260,000 simulated particles were employed. Sampling of the distribution functions also began after 50,000 simulation steps, though entropy calculations were delayed from beginning until after 65,000 iterations to ensure sufficient data had been used in generating the distribution functions. The velocity distributions spanned six standard deviations and utilized approximately ten subintervals per standard deviation. In the nitrogen cases, three-hundred rotational levels were included in the distributions, along with fifty vibrational levels. In reality, dissociation is achieved well before fifty levels, however, this parameter was purposely set high to ensure the calculations were proceeding normally. Data was then sampled over the next 85,000 simulation steps. The relevant gas properties employed in these simulations is given in Table 1.

3.6.2 Numerical Integration of the Navier-Stokes Equations. The one-dimensional, steady Navier-Stokes equations, are given in equations (98) through (100) [12].

$$\frac{d}{dx}(\rho u) = 0 \tag{98}$$

Table 1: Molecular Parameters used in DSMC Simulations

Property	Argon	Nitrogen
Molecular Weight (g/mol)	39.95	28.01
VHS Exponent ω [4]	0.31	0.24
VHS Reference Diameter (pm) [4]	417.0	407.0
VHS Reference Temperature (K)	273.0	273.0
Rotational degrees of freedom	0	2
Vibrational degrees of freedom	0	1.8
Θ_{rot} (K) [5]	N/A	2.88
Θ_{vib} (K) [5]	N/A	3371.0
Max Rotational Collision # [31]	N/A	15.7
T_{ref} in Rot. Model (K) [31]	N/A	80
Probability of Vibrational Exchange	N/A	0.01
Equilibrium Separation (pm) [1]	N/A	109.769
Oscillating Frequency (Hz) [1]	N/A	7.071×10^{13}

$$\frac{d}{dx}\left(\rho u^2 + p - \tau\right) = 0 \tag{99}$$

$$\frac{d}{dx}\left(\rho u e + p u - \tau u + q\right) = 0 \tag{100}$$

where e is the specific internal energy of the gas, and the form assumed by the Navier-Stokes equations for the shear stress tensor and heat flux vector is given in equations (101) and (102) below.

$$\tau = (2\mu + \lambda)\frac{du}{dx} \tag{101}$$

$$q = -\kappa\frac{dT}{dx} \tag{102}$$

where μ is the coefficient of viscosity, λ is the bulk viscosity, and κ is the thermal conductivity. The bulk viscosity term is seldom important, but in the case of the normal shock the large flow gradients typically warrant its inclusion. λ is typically determined using Stokes' hypothesis.

$$\lambda = -\frac{2}{3}\mu \tag{103}$$

It can be shown that this result is predicted by Chapman-Enskog theory for monatomic gases [38].

These equations, augmented with the ideal gas equation of state and temperature dependent Sutherland's Law expressions for the viscosity and thermal conductivity, were numerically integrated using a Mathematica solver developed by Camberos and Chen. Camberos and Chen also showed that the entropy generation rate for the Navier-Stokes equations can be computed by the following [12].

$$\dot{\bar{s}} = \frac{(2\mu + \lambda)}{T} \left(\frac{du}{dx} \right)^2 + \frac{\kappa}{T^2} \left(\frac{dT}{dx} \right)^2 \tag{104}$$

These equations were solved for the same upstream conditions and Mach numbers as discussed in the previous section.

IV. Results

This chapter discusses the results obtained from the simulations discussed in the previous chapter. Results for normal shocks in argon will be presented first, followed by those obtained for nitrogen. DSMC data is compared against results obtained by numerically integrating the Navier-Stokes equations. Deviation between the methods is examined in terms of entropy generation.

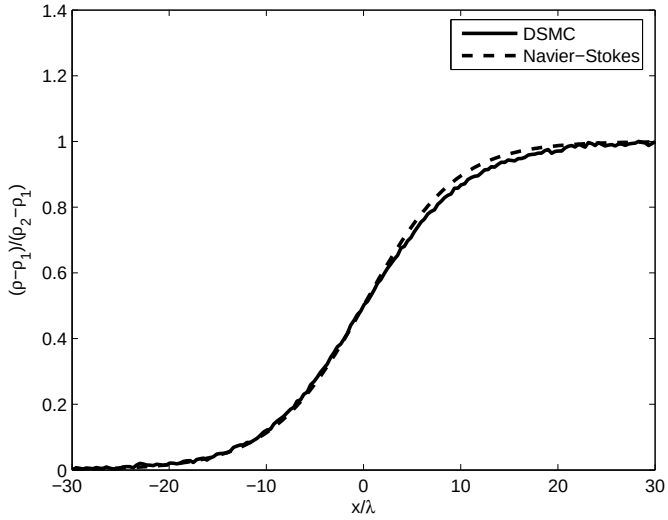
4.1 Argon Results

Breakdown of the Navier-Stokes equations in the normal shock problem has traditionally been accepted as becoming evident at a Mach number somewhere around 1.9 [2, 27]. Below this Mach number, fairly good agreement is observed; at higher Mach numbers the Navier-Stokes equations predict much too thin of a shock [20, 22, 27, 33]. These trends are also observed in the results of the current work.

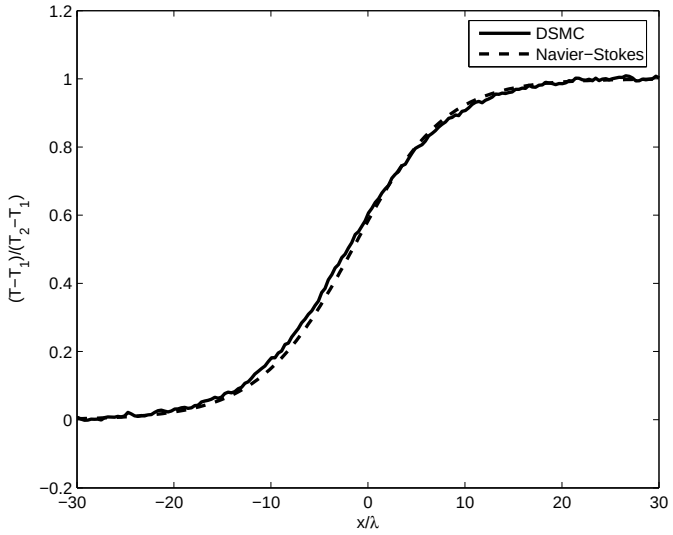
It should be noted that in order to obtain accurate results from the Navier-Stokes solver, the bulk viscosity term from equation (101) was set to zero. White [41] also observed this phenomena when attempting to match the experimental data for normal shocks in air, although for helium Stokes' hypothesis provided better results. Additionally, since there is no constraint on the shock location in either method, all of the results presented here have been centered about the same point. Namely, the coordinate $x = 0$ was chosen to represent the location inside the shock where the density has attained one-half of its final value.

Figure 5 shows the density and temperature profiles calculated at a Mach number of 1.2. Good agreement is seen in both temperature and pressure, and the shocks span approximately the same distance.

The entropy and entropy generation profiles for this case can be seen in Figure 6. In both Figures 6(a) and 6(b), a significant level of scatter is observed in the DSMC data. At this low Mach number, the change in flow properties across the shock are so small that DSMC has difficulty resolving the jump [5]. This is true for all of the flow variables, however, the entropy variables seem to be particularly sensitive.



(a) Density Profiles



(b) Temperature Profiles

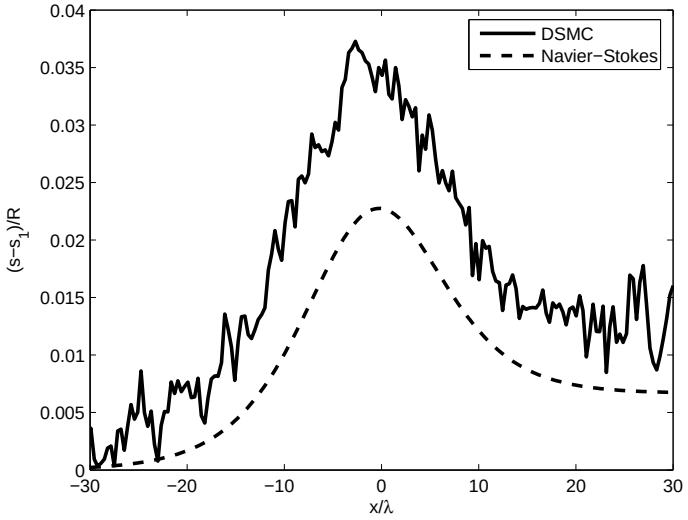
Figure 5: Mach 1.2 Argon Density and Temperature Profiles

Instantaneous DSMC results exhibit so much scatter at these lower Mach numbers that the shock is scarcely discerned; it is only through averaging over a large number of simulation steps that the shock is revealed.

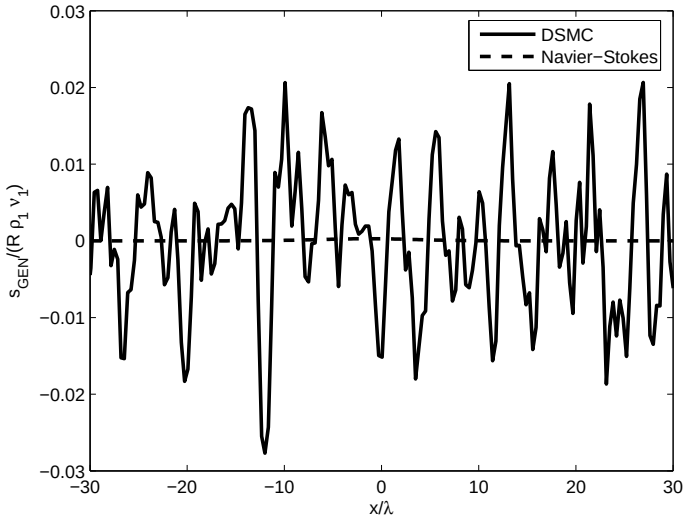
The entropy profiles seen in Figure 6(a) are seen to agree qualitatively, although the DSMC profile shows slightly increased magnitude over the Navier-Stokes profile over the entire domain. This is not necessarily disconcerting when the very small change in entropy across this weak shock is considered in light of the problems inherent to the DSMC process discussed above. The DSMC entropy generation profile seen in Figure 6(b) shows no decipherable peak where the shock should be and is aptly characterized as fluctuations about a zero mean value. The Navier-Stokes profile exhibits a very small peak at the shock location, although in comparison to the scale of the scatter in the DSMC data, the peak is virtually indecipherable in Figure 6(b).

Examining only the DSMC data, one would likely conclude that the essentially zero entropy generation would imply that the continuum equations are valid. From the temperature and density data, this is seen to be true; the very small errors observed in these profiles suggest that the continuum equations are valid through this shock. These results demonstrate that entropy generation computed via DSMC is able to capture continuum onset. Further, DSMC is seen to be a nonoptimal solver for borderline continuum flows. The computational time and effort used to reduce the statistical scatter produces results which are, at best, only marginally better than the results of the Navier-Stokes equations. The computational time associated with the Navier-Stokes equations is minuscule compared to that required of DSMC at these low Mach numbers,and hence, would be the preferable solver to use here.

Density and temperature results for the Mach 1.75 shock in argon are shown in Figure 7. At this Mach number, discrepancies can be seen in both the density and temperature profiles. The shock predicted by DSMC is slightly thicker than that predicted by the Navier-Stokes solver, however both predicted shock profiles have become thinner than the Mach 1.2 case. This effect is observed experimentally [2, 27, 30, 34]. Specifically, shock thickness in argon is seen to initially decrease fairly rapidly but as Mach number continues to increase the shock thickness eventually levels out and may begin to increase at higher Mach numbers. The temperature predicted by DSMC is seen to begin increasing slightly earlier than Navier-Stokes predicts. This



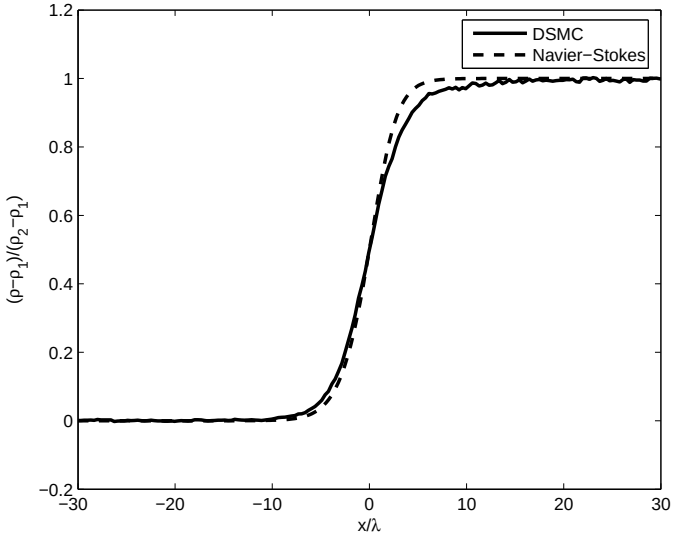
(a) Entropy Profiles



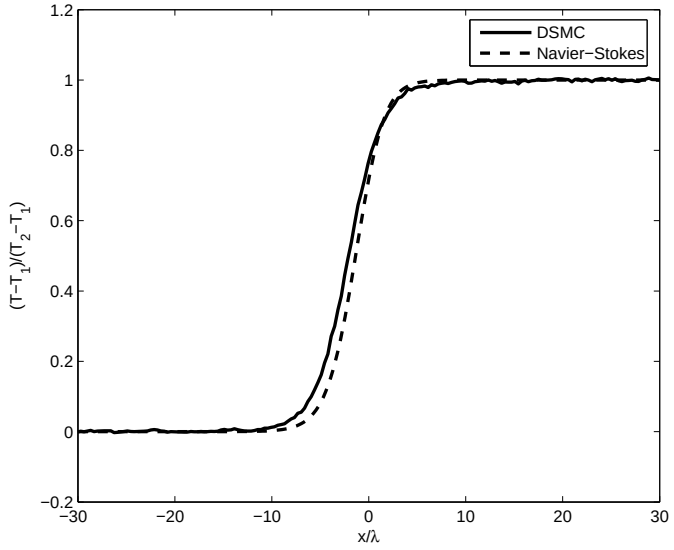
(b) Entropy Generation

Figure 6: Mach 1.2 Argon Entropy and Entropy Generation Profiles

flow is approaching the traditional borderline of continuum breakdown. As the Mach number is further increased, stronger regions of nonequilibrium will invalidate the Navier-Stokes equations and both of these effects can be expected to propagate.



(a) Density Profiles



(b) Temperature Profiles

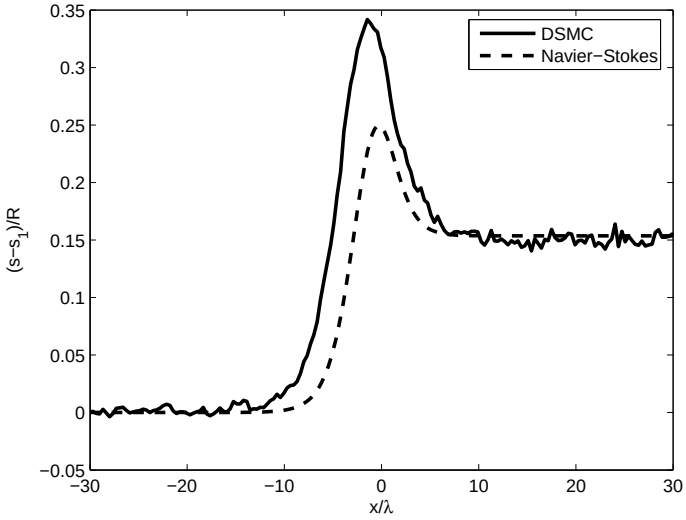
Figure 7: Mach 1.75 Argon Density and Temperature Profiles

Figure 8 presents the entropy data for the Mach 1.75 shock in argon. In Figure 8(a), the entropy predicted by DSMC is seen to begin to increase several mean free paths before the Navier-Stokes result. This shows that nonequilibrium actually exists

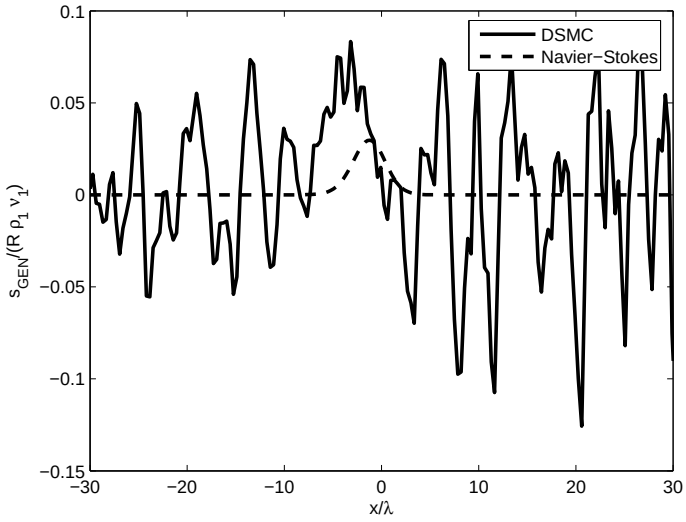
before it is observed in the Navier-Stokes solution. The peak entropy value observed in the DSMC data is also larger than that predicted by Navier-Stokes, indicating that a larger degree of nonequilibrium exists in the shock than is predicted by the continuum equations. The Navier-Stokes equations are therefore seen to limit the extent of observable nonequilibrium versus what is seen in reality. The entropy generation rate predicted by the Navier-Stokes equations is now seen to be almost on the same order as the scatter in the DSMC data. Also, a peak at the approximate shock location is almost decipherable in the DSMC data, though the level of scatter prohibits its rigorous quantification. In accordance with traditional understanding, this Mach 1.75 shock is nearing the edge of the continuum regime; as Mach number continues to increase, the peak in entropy generation should become more decipherable.

The density and temperature profiles for the Mach 2.5 argon shock are shown in Figure 9. The Navier-Stokes equations predict a significantly thinner shock than DSMC which causes correspondingly sizable errors in both density and temperature due to the delayed shock front. The Navier-Stokes equations are losing their validity quickly in this regime as the degree of nonequilibrium continues to grow.

Entropy results for the Mach 2.5 argon shock are seen in Figure 10. The DSMC entropy profile in Figure 10(a) begins significantly increasing more than six mean free paths before the Navier-Stokes profile. This explains why the shock front has been so hard to capture with approaches using continuum data. The Navier-Stokes equations are seen to limit the degree of observable nonequilibrium in the normal shock flow. This means that no nonequilibrium effects can be observed in the Navier-Stokes results until after the flow has already entered a region of significant nonequilibrium. Therefore, it is likely that any breakdown parameter computed using continuum data will fail to adequately capture the shock front. This explains why even the entropy based parameters examined by Camberos, Chen, and Boyd [12, 15] were shown to be insufficient indicators of continuum breakdown in the normal shock scenario.



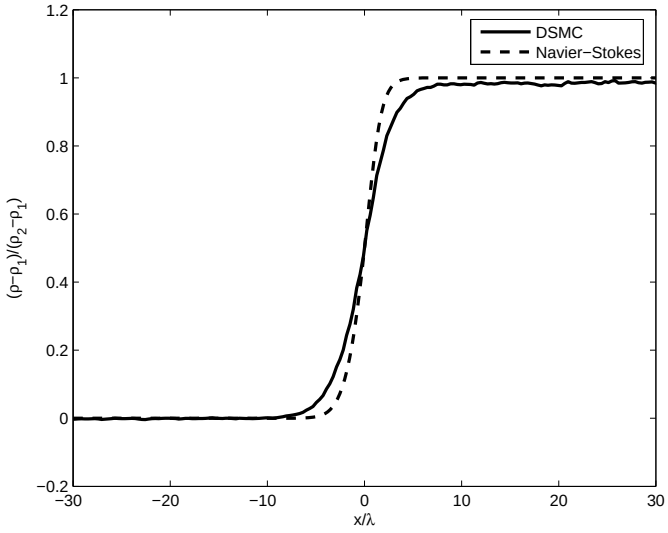
(a) Entropy Profiles



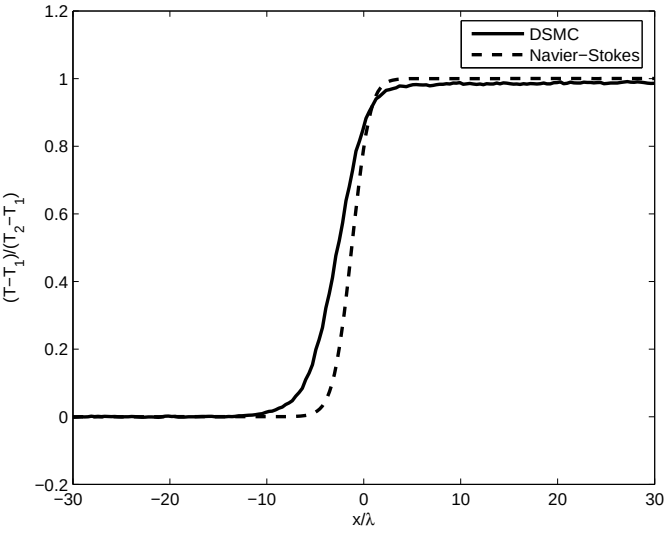
(b) Entropy Generation

Figure 8: Mach 1.75 Argon Entropy and Entropy Generation Profiles

The DSMC entropy generation through the shock illustrated in Figure 10(b) has developed a significant spike at the shock location which is distinct from the statistical scatter. Here, it is also evident that significant nonequilibrium effects are occurring before the Navier-Stokes equations predict. This is seen in the fact that



(a) Density Profiles

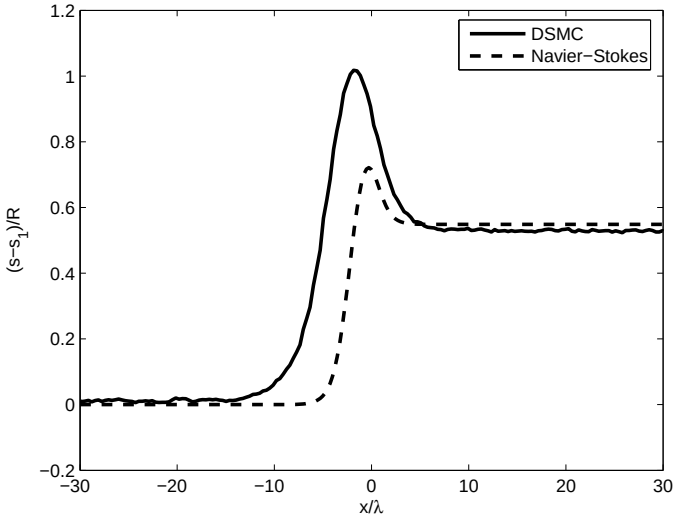


(b) Temperature Profiles

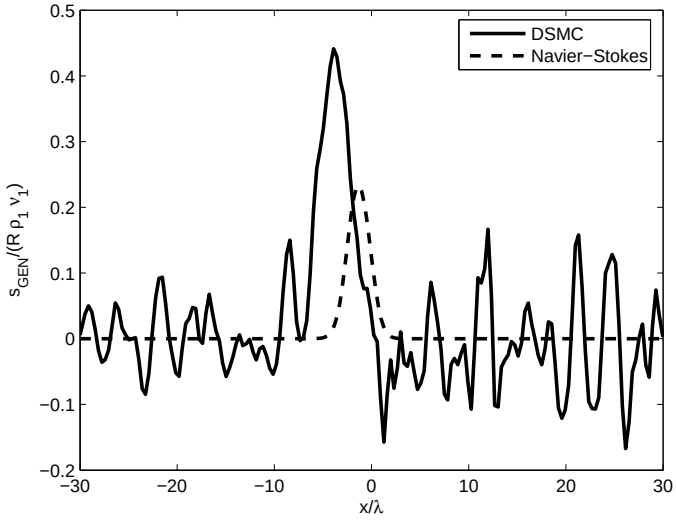
Figure 9: Mach 2.5 Argon Density and Temperature Profiles

the region of entropy generation predicted by DSMC is thicker and begins further upstream versus the Navier-Stokes results. Since the Navier-Stokes equations have

failed to capture these significantly nonequilibrium effects it is safe to conclude that continuum breakdown has occurred in this flow.



(a) Entropy Profiles



(b) Entropy Generation

Figure 10: Mach 2.5 Argon Entropy and Entropy Generation Profiles

Figure 11 shows the density and temperature results for the Mach 6 shock wave. At this point the Navier-Stokes equations have lost any ability to properly capture the

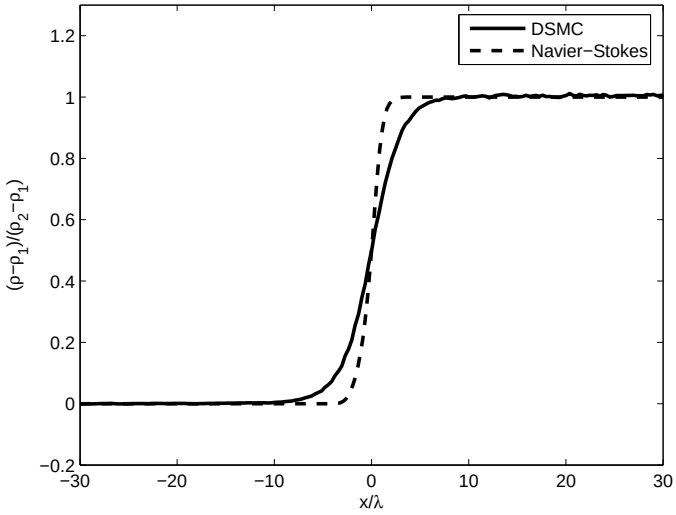
shock profile. The shock is much too thin, and the temperature begins to increase well upstream of the Navier-Stokes prediction. This shows why the local Knudsen number based parameters computed using continuum data were insufficient for examining the normal shock. Although strong gradients in the macroscopic variables are seen to exist in the DSMC data, no gradients can be observed in the Navier-Stokes data until well after the shock region has been entered.

The entropy results for the Mach 6.0 argon shock in Figure 12 confirm that the Navier-Stokes equations fail to exhibit any indicators of nonequilibrium until after the shock region has already been entered. The Navier-Stokes peak entropy in Figure 12(a) is again less than its DSMC counterpart. Interestingly, the shock predicted by Navier-Stokes shown in Figure 12(b) has a larger spike in entropy generation than DSMC. The predicted shock has now become so thin that in order to meet the entropy jump conditions the equations must over predict the entropy generation rate through the shock. Significant nonequilibrium effects are seen to exist well upstream of where Navier-Stokes predicts as typified by the strong entropy production rate observed in in the DSMC data. This result shows that if the breakdown of the continuum equations is to be examined, it must be examined from a kinetic theory based approach. It should not be expected that the continuum equations can adequately predict their own failure, as the small perturbation from equilibrium assumption inherent in their derivation is seen to limit the magnitude of nonequilibrium observable in their results.

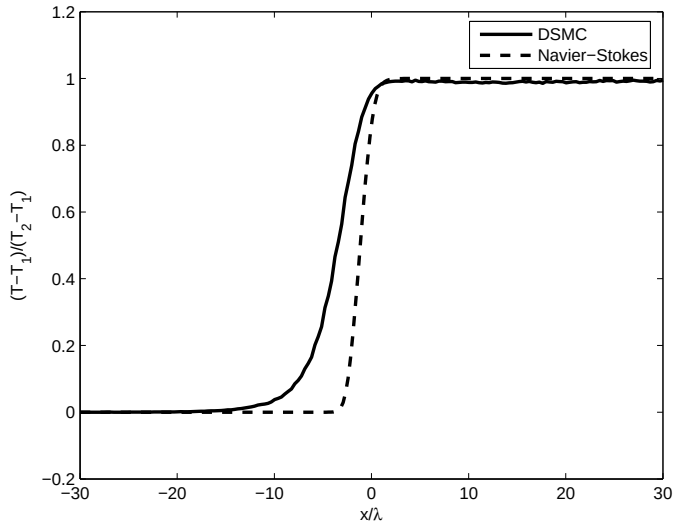
To better quantify the accuracy of these findings, shock thicknesses were compared with the data of Alsmeyer [2]. The density shock thickness is defined as follows.

$$t_s = \frac{\rho_2 - \rho_1}{\left(\frac{d\rho}{dx}\right)_{max}} \tag{105}$$

The comparison of DSMC and Navier-Stokes shock thicknesses with Alsmeyer's data is presented in Figure 13. The Navier-Stokes equations are seen to predict much



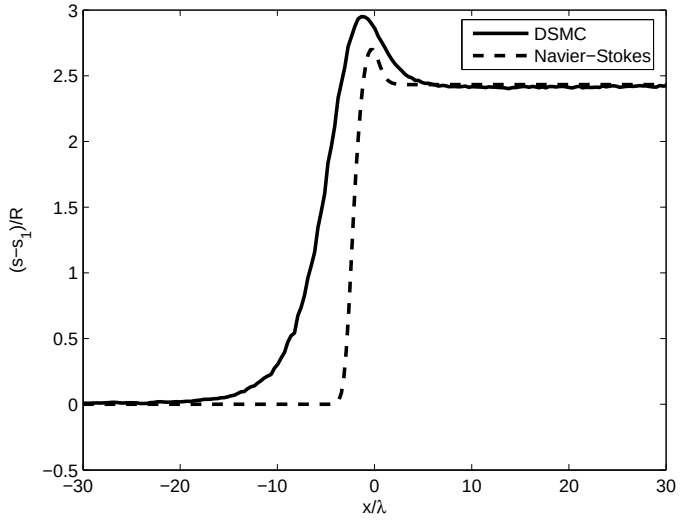
(a) Density Profiles



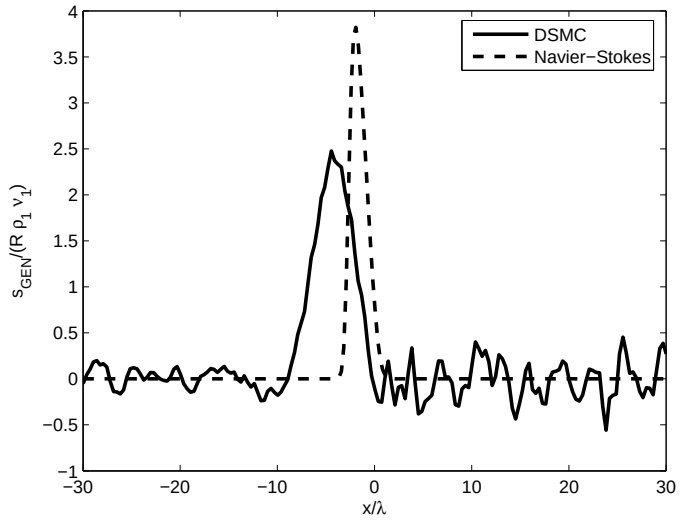
(b) Temperature Profiles

Figure 11: Mach 6.0 Argon Density and Temperature Profiles

thinner shocks than observed experimentally, especially at Mach numbers greater than two. Furthermore, they fail to capture the decrease in reciprocal shock thickness seen in the data at Mach numbers past four. The DSMC results are seen to be in much closer agreement with the experimental data, especially at Mach numbers lower



(a) Entropy Profiles



(b) Entropy Generation

Figure 12: Mach 6.0 Argon Entropy and Entropy Generation Profiles

than four. At the higher Mach numbers, DSMC is seen to predict thicker shocks than those observed experimentally. This is likely due to the random walk effect induced by the boundary conditions as discussed in the previous chapter. As the shock executes this random walk, the averaged data becomes smeared out and gives the appearance

of a thicker shock. This may be somewhat alleviated by decreasing the statistical particle weight, but will likely persist to some degree as long as the current boundary conditions are used. The DSMC results show good agreement with the experimental data and exhibit the correct trend with Mach number. Even with the random walk effect most of the DSMC data lies within the scatter bounds in Figure 13. This shows that the DSMC results provide an adequate representation of reality and are valid for examining continuum breakdown.

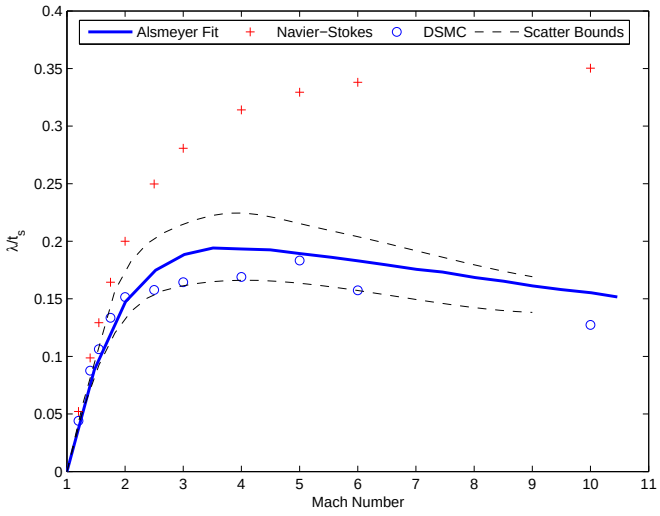


Figure 13: Reciprocal Shock Thickness in Argon

To examine the effect of entropy generation on continuum breakdown, the flow variable error between the two sets of results was defined as follows.

$$e_Q = \frac{|Q_{DSMC} - Q_{NS}|}{Q_{DSMC}} \tag{106}$$

where Q is the flow variable of interest. The maximum error observed in velocity, density, temperature, and entropy is plotted against Mach number in Figure 14. The overall error is defined below.

$$e_{max} = max\left\{e_{u,max}, e_{\rho,max}, e_{T,max}, e_{s,max}\right\} \tag{107}$$

Below a Mach number of 2, the Navier-Stokes equations yield solutions with no more than ten percent deviation from the DSMC results anywhere in the flow field. This is in qualitative agreement with the traditionally held belief that the Navier-Stokes equations breakdown somewhere around Mach 2. At Mach numbers below 1.75, maximum error is observed in the flow velocity, while at Mach 1.75 and above the maximum error is exclusively in the temperature. The large errors observed in temperature are due to the relatively large regions of nonequilibrium upstream of where the Navier-Stokes equations predict. The trend observed with Mach number is for the most part monotonic, though the maximum error in the flow velocity seems to be less sensitive to the effect of Mach number than the other flow variables. The overall error may exhibit an inflection point somewhere around Mach 1.6 where the temperature error appears to begin to dominate.

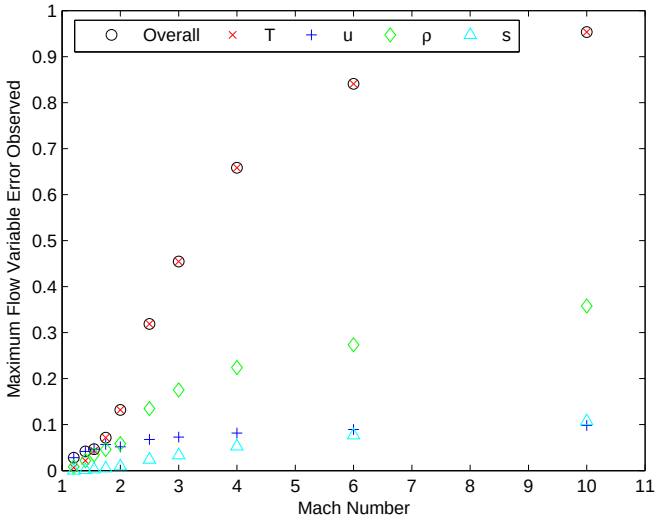


Figure 14: Maximum Error Observed in Flow Variables of Argon Shocks as a Function of Mach Number

In Figure 15, the error data is plotted against maximum entropy generation rate. The entropy generation was normalized by the product of the local entropy density and collision rate ($\tilde{s}\nu = \rho s \nu$), rather than by the more global parameters previously used. This provides a local description of the entropy generation as related

to the local entropy value and characteristic timescale. It is somewhat akin to an entropy based Knudsen parameter, and is likely more useful for examining continuum breakdown than normalizing by $R\rho_1\nu_1$ as before.

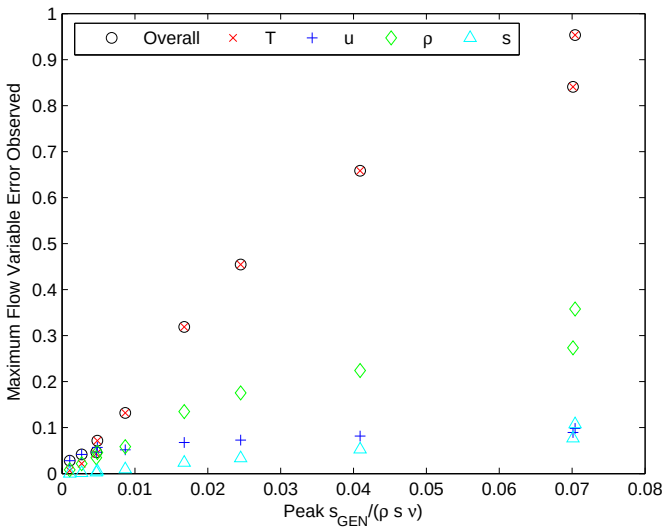


Figure 15: Maximum Error Observed in Flow Variables of Argon Shocks as a Function of Entropy Generation

The error is seen to increase, almost monotonically, with entropy generation. There is some scatter seen here, but the error is seen to be strongly tied to the peak entropy generation rate. The results suggest that less than ten percent error is observed so long as $\dot{\tilde{s}}/\rho s\nu \lesssim 0.006$.

The strong dependence of the error on entropy generation shows that this parameter is fundamentally tied to the processes which cause the Navier-Stokes equations to become invalid as was asserted in previous chapters. This is expected since entropy is produced in any form of nonequilibrium, including the translational nonequilibrium responsible for the failure of the continuum equations here.

4.2 Nitrogen Results

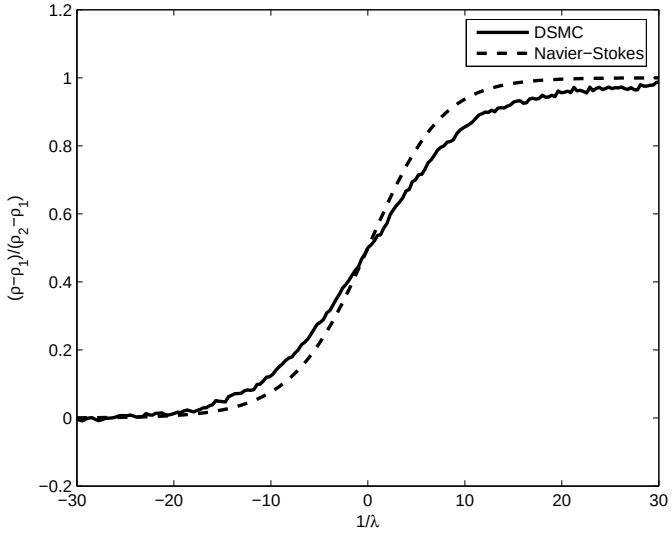
The normal shock simulations preformed in nitrogen show qualitatively similar behavior as those performed in argon. The nitrogen simulations allow for the ex-

amination of rotational and vibrational nonequilibrium which are not present in the argon cases. As with the argon simulations, the Navier-Stokes results were found to correlate better with experiment when the bulk viscosity term was removed from the solver.

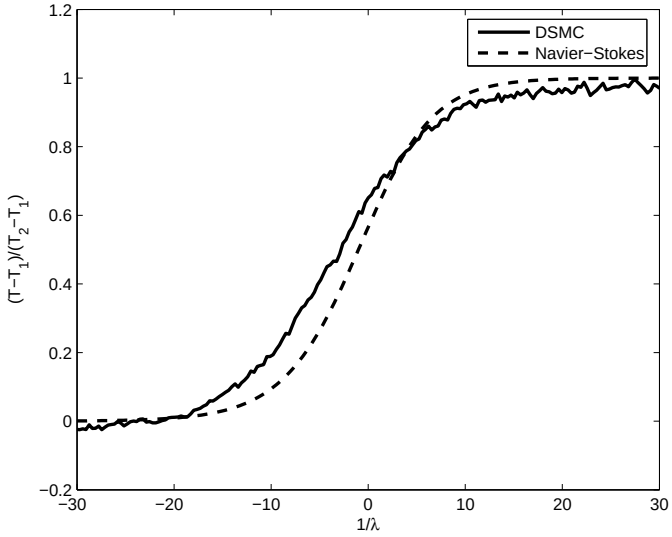
Accurate modeling of energy transfer to and from the internal modes is critical for calculating the entropy contribution of the internal structure. The parameters used to govern the simulation of these modes were listed in Table 1 in the previous chapter. These numbers represent reasonable values that are found in the various literature previously referenced. However, the specific values which work best in the solver for the present flow scenarios were unknown to the author.

Figure 16 presents the density and temperature profiles for the Mach 1.2 case. Agreement between the two methods is not as good as was seen in the argon case, and the Navier-Stokes equations are already seen to predict a thinner shock than DSMC. This is somewhat unexpected, as at this low Mach number the Navier-Stokes equations should be valid. The cause of this is unknown, but likely lies in the Navier-Stokes solver as the DSMC results will later be shown to be in excellent agreement with the experimental data at this Mach number. The problem may stem from the bulk viscosity issue previously discussed, or may be correctable by adjusting the coefficients used in the Sutherland's law expressions for viscosity and thermal conductivity.

The entropy results for the Mach 1.2 nitrogen case are shown in Figure 17. Like the Mach 1.2 argon case, Figure 17(a) shows the total entropy predicted by DSMC to be higher over the entire shock region. Also as in the argon case, a significant amount of scatter is present, likely due to trouble in resolving the relatively small change across this weak shock. No entropy generation spike is observable in the DSMC data of Figure 17(b) and the peak in the Navier-Stokes results is barely decipherable with the scale used. Figure 17(c) shows the entropy contributions of the various modes computed via DSMC. Interestingly, the translational entropy decreases across the shock. However, the rotational entropy increases more than enough to ensure that



(a) Density Profiles



(b) Temperature Profiles

Figure 16: Mach 1.2 Nitrogen Density and Temperature Profiles

the second law is not violated. Since the characteristic temperature of rotation is so low for nitrogen (2.88 K), the mode is fully activated and exhibits the strong coupling seen here with the translational mode. The Sackur-Tetrode equation for the

equilibrium translational entropy also predicts this decrease. This equation is given below. [38].

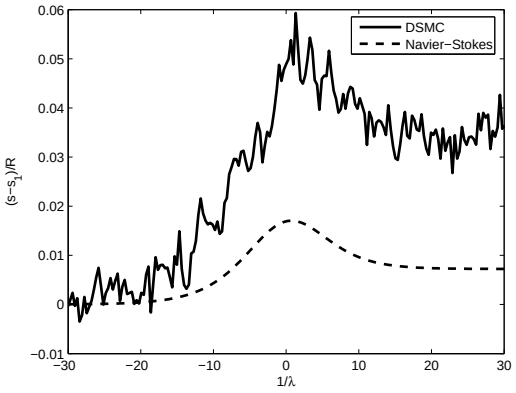
$$s_{tr} = \frac{5}{2} R \ln T - R \ln p + R \left\{ \ln \left[\left(\frac{2\pi m}{h^2} \right)^{3/2} k^{5/2} \right] + \frac{5}{2} \right\} \tag{108}$$

The vibrational entropy is identically zero across the entire domain, which means that all of the particles simulated are occupying the ground vibrational state. This is expected because the temperature jump across the shock is insufficient at this Mach number to activate the vibrational mode, whose characteristic temperature is 3371 K. Figure 17(d) displays contribution of the individual modes to the entropy generation rate. Both translation and rotation are essentially fluctuations about a zero mean value, and the vibrational mode is seen as being dormant.

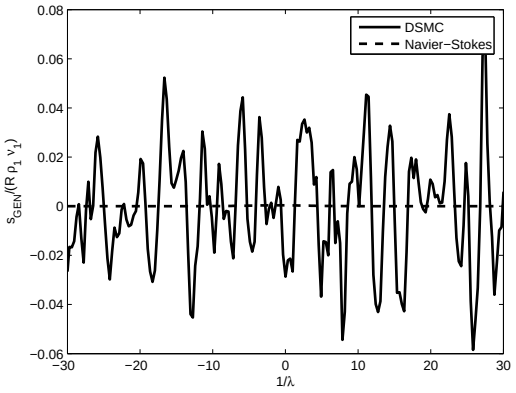
Density and temperature results for the Mach 1.75 nitrogen shock can be seen in Figure 18. Both methods predict a thinner shock than the Mach 1.2 case, however, a significant discrepancy is observed between the two methods. The temperature predicted by DSMC begins to increase several mean free paths before the Navier-Stokes data predicts, and the shock predicted by Navier-Stokes is substantially thinner than that of DSMC.

The entropy data for the Mach 1.75 shock are shown in Figure 19. The peak entropy predicted by DSMC is significantly higher than the Navier-Stokes peak in Figure 19(a). In addition, the entropy begins to increase sooner, signaling the presence of nonequilibrium not detected by the Navier-Stokes equations. This is an indication that the continuum hypothesis is beginning to break down.

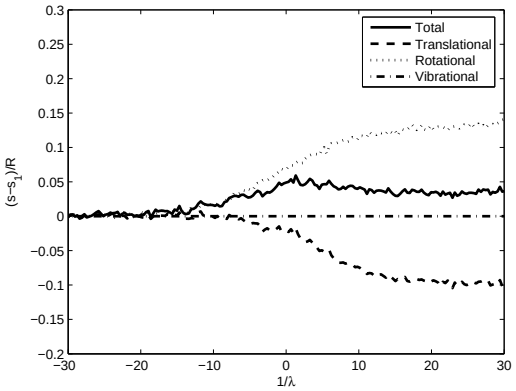
The entropy is seen to converge to two slightly different values aft of the shock. The final value of the entropy in both cases is completely determined by the equilibrium macroscopic properties downstream. In both cases these properties must satisfy the Rakine-Hugoniot relations and should therefore should be the same. This means the entropy should also be the same, provided the vibrational mode is not activated. This discrepancy may be corrected by adjusting the parameters controlling the in-



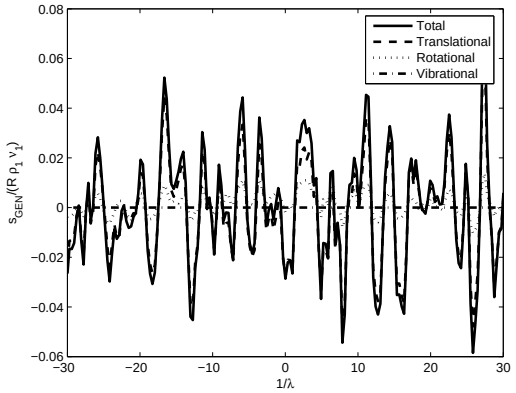
(a) Entropy Profiles



(b) Entropy Generation



(c) Entropy Contributions

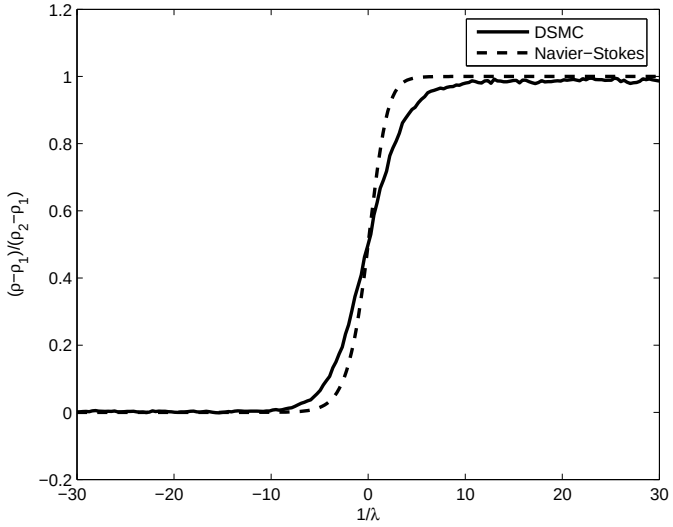


(d) Entropy Generation Contributions

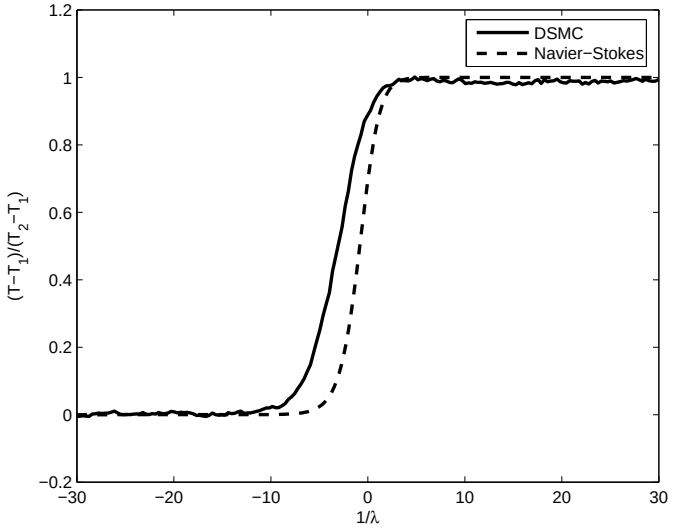
Figure 17: Mach 1.2 Nitrogen Entropy and Entropy Generation Profiles

ternal energy transfer in the DSMC solver. If these parameters are not correct, the rotational energy distribution may be somewhat erroneous, and although no large discrepancies are seen in the other flow variables the entropy will be adversely affected as the distribution function completely determines its value. Since the rotational mode is such a significant contributor, the overall entropy will also exhibit some error.

In Figure 19(b), a peak is seen to be developing in the entropy generation predicted by DSMC which is larger than the peak predicted by Navier-Stokes. The translational entropy decreases across the shock as shown in Figure 19(c), though



(a) Density Profiles

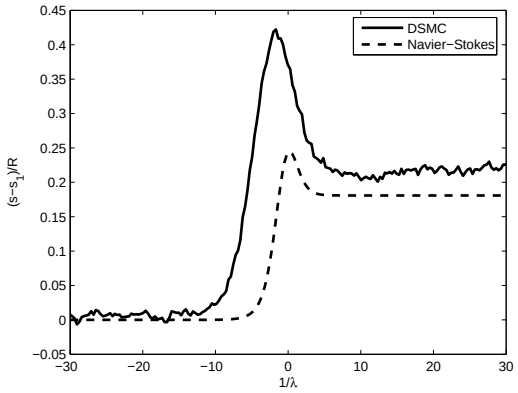


(b) Temperature Profiles

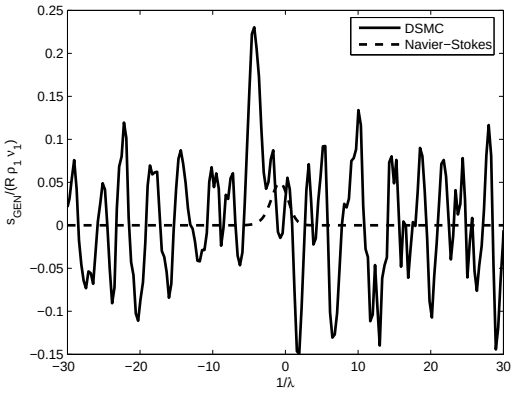
Figure 18: Mach 1.75 Nitrogen Density and Temperature Profiles

again the rotational contribution is seen to increase enough to compensate. Additionally, Figure 19(d) shows a noticeable region of positive rotational entropy generation at the approximate location of the shock. In this case, the scatter in the rotational

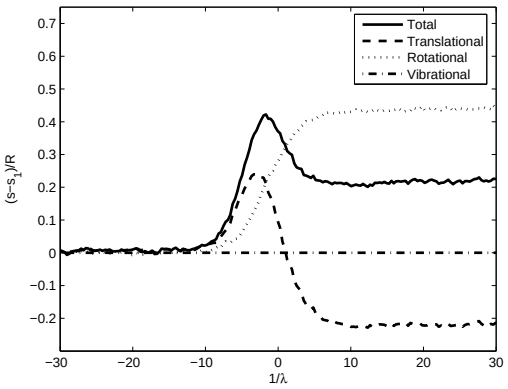
mode is somewhat less that that of the translational mode, and the vibrational mode is again seen to be inactive.



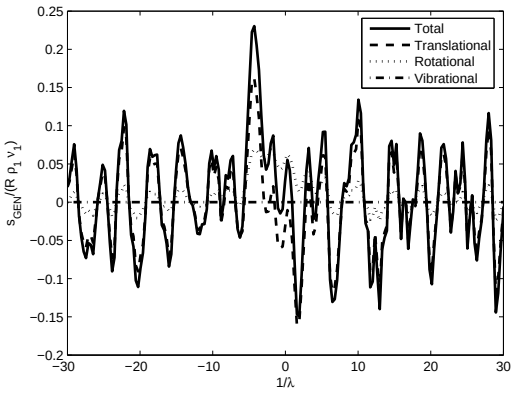
(a) Entropy Profiles



(b) Entropy Generation



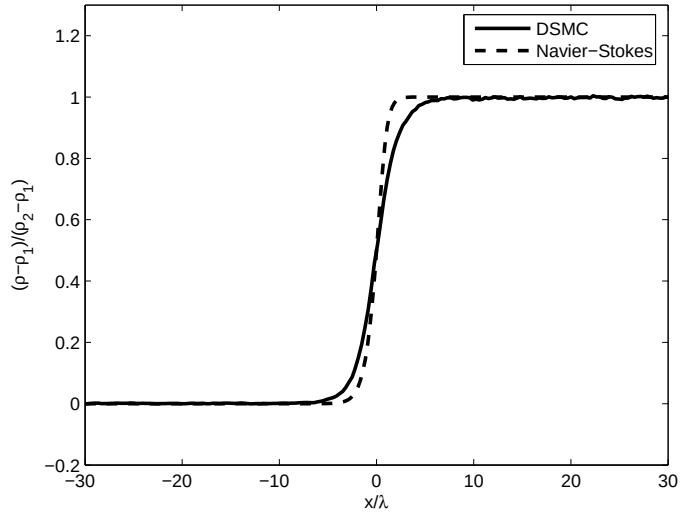
(c) Entropy Contributions



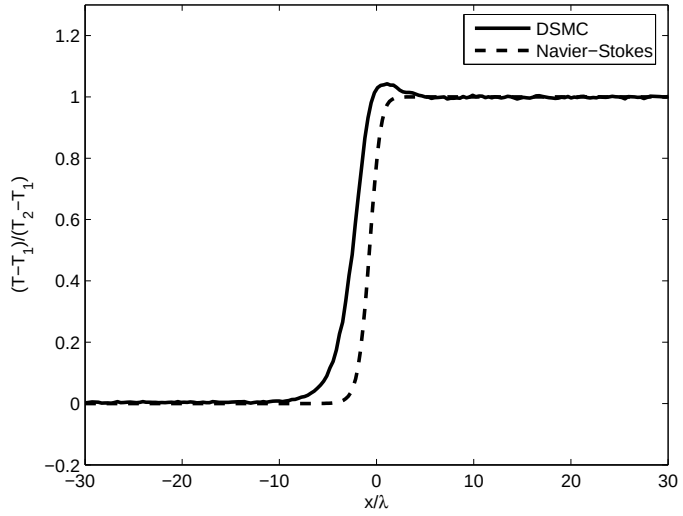
(d) Entropy Generation Contributions

Figure 19: Mach 1.75 Nitrogen Entropy and Entropy Generation Profiles

Figure 20 contains the density and temperature profiles for the Mach 2.5 shock. The Navier-Stokes shock remains too thin and the DSMC temperature profile is seen to have developed a slight overshoot which is not captured by the Navier-Stokes equations. This overshoot is by itself an indicator of the presence of significant nonequilibrium effects. As expected, the Navier-Stokes equations have broken down in this flow.



(a) Density Profiles



(b) Temperature Profiles

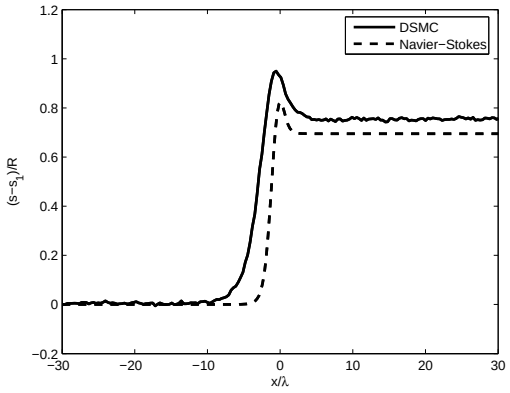
Figure 20: Mach 2.5 Nitrogen Density and Temperature Profiles

Figure 21 shows the entropy results for the Mach 2.5 shock. Again the peak entropy is seen to be higher than predicted by the Navier-Stokes equations. The entropy also begins to increase sooner in the DSMC data. The downstream value is again slightly different but likely correctable by adjusting the internal energy transfer

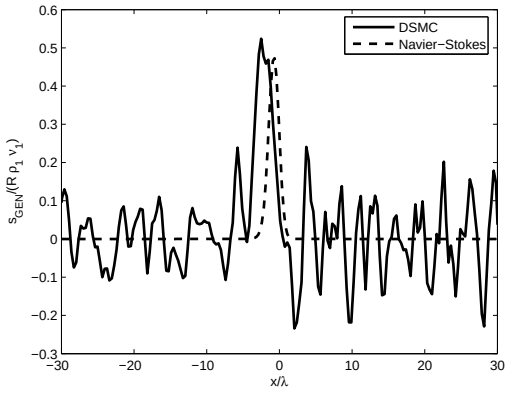
parameters. Entropy generation is seen to be of approximately the same magnitude for both cases, though the DSMC spike is slightly thicker and displaced slightly upstream signaling the presence of nonequilibrium before the Navier-Stokes equations. Figure 21(c) shows that the translational entropy drops only slightly across the shock in comparison with the previous cases. The entropy jump is achieved almost entirely by the increase in rotational entropy. The regions of translational and rotational nonequilibrium are very distinct in 21(d). Particularly, the region of rotational nonequilibrium has grown significantly over the previous case. The Navier-Stokes equations allow only for small perturbations from translational equilibrium and make no provision for rotational nonequilibrium at all. The significant rotational nonequilibrium seen here is a definite invalidation of these equations and they cannot adequately predict its effect. This is another reason the continuum equations have broken down.

The density and temperature results for the Mach 6.0 case is presented in Figure 22. The shock predicted by Navier-Stokes is far too thin. This causes the significant error in the temperature profile. No temperature overshoot is predicted by the Navier-Stokes. Downstream of the shock the vibrational mode has been activated, and is likely responsible for the discrepancy in downstream temperature, as the Navier-Stokes equations fail to account for the energy transferred from the translational mode to the vibrational mode. The activation of the vibrational mode brings another form of nonequilibrium to light that is not accounted for by the Navier-Stokes equations without further augmentation.

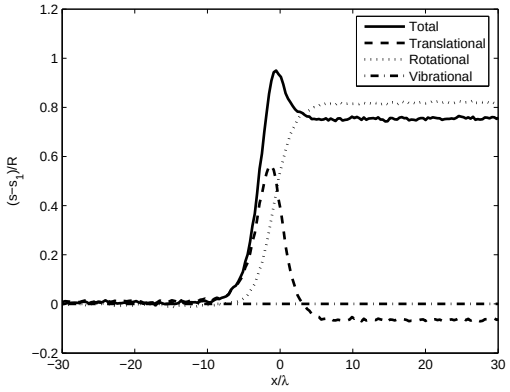
Figure 23 gives the various entropy results for the Mach 6.0 case. Here the discrepancy in the entropy profile is very large due to the extremely thin shock predicted by the Navier-Stokes equations. The peak entropy predicted by DSMC in Figure 23(a) is much higher than the Navier-Stokes result. Figure 23(b) clearly illustrates a much larger region of nonequilibrium exists than is predicted by the Navier-Stokes equations As in the argon cases, the shock predicted by the Navier-Stokes equations is so thin that the entropy generation rate must be significantly higher than reality in order to meet the jump conditions. In Figure 23(c), the translational mode is seen to



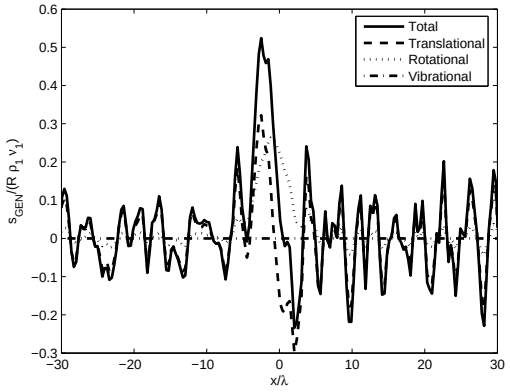
(a) Entropy Profiles



(b) Entropy Generation



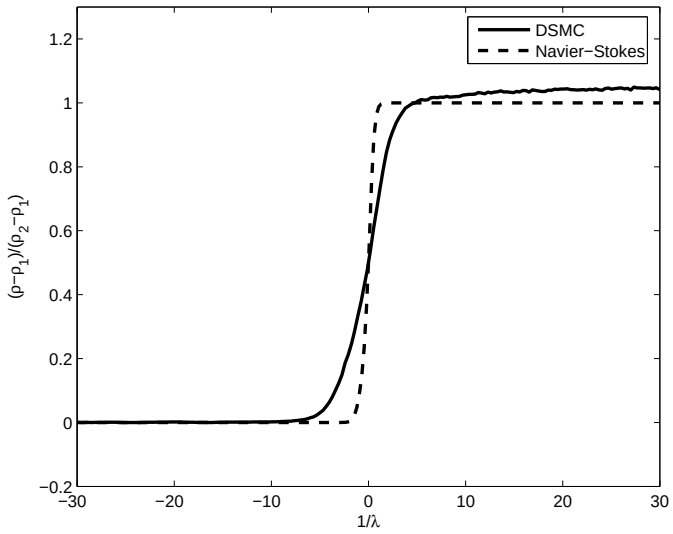
(c) Entropy Contributions



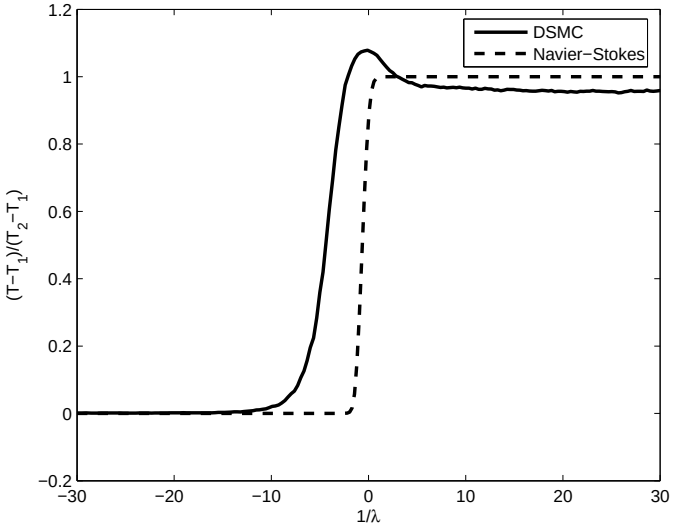
(d) Entropy Generation Contributions

Figure 21: Mach 2.5 Nitrogen Entropy and Entropy Generation Profiles

exhibit a positive change in value across the shock. The actual point where a positive change in translational entropy is observed occurs somewhere around Mach 3 for these conditions. Also it is seen that the vibrational mode is now partially activated and is now making a small contribution to the entropy which is not accounted for in the Navier-Stokes formulation.

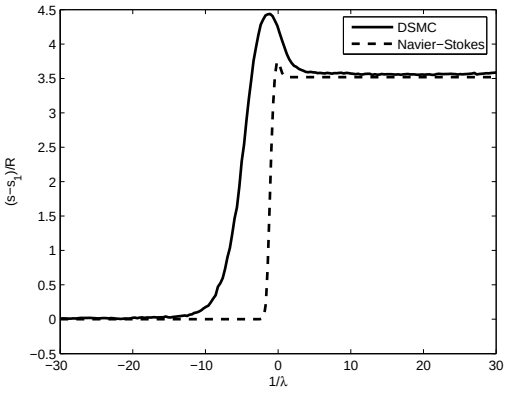


(a) Density Profiles

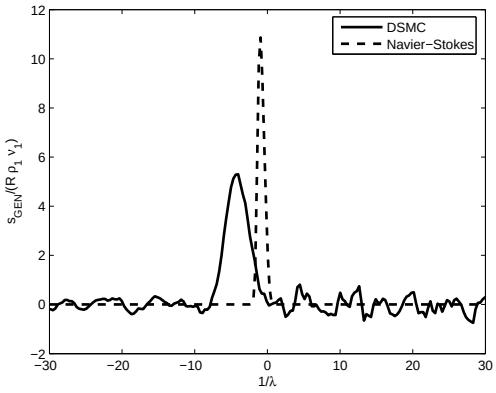


(b) Temperature Profiles

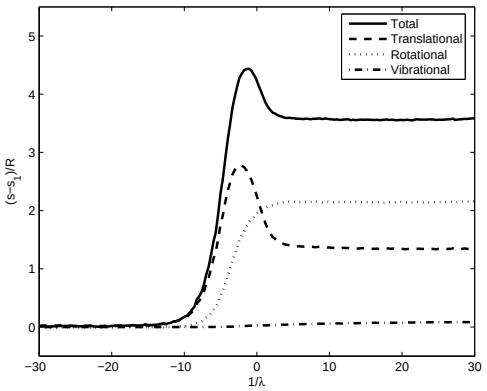
Figure 22: Mach 6.0 Nitrogen Density and Temperature Profiles



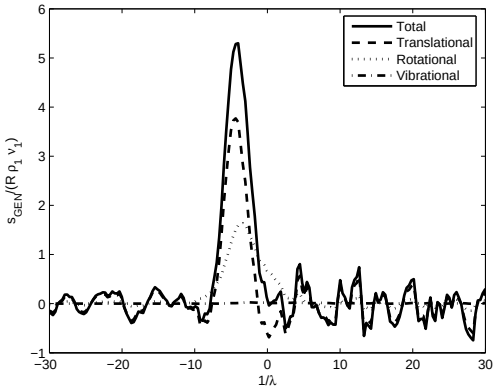
(a) Entropy Profiles



(b) Entropy Generation



(c) Entropy Contributions



(d) Entropy Generation Contributions

Figure 23: Mach 6.0 Nitrogen Entropy and Entropy Generation Profiles

As in the case of the argon shocks, reciprocal shock thickness was computed and compared to Alsmeyer's nitrogen data in Figure 24 below. Very good agreement is seen between the DSMC data and experiment at Mach numbers less than four. At Mach numbers of four and above, the DSMC data over predicts the shock thickness and falls outside the experimental scatter bounds. There are two likely causes for this. First, the random walk effect discussed in the previous section remains and continues to detract from the results. Secondly, Mach 4 happens to be about the Mach number where the vibrational mode begins to become activated. It is likely that

an adjustment to the vibrational probability given in Table 1 is required to achieve more reasonable results. This term had the most uncertainty associated with its use. A higher value should tend to thin out the shocks at higher Mach numbers because it will take fewer collisions to reach vibrational equilibrium. Another possible reason for the discrepancy with experimental data is that dissociation was not modeled in the computational results. Certainly at the higher Mach numbers, the temperature on the aft side of the shock is large enough to cause notable levels of dissociation in the experimental results. Nevertheless, this figure shows that the Navier-Stokes equations break down long before these errors are observed in the DSMC data. The DSMC data employed before Mach 4.0 is seen to agree quite well with experiment and is therefore valid for examining continuum breakdown.

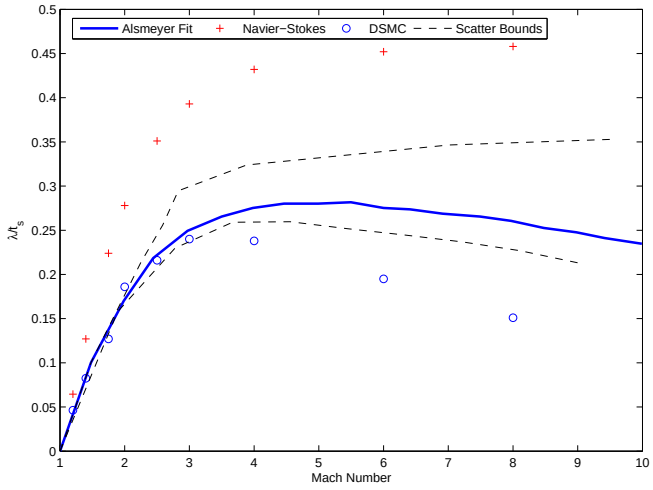


Figure 24: Reciprocal Shock Thickness in Nitrogen

Maximum flow variable error for the nitrogen shocks is plotted against Mach number in Figure 25. The error increases monotonically with Mach number and very little scatter is observed. At Mach numbers less than 1.75 maximum error is observed in the velocity, though the temperature dominates the error after that point. Less than ten percent error seems to occur at Mach numbers less than 1.75 which implies that the continuum equations are valid in this regime.

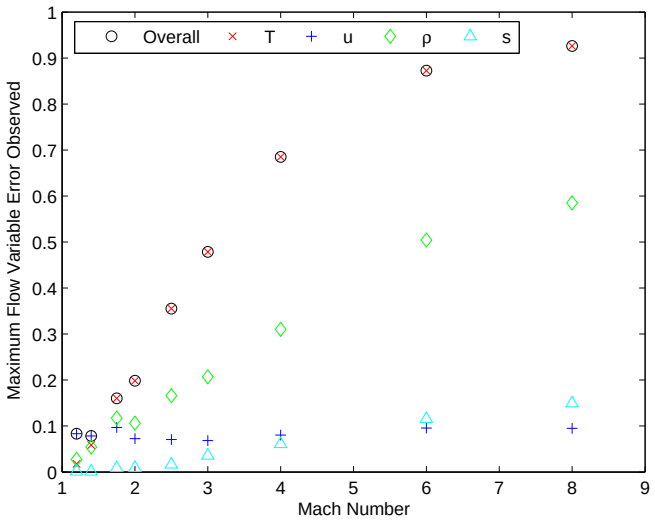


Figure 25: Maximum Error Observed in Flow Variables as a Function of Mach Number in Nitrogen Shocks

Maximum flow variable error is also plotted against peak entropy generation observed in the field in Figure 26. The error is seen to increase almost monotonically with entropy generation and very little scatter is observed. Here, a slightly more conservative bound is suggested, namely that the error appears to be less than ten percent for $\dot{\tilde{s}}/\rho s\nu \lesssim 0.005$. The proximity of this number to the one determined in the argon case may suggest that a value in this neighborhood may be a good indicator of the validity of the continuum equations, especially in the normal shock case. Simulations of different flow scenarios and other species are required to determine if such a value is universally applicable.

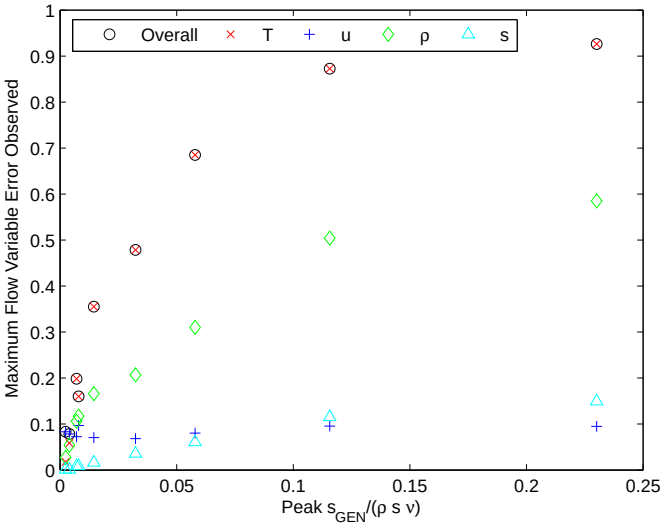


Figure 26: Maximum Error Observed in Flow Variables as a Function of Entropy Generation in Nitrogen Shocks

V. Conclusions and Future Work

The results of this research suggest that entropy generation is a valid parameter for examining the validity and breakdown of the continuum fluid equations. This parameter has stronger theoretical grounding than other parameters previously examined for this purpose. Formulations were presented for entropy generation due to translational, rotational and vibrational nonequilibrium.

The Navier-Stokes equations were shown to exhibit significant errors in normal shock wave flows, especially at Mach numbers greater than two, in both argon and nitrogen. Specifically, significantly thinner shocks were observed. A possible cause of this was seen to be the fact that the Navier-Stokes equations limit the degree of observable nonequilibrium across the shock. Nonequilibrium processes were seen to occur well upstream of where the Navier-Stokes equations predicted them to begin. This explains the difficulty in determining continuum breakdown in the shock front by using continuum data as has been done in the past. The continuum data delays any sign of nonequilibrium until well past the actual onset of nonequilibrium. This explains why even the entropy parameters examined by Camberos, Chen and Boyd [12, 15] were insufficient in capturing the shock front as they were based on continuum data. The results of this research imply that it is highly unlikely that any parameter computed using continuum data will perform adequately in the shock problem.

Maximum error observed in the flow variables was quantified in terms of entropy generation. The error was seen to be a strong function of this quantity, confirming its responsibility for the breakdown of the continuum equations. It was found that values of $\dot{\tilde{s}}/\rho s\nu$ less than 0.005 resulted in less than ten percent error in the Navier-Stokes results for all of the flow variables considered. Below this value, the Navier-Stokes equations produce acceptable results, with maximum error being observed in the velocity profile. Above this value, breakdown of the Navier-Stokes equations is evident and maximum error is observed in the temperature. This parameter could serve as a continuum onset parameter (computed using kinetic data) for use in a hybrid code, though examination of its consistency in other flow scenarios is warranted to

further quantify its general validity. This parameter would provide little advantage in determining continuum breakdown in a hybrid code because only continuum data would be available to compute this. Since the continuum equations limit the degree of observable nonequilibrium, there is likely no parameter which is totally effective at signaling breakdown using continuum data. This parameter does, however, explain the fundamental limitations of the continuum equations. Namely, rather than examining a somewhat arbitrary Knudsen number type limitation, it is seen that an entropy generation rate limitation may be used to quantify the validity of the continuum equations.

To better quantify the limitations of the Navier-Stokes equations in this regard, it is recommended that a number of other flow scenarios be examined. This will help to determine whether the limitation discussed above is universal or flow specific, and will establish an acceptable value which signifies the breakdown of the continuum equations. Study of entropy generation in boundary layers, oblique shocks and strong expansions will provide data from other relatively basic flows containing nonequilibrium. Like the normal shock, several theoretical and experimental resources are available for these flows. Following this, more realistic flows should be examined which posses multiple regions of nonequilibrium. These could consist of wedge, double cone, or blunt body flows. Additionally, the current results can be refined by inclusion of significantly more Mach numbers between, say, Mach 1.2 and 2.1. This will help to better quantify the behavior of the error as a result of the failure of the Navier-Stokes equations.

If further study of the normal shock at higher Mach numbers is desired, then action should be taken to reduce the random walk effect. The simplest method of controlling this will likely be to significantly decrease the statistical particle weight. This, however, may result in runs whose computational time is prohibitive. If this is the case, the specialized moving downstream boundary condition or stabilization subroutine of Bird [5] should be considered, however substantial effort may be required

to implement it within MONACO and a one-dimensional DSMC solver would be more desirable.

The discrepancy seen in the nitrogen shocks at higher Mach numbers is likely due to the parameters used for modeling the internal energy modes. The effect of varying these parameters should be further investigated and suitable values settled upon. Furthermore, in flows where the vibrational mode is significantly activated, the harmonic oscillator model used in the entropy calculations will become inaccurate and should be replaced with a more sophisticated model.

Finally, although MONACO is a parallel code, none of the calculations implemented in this research were written to be parallelized. To reduce the computational time involved with generating these results, the entropy calculations should be parallelized to take advantage of MONACO's parallel efficiency.

Bibliography

1. Alberty, R.A. and R.J. Silbey. *Physical Chemistry*. New York: Wiley, 1992.

2. Alsmeyer, H. "Density Profiles in Argon and Nitrogen Shock Waves Measured by the Absorption of an Electron Beam," *Journal of Fluid Mechanics*, *VOL. 74*:487–513 (July 1975).

3. Bird, G.A. "Breakdown of Translational and Rotational Equilibrium in Gaseous Expansions," *AIAA Journal*, *VOL. 8 NO.11*:1997–2003 (November 1970).

4. Bird, G.A. "Monte Carlo Simulation in an Engineering Context." *Rarefied Gas Dynamics VOL. 74*. Progress in Astronautics and Aeronautics, edited by S.S. Fisher, 239–255, New York: AIAA, 1981.

5. Bird, G.A. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*. New York: Oxford University Press, 1994.

6. Blazek, J. *Computational Fluid Dynamics: Principles and Applications*. Oxford, UK: Elsevier Science, 2001.

7. Bohm, D. *Quantum Theory*. New York: Dover Publications, 1951.

8. Boltzmann, L. *Lectures on Gas Theory*. Berkley: University of California Press, 1964. Orignially published in German under the title *Vorlesungen über Gastheorie* 1898. Translated by S.G. Brush.

9. Boyd, I.D. *Predicting the Breakdown of the Continuum Equations Under Rarefied Flow Conditions*. Proceedings of the 23^{rd} International Symposium on Rarefied Gas Dynamics, American Institute of Physics, May 2003.

10. Boyd, I.D. and Dietrich, S. "Scalar and Parallel Optimized Implementation of the Direct Simulation Monte Carlo Method," *Journal of Computational Physics*, *VOL. 126*:328–342 (September 1996).

11. Boyd, I.D., Chen, G. and Candler, G.V. "Predicting the Failure of the Continuum Fluid Equations in Transitional Hypersonic Flows," *Physics of Fluids*, *VOL. 7 NO. 1*:210–219 (January 1995).

12. Camberos, J.A., and Chen, P.H. *Continuum Breakdown Parameter Based on Entropy Generation Rates*. Paper 2003–157, AIAA, January 2003.

13. Carlson, H.A., Roveda, R., Boyd, I.D., and Candler, G.V. *A Hybrid CFD-DSMC Method of Modeling Continuum-Rarefied Flows*. Paper 2004–1180, AIAA, January 2004.

14. Chapman, S. and T.G. Cowling. *The Mathematical Theory of Non-Uniform Gases*. London: Cambridge University Press, 1970.

15. Chen, P.H., Boyd I.D. and Camberos, J.A. *Assessment of Entropy Generation Rate as a Predictor of Continuum Breakdown*. Paper 2003–3783, AIAA, June 2003.

16. Chung, C.H. *Kinetic Model Solution for Microscale Gas Flows*. Paper 2004–2590, AIAA, 2004.

17. Chung, C.H. *Numerical Simulation of Low-Speed Gas Flows in a Microfluidic System*. Paper 2004–2675, AIAA, 2004.

18. Cohen-Tannoudji, C. et al. *Quantum Mechanics*. Paris: Hermann, 1977.

19. Couture, L. and R. Zitoun. *Statistical Thermodynamics and Properties of Matter*. Amsterdam: Gordon and Breach, 2000.

20. Fiscko, K.A. and D.R. Chapman. "Comparison of Burnett, Super-Burnett and Monte Carlo Solutions for Hypersonic Shock Structure." *Rarefied Gas Dynamics: Theoretical and Computational Techniques VOL. 118*. Progress in Astronautics and Aeronautics, edited by D.P. Muntz, E.P. Weaver and D.H. Campbell, 374–395, Washington DC: AIAA, 1989.

21. Garcia, A.L., et al. "Adaptive Mesh and Algorithm Refinement Using Direct Simulation Monte Carlo," *Journal of Computational Physics*, *VOL. 154*:134–155 (1999).

22. Garen, W., Synofzig, and Frohn, A. "Shock Tube for Generating Weak Shock Waves," *AIAA Journal*, *VOL. 12 NO. 8*:1132–1134 (August 1974).

23. George, J.D. and Boyd, I.D. *Simulation of Nozzle Plume Flows Using a Combined CFD-DSMC Approach*. Paper 99–33880, AIAA, January 1999.

24. Kannenberg, K.C. and Wang, W. *MONACO Version 2.1 User's Guide*, September 2001.

25. Kolobov, H.Q., et al. *Unified Methods for Continuum and Rarefied Flows*. Paper 2004–1177, AIAA, January 2004.

26. Ladeinde, F., Cai, X., Li, W. and Agarwal, R. *A Unified Computational Methodology for Rarefied and Continuum Flow Regimes*. Paper 2004–1178, AIAA, January 2004.

27. Linzer, M. and Hornig, D.F. "Structure of Shock Fronts in Argon and Nitrogen," *Physics of Fluids*, *VOL. 6 NO. 12*:1661–1668 (December 1963).

28. Menzel, D.H., editor. *Fundamental Formulas of Physics, Volume One*. New York: Dover Publications, 1960.

29. Michaelis, C.H. *Development of a Continuum/Rarefied Hybrid Scheme for Flows with Thermal and Chemical Non-Equilibrium*. PhD dissertation, Stanford University, 2001.

30. Muckenfuss, C. "Some Aspects of Shock Structure According to the Bimodal Model," *Physics of Fluids*, *VOL. 5 NO. 11*:1325–1336 (November 1962).

31. Parker, J.G. "Rotational and Vibrational Relaxation in Diatomic Gases," *Physics of Fluids*, *VOL. 2*:449–462 (1959).

32. Pham-Van-Diep, G.C. and D.A. Erwin. "Validation of MCDS by Comparison of Predicted with Experimental Velocity Distribution Functions in Rarefied Normal Shocks." *Rarefied Gas Dynamics: Theoretical and Computational Techniques VOL. 118*. Progress in Astronautics and Aeronautics, edited by D.P. Muntz, E.P. Weaver and D.H. Campbell, 271–283, Washington DC: AIAA, 1989.

33. Robben, F. and Talbot, L. "Measurement of Shock Wave Thickness by the Electron Beam Fluorescence Method," *Physics of Fluids*, *VOL. 9 NO. 4*:633–643 (April 1966).

34. Schwartz, L.M. and Hornig, D.F. "Navier-Stokes Calculations of Argon Shock Wave Structure," *Physics of Fluids*, *VOL. 6 NO. 12*:1669–1675 (December 1963).

35. Sun, W. Wang Q. and I.D. Boyd. *Towards Development of a Hybrid DSMC-CFD Method for Simulating Hypersonic Interacting Flows*. Paper 2002–3099, AIAA, June 2002.

36. Sun, Q., and Boyd, I.D. "A Direct Simulation Method for Subsonic Microscale Gas Flows," *Journal of Computational Physics*, *VOL. 179*:400–425 (2002).

37. Sun, Q., Boyd, I.D., and Tatum, K.E. *Particle Simulation of Gas Expansions and Condensation in Supersonic Jets*. Paper 2004–2587, AIAA, 2004.

38. Vincenti, W.G. and C.H. Kruger. *Introduction to Physical Gas Dynamics*. New York: Wiley & Sons, Inc., 1967.

39. Wang, W. and I.D. Boyd. *Continuum Breakdown in Hypersonic Viscous Flows*. Paper 2002–0651, AIAA, January 2002.

40. Wannier, G.H. *Statistical Physics*. New York: Dover Publications, 1966.

41. White, F.M. *Viscous Fluid Flow*. McGraw-Hill, 1974.

Appendix A. Source Code Listings

A.1 pdfsetup.c

```
/*********************************************************
*                                                        *
*  pdfsetup.c - This routine was written by C.Schrock    *
*      to setup the PDF variables for a cell             *
*                               Created: 12/5/04         *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"
#include "physutil.h"
#include "spec.h"

void pdfsetup(int cellid,         /*Cell ID number*/
      float pdffactor,
      int pdfbins,
      int monodi,
      float req,
      float vibfreq,
      int rotstates)
{
  double avg,min,max,delta,stddev;
  int ispec=0;
  int i,nmax;
  double PLANCK,PLANCK2,amu2kg,mass;

  PLANCK=6.626075540E-34; /*J s*/
  PLANCK2=PLANCK/(2.0*PI);
  amu2kg=1.660538E-27;
  mass=species[0].mass*amu2kg;

  avg=(cellptr[cellid]->phys.sums[ispec][SUM_U])/
      (cellptr[cellid]->phys.sums[ispec][SUM_N]);
  stddev=sqrt(((cellptr[cellid]->phys.sums[ispec][SUM_UU])/
         (cellptr[cellid]->phys.sums[ispec][SUM_N]))-pow(avg,2));
  min=avg-pdffactor*stddev;
  max=avg+pdffactor*stddev;
  delta=(max-min)/pdfbins;
```

```
cellptr[cellid]->phys.xvdf=malloc(pdfbins*sizeof(float));
cellptr[cellid]->phys.xmid=malloc(pdfbins*sizeof(float));

cellptr[cellid]->phys.xmid[0]=min+delta/2;
cellptr[cellid]->phys.xmid[pdfbins-1]=max-delta/2;

for(i=1;i<pdfbins-1;i++)
  {
    cellptr[cellid]->phys.xmid[i]=cellptr[cellid]->phys.xmid[i-1]+delta;
  }
for(i=0;i<pdfbins-1;i++)
  {
    cellptr[cellid]->phys.xvdf[i]=0;
  }

avg=(cellptr[cellid]->phys.sums[ispec][SUM_V])/
    (cellptr[cellid]->phys.sums[ispec][SUM_N]);
stddev=sqrt(((cellptr[cellid]->phys.sums[ispec][SUM_VV])/
       (cellptr[cellid]->phys.sums[ispec][SUM_N]))-pow(avg,2));
min=avg-pdffactor*stddev;
max=avg+pdffactor*stddev;
delta=(max-min)/pdfbins;

cellptr[cellid]->phys.yvdf=malloc(pdfbins*sizeof(float));
cellptr[cellid]->phys.ymid=malloc(pdfbins*sizeof(float));

cellptr[cellid]->phys.ymid[0]=min+delta/2;
cellptr[cellid]->phys.ymid[pdfbins-1]=max-delta/2;

for(i=1;i<pdfbins-1;i++)
  {
    cellptr[cellid]->phys.ymid[i]=cellptr[cellid]->phys.ymid[i-1]+delta;
  }
for(i=0;i<pdfbins-1;i++)
  {
    cellptr[cellid]->phys.yvdf[i]=0;
  }

avg=(cellptr[cellid]->phys.sums[ispec][SUM_W])/
    (cellptr[cellid]->phys.sums[ispec][SUM_N]);
stddev=sqrt(((cellptr[cellid]->phys.sums[ispec][SUM_WW])/
       (cellptr[cellid]->phys.sums[ispec][SUM_N]))-pow(avg,2));
min=avg-pdffactor*stddev;
```

```
  max=avg+pdffactor*stddev;
  delta=(max-min)/pdfbins;

  cellptr[cellid]->phys.zvdf=malloc(pdfbins*sizeof(float));
  cellptr[cellid]->phys.zmid=malloc(pdfbins*sizeof(float));
  if (cellptr[cellid]->phys.zmid==NULL)
      mcexit("Memory limit reached while allocating cell PDFs");

  cellptr[cellid]->phys.zmid[0]=min+delta/2;
  cellptr[cellid]->phys.zmid[pdfbins-1]=max-delta/2;

  for(i=1;i<pdfbins-1;i++)
    {
      cellptr[cellid]->phys.zmid[i]=cellptr[cellid]->phys.zmid[i-1]+delta;
    }
  for(i=0;i<pdfbins-1;i++)
    {
      cellptr[cellid]->phys.zvdf[i]=0;
    }



  if(monodi==1)
    {
        avg=(cellptr[cellid]->phys.sums[ispec][SUM_ROT])/
    (cellptr[cellid]->phys.sums[ispec][SUM_N]);
min=0;
cellptr[cellid]->phys.rotpdf=malloc(rotstates*sizeof(float));
cellptr[cellid]->phys.rotmid=malloc(rotstates*sizeof(float));
for(i=0;i<rotstates;i++)
  {
    cellptr[cellid]->phys.rotmid[i]=pow(PLANCK2,2.0)*(i+1)*(i+2)/
                                          (2.0*(mass/4.0)*pow(req,2.0));
    cellptr[cellid]->phys.rotpdf[i]=0.0;
  }

avg=(cellptr[cellid]->phys.sums[ispec][SUM_VIB])/
    (cellptr[cellid]->phys.sums[ispec][SUM_N]);
/*printf("%d %e %e %e\n",cellid,avg,
         cellptr[cellid]->phys.sums[ispec][SUM_VIB],
                 cellptr[cellid]->phys.sums[ispec][SUM_N]);*/
min=0;
max=pdfbins*PLANCK*vibfreq;
```

```
delta=(max-min)/pdfbins;

cellptr[cellid]->phys.vibpdf=malloc(pdfbins*sizeof(float));
cellptr[cellid]->phys.vibmid=malloc(pdfbins*sizeof(float));

cellptr[cellid]->phys.vibmid[0]=min+delta/2;
cellptr[cellid]->phys.vibmid[pdfbins-1]=max-delta/2;

for(i=1;i<pdfbins-1;i++)
  {
    cellptr[cellid]->phys.vibmid[i]=cellptr[cellid]->phys.vibmid[i-1]+
    delta;
  }
  for(i=0;i<pdfbins-1;i++)
    {
      cellptr[cellid]->phys.vibpdf[i]=0;
    }
    }
}
```

A.2 pdfcalc.c

```
/*********************************************************
*                                                        *
*  pdfcalc.c - This routine was written by C.Schrock     *
*      to calculate the PDF variables for a cell         *
*                               Created: 12/5/04         *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"

void pdfcalc(int cellid,
     int pdfbins,
     int nobj,
     float v_x[MAXNOBJ],
     float v_y[MAXNOBJ],
     float v_z[MAXNOBJ],
     double e_trans[MAXNOBJ],
     double e_rot[MAXNOBJ],
     double e_vib[MAXNOBJ],
     int monodi,
     int rotstates)
{
  int status;
  int i,j;
  double low,high,delta,delta1,delta2;
  int pdfparticles;

  pdfparticles=cellptr[cellid]->phys.pdfparticles;

  delta=cellptr[cellid]->phys.xmid[1]-cellptr[cellid]->phys.xmid[0];
  for(i=0;i<pdfbins;i++)
    {
      cellptr[cellid]->phys.xvdf[i]=pdfparticles*delta*
(cellptr[cellid]->phys.xvdf[i]);
    }
  for(i=0;i<nobj;i++)
    {
      status=0;
```

```
      j=0;
      low=cellptr[cellid]->phys.xmid[0]-(delta/2);
      high=cellptr[cellid]->phys.xmid[pdfbins-1]+(delta/2);
      if(v_x[i]<low)
{
  cellptr[cellid]->phys.xvdf[0]=(cellptr[cellid]->phys.xvdf[0])+1.0;
}
      if(v_x[i]>high)
{
  cellptr[cellid]->phys.xvdf[pdfbins-1]=
    (cellptr[cellid]->phys.xvdf[pdfbins-1])+1.0;
}
      if((v_x[i]>=low)&&(v_x[i]<=high))
{
  while((status!=1)&&(j<pdfbins))
    {
      low=cellptr[cellid]->phys.xmid[j]-(delta/2);
      high=cellptr[cellid]->phys.xmid[j]+(delta/2);
      if((v_x[i]>=low)&&(v_x[i]<=high))
{
  cellptr[cellid]->phys.xvdf[j]=
    (cellptr[cellid]->phys.xvdf[j])+1.0;
  status=1;
}
      j++;
    }
}
    }


  delta=cellptr[cellid]->phys.ymid[1]-cellptr[cellid]->phys.ymid[0];
  for(i=0;i<pdfbins;i++)
    {
      cellptr[cellid]->phys.yvdf[i]=pdfparticles*delta*
(cellptr[cellid]->phys.yvdf[i]);
    }
  for(i=0;i<nobj;i++)
    {
      status=0;
      j=0;
      low=cellptr[cellid]->phys.ymid[0]-(delta/2);
      high=cellptr[cellid]->phys.ymid[pdfbins-1]+(delta/2);
      if(v_y[i]<low)
```

```
{
  cellptr[cellid]->phys.yvdf[0]=(cellptr[cellid]->phys.yvdf[0])+1.0;
}
      if(v_y[i]>high)
{
  cellptr[cellid]->phys.yvdf[pdfbins-1]=
    (cellptr[cellid]->phys.yvdf[pdfbins-1])+1.0;
}
      if((v_y[i]>=low)&&(v_y[i]<=high))
{
  while((status!=1)&&(j<pdfbins))
    {
      low=cellptr[cellid]->phys.ymid[j]-(delta/2);
      high=cellptr[cellid]->phys.ymid[j]+(delta/2);
      if((v_y[i]>=low)&&(v_y[i]<=high))
{
  cellptr[cellid]->phys.yvdf[j]=
    (cellptr[cellid]->phys.yvdf[j])+1.0;
  status=1;
}
      j++;
    }
}
    }

  delta=cellptr[cellid]->phys.zmid[1]-cellptr[cellid]->phys.zmid[0];
  for(i=0;i<pdfbins;i++)
    {
      cellptr[cellid]->phys.zvdf[i]=pdfparticles*delta*
(cellptr[cellid]->phys.zvdf[i]);
    }
  for(i=0;i<nobj;i++)
    {
      status=0;
      j=0;
      low=cellptr[cellid]->phys.zmid[0]-(delta/2);
      high=cellptr[cellid]->phys.zmid[pdfbins-1]+(delta/2);
      if(v_z[i]<low)
{
  cellptr[cellid]->phys.zvdf[0]=(cellptr[cellid]->phys.zvdf[0])+1.0;
}
      if(v_z[i]>high)
{
```

```
  cellptr[cellid]->phys.zvdf[pdfbins-1]=
    (cellptr[cellid]->phys.zvdf[pdfbins-1])+1.0;
}
      if((v_z[i]>=low)&&(v_z[i]<=high))
{
  while((status!=1)&&(j<pdfbins))
    {
      low=cellptr[cellid]->phys.zmid[j]-(delta/2);
      high=cellptr[cellid]->phys.zmid[j]+(delta/2);
      if((v_z[i]>=low)&&(v_z[i]<=high))
{
  cellptr[cellid]->phys.zvdf[j]=
    (cellptr[cellid]->phys.zvdf[j])+1.0;
  status=1;
}
      j++;
    }
}
    }

      if(monodi==1)
{
    delta=cellptr[cellid]->phys.rotmid[1]-
      cellptr[cellid]->phys.rotmid[0];
    for(i=0;i<rotstates;i++)
      {
/*cellptr[cellid]->phys.rotpdf[i]=pdfparticles*delta*
  (cellptr[cellid]->phys.rotpdf[i]);*/
cellptr[cellid]->phys.rotpdf[i]=pdfparticles*
  (cellptr[cellid]->phys.rotpdf[i]);
      }
    for(i=0;i<nobj;i++)
      {
status=0;
j=0;
low=0.0;
high=cellptr[cellid]->phys.rotmid[rotstates-1];
if(e_rot[i]<low)
  {
    cellptr[cellid]->phys.rotpdf[0]=
      (cellptr[cellid]->phys.rotpdf[0])+1.0;
  }
if(e_rot[i]>high)
```

```
  {
    cellptr[cellid]->phys.rotpdf[rotstates-1]=
      (cellptr[cellid]->phys.rotpdf[pdfbins-1])+1.0;
  }
if((e_rot[i]>=low)&&(e_rot[i]<=high))
  {
    while((status!=1)&&(j<rotstates))
      {
delta1=cellptr[cellid]->phys.rotmid[j]-
  cellptr[cellid]->phys.rotmid[j-1];
delta2=cellptr[cellid]->phys.rotmid[j+1]-
  cellptr[cellid]->phys.rotmid[j];
low=cellptr[cellid]->phys.rotmid[j]-(delta1/2.0);
high=cellptr[cellid]->phys.rotmid[j]+(delta2/2.0);
if(j==0)
  {
    low=0.0;
  }
if(j==rotstates-1)
  {
    high=cellptr[cellid]->phys.rotmid[j];
  }
if((e_rot[i]>=low)&&(e_rot[i]<=high))
  {
    cellptr[cellid]->phys.rotpdf[j]=
      (cellptr[cellid]->phys.rotpdf[j])+1.0;
    status=1;
  }
j++;
      }
  }
      }

    delta=cellptr[cellid]->phys.vibmid[1]-
      cellptr[cellid]->phys.vibmid[0];
    /*for(i=0;i<pdfbins;i++)
      {
cellptr[cellid]->phys.vibpdf[i]=pdfparticles*delta*
(cellptr[cellid]->phys.vibpdf[i]);
}*/
    for(i=0;i<pdfbins;i++)
      {
cellptr[cellid]->phys.vibpdf[i]=pdfparticles*
```

```
  (cellptr[cellid]->phys.vibpdf[i]);
      }
    for(i=0;i<nobj;i++)
      {
status=0;
j=0;
low=cellptr[cellid]->phys.vibmid[0]-(delta/2.0);
high=cellptr[cellid]->phys.vibmid[pdfbins-1]+(delta/2.0);
if(e_vib[i]<low)
  {
    cellptr[cellid]->phys.vibpdf[0]=
      (cellptr[cellid]->phys.vibpdf[0])+1.0;
  }
if(e_vib[i]>high)
  {
    cellptr[cellid]->phys.vibpdf[pdfbins-1]=
      (cellptr[cellid]->phys.vibpdf[pdfbins-1])+1.0;
  }
if((e_vib[i]>=low)&&(e_vib[i]<=high))
  {
    while((status!=1)&&(j<pdfbins))
      {
low=cellptr[cellid]->phys.vibmid[j]-(delta/2);
high=cellptr[cellid]->phys.vibmid[j]+(delta/2);
if((e_vib[i]>=low)&&(e_vib[i]<=high))
  {
    cellptr[cellid]->phys.vibpdf[j]=
      (cellptr[cellid]->phys.vibpdf[j])+1.0;
    status=1;
  }
j++;
      }
  }
      }
}
      pdfparticles=pdfparticles+nobj;
      cellptr[cellid]->phys.pdfparticles=pdfparticles;
}
```

A.3 getentropy.c

```
/*********************************************************
*                                                        *
*  getentropy.c - This routine was written by C.Schrock *
*      to calcuate the entropy contained in a cell via  *
*      Boltzmann's H-Theorem                            *
*                                Created: 9/20/04-10/3/04 *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"
#include "spec.h"
#include "physutil.h"

void getentropy(int cellid,        /*Cell ID number*/
     int nobj,          /*Number of simulated particles in cell*/
     float Wp,          /*Number of actual particles represented*/
     int monodi,
     float req,
     float vibfreq,
     int pdfbins,
     int rotstates)
{
  double numdensity; /*number density*/
  double volinv;     /*inverse cell volume*/
  double PLANCK;    /*Planck constant*/
  double PLANCK2;    /*h/2pi*/
  double amu2kg; /*conversion from amus to kilograms*/
  double binsize; /*bin size, used in discrete PDF formulations*/
  double entropy=0;
  double enttrans=0,entrot=0,entvib=0;
  double mass; /*particle mass*/
  double totmass;
  int i,j; /*loop counter*/
  double numparticles;
  double logterm;
  double reducedmass;
  double rotterm;
  double epsilon;
```

```
  double f,oldent,Nj,cj,lnOmega,nmax,nmin,emax,emin;
  int ispec=0;

  /*printf("Cell number = %i\n",cellid);*/
  PLANCK=6.626075540E-34; /*J s*/
  PLANCK2=PLANCK/(2.0*PI);
  amu2kg=1.660538E-27;
  mass=species[0].mass*amu2kg;
  volinv=cellptr[cellid]->geom.volinv;
  numparticles=nobj*Wp;
  /*  numparticles=(cellptr[cellid]->phys.sums[ispec][SUM_N])*Wp;*/
  numdensity=numparticles*volinv;
  totmass=mass*numparticles;
  reducedmass=mass/2;
#ifndef DIM_3D
   enttrans=BOLTZ*numparticles;
   if(nobj>1)
    {
      binsize=cellptr[cellid]->phys.xmid[1]-cellptr[cellid]->phys.xmid[0];
      for( i=0; i<pdfbins;i++)
{
  f=cellptr[cellid]->phys.xvdf[i];
  if (f>0)
    {
      enttrans=enttrans-BOLTZ*numparticles*(f*log(PLANCK*
pow(numdensity,(1.0/3.0))*f/mass))*binsize;
    }
}

      binsize=cellptr[cellid]->phys.ymid[1]-cellptr[cellid]->phys.ymid[0];
      for( i=0; i<pdfbins;i++)
{
  f=cellptr[cellid]->phys.yvdf[i];
  if (f>0)
    {
      enttrans=enttrans-BOLTZ*numparticles*(f*log(PLANCK*
        pow(numdensity,(1.0/3.0))*f/mass))*binsize;
    }
}

      binsize=cellptr[cellid]->phys.zmid[1]-cellptr[cellid]->phys.zmid[0];
      for( i=0; i<pdfbins;i++)
{
```

```
  f=cellptr[cellid]->phys.zvdf[i];
  if (f>0)
    {
      enttrans=enttrans-BOLTZ*numparticles*(f*log(PLANCK*
pow(numdensity,(1.0/3.0))*f/mass))*binsize;
    }
}
      if(monodi==1)
{
          /*entrot=BOLTZ*numparticles;*/
  /*entvib=BOLTZ*numparticles;*/

  binsize=cellptr[cellid]->phys.rotmid[1]-
    cellptr[cellid]->phys.rotmid[0];
  binsize=binsize;
  lnOmega=0.0;
  for(i=0;i<rotstates;i++)
    {
      f=(cellptr[cellid]->phys.rotpdf[i]);
      if(f>0)
{
  entrot=entrot-BOLTZ*numparticles*f*log(f/(2.0*i+1.0));
}
    }

          binsize=cellptr[cellid]->phys.vibmid[1]-
    cellptr[cellid]->phys.vibmid[0];
  lnOmega=0.0;
  if(binsize!=0)
    {
      for(i=0;i<pdfbins;i++)
{
  f=(cellptr[cellid]->phys.vibpdf[i]);
  if(f>0)
    {
      entvib=entvib-BOLTZ*numparticles*f*log(f);
    }
}
    }
}
    }
   else
     {
```

```
      entropy=0;
      enttrans=0;
      entrot=0;
      entvib=0;
    }



  enttrans=enttrans/totmass;
  entrot=entrot/totmass;
  entvib=entvib/totmass;


  if(monodi==1)
    {
      entropy=enttrans+entrot+entvib;
    }
  else
    {
      entropy=enttrans;
    }

  /*printf("Cell Number %i\n",cellid);
  printf("K*N/m %e\n",BOLTZ*numparticles/totmass);
  printf("totmass %e\n",totmass);
  printf("Translational Entropy = %e\n",enttrans);
  printf("Rotational Entropy = %e\n",entrot);
  printf("Vibrational Entropy = %e\n",entvib);
  printf("Entropy in fcn=%e\n",entropy);*/

  /*Shove entropy into cell structure*/

  cellptr[cellid]->phys.entropy=entropy;
  cellptr[cellid]->phys.transent=enttrans;
  cellptr[cellid]->phys.rotent=entrot;
  cellptr[cellid]->phys.vibent=entvib;

#else
  mcexit("3D ENTROPY CALC NOT IMPLEMENTED YET. Exiting...");
#endif

}
```

A.4 entropflux.c

```
/*********************************************************
*                                                        *
*  entropflux.c - This routine was written by C.Schrock*
*      to calcuate the entropy flux(velocity space)     *
*      for use in the entropy gradient calculation      *
*                              Created: 10/21/04         *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"
#include "physutil.h"
#include "spec.h"


void entropflux(int cellid,        /*Cell ID number*/
int nobj,          /*Number of simulated particles in cell*/
float Wp,          /*Number of actual particles represented*/
int pdfbins,
int monodi)
{
  float PLANCK=6.626075540E-34; /*J s*/
  float numdensity; /*number density*/
  float volinv;     /*inverse cell volume*/
  float xbin,ybin,zbin; /*bin size, used in discrete PDF formulations*/
  int i; /*loop counter*/
  float u1,u2,u3;  /* x, y, and z velocities*/
  float F1=0,F2=0,F3=0; /*Three components of flux*/
  float ent;
  float mass;
  float amu2kg=1.660538E-27;
  float fx,fy,fz,xmid,ymid,zmid;
  mass=species[0].mass*amu2kg;

  int ispec;     /*species number*/

  ispec=0; /*****ONLY ONE COMPONENT GASSES IMPLEMENTED SO FAR********/
  volinv=cellptr[cellid]->geom.volinv;
  numdensity=nobj*Wp*volinv;
```

```
#ifndef DIM_3D
  u1=cellptr[cellid]->phys.sums[ispec][SUM_U]/
    cellptr[cellid]->phys.sums[ispec][SUM_N];
  u2=cellptr[cellid]->phys.sums[ispec][SUM_V]/
    cellptr[cellid]->phys.sums[ispec][SUM_N];
  u3=cellptr[cellid]->phys.sums[ispec][SUM_W]/
    cellptr[cellid]->phys.sums[ispec][SUM_N];

  F1=BOLTZ*numdensity*u1;
  F2=BOLTZ*numdensity*u2;
  F3=BOLTZ*numdensity*u3;

  xbin=cellptr[cellid]->phys.xmid[1]-cellptr[cellid]->phys.xmid[0];
  ybin=cellptr[cellid]->phys.ymid[1]-cellptr[cellid]->phys.ymid[0];
  zbin=cellptr[cellid]->phys.ymid[1]-cellptr[cellid]->phys.zmid[0];

  for(i=0;i<pdfbins;i++)
    {
      fx=cellptr[cellid]->phys.xvdf[i];
      fy=cellptr[cellid]->phys.yvdf[i];
      fz=cellptr[cellid]->phys.zvdf[i];
      xmid=cellptr[cellid]->phys.xmid[i];
      ymid=cellptr[cellid]->phys.ymid[i];
      zmid=cellptr[cellid]->phys.zmid[i];

      if(fx>0)
{
  F1=F1-(BOLTZ*numdensity)*fx*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fx/mass))*xmid*xbin;
  F2=F2-(BOLTZ*numdensity)*u2*fx*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fx/mass))*xbin;
  F3=F3-(BOLTZ*numdensity)*u3*fx*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fx/mass))*xbin;
}
      if(fy>0)
{
  F1=F1-(BOLTZ*numdensity)*u1*fy*(log(PLANCK*
 pow(numdensity,(1.0/3.0))*fy/mass))*ybin;
  F2=F2-(BOLTZ*numdensity)*fy*(log(PLANCK*
                 pow(numdensity,(1.0/3.0))*fy/mass))*ymid*ybin;
  F3=F3-(BOLTZ*numdensity)*u3*fy*(log(PLANCK*
                 pow(numdensity,(1.0/3.0))*fy/mass))*ybin;
```

```
}
      if(fz>0)
{
  F1=F1-(BOLTZ*numdensity)*u1*fz*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fz/mass))*zbin;
  F2=F2-(BOLTZ*numdensity)*u2*fz*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fz/mass))*zbin;
  F3=F3-(BOLTZ*numdensity)*fz*(log(PLANCK*
                pow(numdensity,(1.0/3.0))*fz/mass))*zmid*zbin;
}

    }


  cellptr[cellid]->phys.F1tr=F1;
  cellptr[cellid]->phys.F2tr=F2;

  if(monodi==1)
    {
      ent=cellptr[cellid]->phys.rotent;
      F1=mass*numdensity*u1*ent;
      F2=mass*numdensity*u2*ent;
      cellptr[cellid]->phys.F1rot=F1;
      cellptr[cellid]->phys.F2rot=F2;

      ent=cellptr[cellid]->phys.vibent;
      F1=mass*numdensity*u1*ent;
      F2=mass*numdensity*u2*ent;
      cellptr[cellid]->phys.F1vib=F1;
      cellptr[cellid]->phys.F2vib=F2;
    }
  else
    {
      cellptr[cellid]->phys.F1rot=0;
      cellptr[cellid]->phys.F2rot=0;
      cellptr[cellid]->phys.F1vib=0;
      cellptr[cellid]->phys.F2vib=0;
    }
  if(monodi==1)
    {
      F1=cellptr[cellid]->phys.F1tr+cellptr[cellid]->phys.F1rot+
cellptr[cellid]->phys.F1vib;
      F2=cellptr[cellid]->phys.F2tr+cellptr[cellid]->phys.F2rot+
```

```
cellptr[cellid]->phys.F2vib;
    }

  cellptr[cellid]->phys.F1=F1;
  cellptr[cellid]->phys.F2=F2;


#else
  mcexit("3D ENTROPY CALC NOT IMPLEMENTED YET. Exiting...");
#endif

}
```

A.5 timederiv.c

```
/*********************************************************
*                                                        *
*  timederiv.c - This routine was written by C.Schrock   *
*      to calcuate the time derivative of entropy        *
*      a cell.                                           *
*                                Created: 10/21/04       *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"

float timederiv(int cellid, float val1, float val2, float dt)
{
  float td=(val2-val1)/dt;
  return(td);
}
```

A.6 spatialgrad.c

```
/*********************************************************
*                                                        *
*  spatialgrad.c - This routine was written by C.Schrock*
*      to calcuate the convective entropy derivatives   *
*      for use in the entropy generation calculation    *
*                              Created: 10/24/04         *
*********************************************************/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "../KERN/constants.h"
#include "../KERN/cell.h"
#include "../KERN/misc.h"


float spatialgrad(int cellid,        /*Cell ID number*/
  int numneigh,        /*Number of neighbors*/
  int numsides,        /*Number of sides of element*/
  int neighs[numsides], /*Neighbor designators*/
  int neighsides[numsides], /*Number of sides for neighbors*/
  int mode, /*Component of entropy to take gradient of*/
  float sums[MAXNSPEC][MAXNSUMS])/*0=total,1=trans,2=rot,3=vib*/


{
  float xi=0,yi=0;     /*x and y coordinates of current cell center*/
  float xj,yj; /*coordinates of neighbor centers*/
  /*  float gamma1=0,gamma2=0;
  float alpha1=0,alpha2=0,alpha3=0;
  float R11=0,R12=0,R22=0;*/
  float deltax,deltay;
  float ddx=0,ddy=0;
  float convect;
  int j,k;
  float F1=0, F2=0, F1neigh=0, F2neigh=0;
  float delF1,delF2;
  float a=0,b=0,c=0,d1=0,e1=0,d2=0,e2=0;
  float dF1dx=0,dF2dy=0;
  float S=0, Sneigh=0, dSdx,dSdy,u1,u2,delS,weight=1;
  int ispec;
  ispec=0;
```

```
  u1=sums[ispec][SUM_U]/sums[ispec][SUM_N];
  u2=sums[ispec][SUM_V]/sums[ispec][SUM_N];

   if(sums[ispec][SUM_N]>0)
    {
      u1=sums[ispec][SUM_U]/sums[ispec][SUM_N];
      u2=sums[ispec][SUM_V]/sums[ispec][SUM_N];
      xi=0;
      yi=0;
       for(j=0;j<numsides;j++)
{
  xi=xi+cellptr[cellid]->geom.x0[j];
  yi=yi+cellptr[cellid]->geom.y0[j];
}
      xi=xi/numsides;
      yi=yi/numsides;

      for(j=0;j<numneigh;j++)
{
  xj=0;
  yj=0;
  for(k=0;k<neighsides[j];k++)
    {
      xj=xj+cellptr[neighs[j]]->geom.x0[k];
      yj=yj+cellptr[neighs[j]]->geom.y0[k];
    }
  xj=xj/neighsides[j];
  yj=yj/neighsides[j];
  deltax=xj-xi;
  deltay=yj-yi;

  weight=1/sqrt(pow(deltax,2)+pow(deltay,2));

  if(mode==0)
    {
      F1=cellptr[cellid]->phys.F1;
      F2=cellptr[cellid]->phys.F2;
      F1neigh=cellptr[neighs[j]]->phys.F1;
      F2neigh=cellptr[neighs[j]]->phys.F2;
    }
  if(mode==1)
    {
      F1=cellptr[cellid]->phys.F1tr;
```

```
      F2=cellptr[cellid]->phys.F2tr;
      F1neigh=cellptr[neighs[j]]->phys.F1tr;
      F2neigh=cellptr[neighs[j]]->phys.F2tr;
    }
  if(mode==2)
    {
      F1=cellptr[cellid]->phys.F1rot;
      F2=cellptr[cellid]->phys.F2rot;
      F1neigh=cellptr[neighs[j]]->phys.F1rot;
      F2neigh=cellptr[neighs[j]]->phys.F2rot;
    }
  if(mode==3)
    {
      F1=cellptr[cellid]->phys.F1vib;
      F2=cellptr[cellid]->phys.F2vib;
      F1neigh=cellptr[neighs[j]]->phys.F1vib;
      F2neigh=cellptr[neighs[j]]->phys.F2vib;
    }
  delF1=F1neigh-F1;
  delF2=F2neigh-F2;


  a=a+pow(weight,2)*pow(deltax,2);
  b=b+pow(weight,2)*deltax*deltay;
  c=c+pow(weight,2)*pow(deltay,2);
  d1=d1+pow(weight,2)*delF1*deltax;
  e1=e1+pow(weight,2)*delF1*deltay;
  d2=d2+pow(weight,2)*delF2*deltax;
  e2=e2+pow(weight,2)*delF2*deltay;
}

      dF1dx=(c*d1-b*e1)/(a*c-pow(b,2));
      dF2dy=(-b*d2+a*e2)/(a*c-pow(b,2));
      convect=dF1dx+dF2dy;

    }
  else
    {
    convect=0;
    }
  return(convect);
}
```

A.7 Modifications to calc_cells.c

```
  /* Following varibles added by C.Schrock 9/20/04*/
  float v_x[MAXNOBJ],v_y[MAXNOBJ],v_z[MAXNOBJ];
  double e_rot[MAXNOBJ],e_vib[MAXNOBJ],e_trans[MAXNOBJ];
  float mass;
  float amu2kg=1.660538E-27;


  .......



      /*Added by C.Schrock 9/20/04 for PDF processing*/
      if(istep>startpdfs)
{
  v_x[iobj]=particles[iobj].Vx;
  v_y[iobj]=particles[iobj].Vy;
  v_z[iobj]=particles[iobj].Vz;
      e_trans[iobj]=mass*(pow(v_x[iobj],2)+pow(v_y[iobj],2)+
      pow(v_z[iobj],2))/2;
  if(monodi==1)
    {
      e_rot[iobj]=particles[iobj].Erot/AVOGADRO;
      e_vib[iobj]=particles[iobj].Evib/AVOGADRO;
    }
}


      .......



    /* Added by C.Schrock 9/20/04*/
    /* Calculate PDFs for current cell*/
    if(istep==startpdfs)
      {
cellptr[cellid]->phys.pdfparticles=0;
pdfsetup(cellid,pdffactor,pdfbins,monodi,req,vibfreq,rotstates);
      }

    if(((istep>=startpdfs)&&(istep%pdfinterval==0))&&(nobj>1))
      {
pdfcalc(cellid,pdfbins,nobj,v_x,v_y,v_z,e_trans,e_rot,e_vib,
```

```
monodi,rotstates);
      }

    if(istep==startent)
      {
cellptr[cellid]->phys.nument=0;
cellptr[cellid]->phys.entavg=0;
cellptr[cellid]->phys.transavg=0;
cellptr[cellid]->phys.rotavg=0;
cellptr[cellid]->phys.vibavg=0;
cellptr[cellid]->phys.entgenavg=0;
cellptr[cellid]->phys.transgenavg=0;
cellptr[cellid]->phys.rotgenavg=0;
cellptr[cellid]->phys.vibgenavg=0;
cellptr[cellid]->phys.q1avg=0;
cellptr[cellid]->phys.q2avg=0;
cellptr[cellid]->phys.q3avg=0;
      }

    if(istep>startent && (istep%entint==0||((istep-1)%entint)==0))
      {

int concurr=0;
if((istep-1)%entint==0)
  {
    concurr=1;
  }
if(nobj>1)
  {
    cellptr[cellid]->phys.entropyold=cellptr[cellid]->phys.entropy;
    cellptr[cellid]->phys.transentold=cellptr[cellid]->phys.transent;
    cellptr[cellid]->phys.rotentold=cellptr[cellid]->phys.rotent;
    cellptr[cellid]->phys.vibentold=cellptr[cellid]->phys.vibent;

    getentropy(cellid,nobj,Wp,monodi,req,vibfreq,pdfbins,rotstates);

    int entcount;
    entcount=cellptr[cellid]->phys.nument;

    cellptr[cellid]->phys.entavg=((cellptr[cellid]->phys.entavg)*
      entcount+(cellptr[cellid]->phys.entropy))/(entcount+1);
    cellptr[cellid]->phys.transavg=((cellptr[cellid]->phys.transavg)*
      entcount+(cellptr[cellid]->phys.transent))/(entcount+1);
```

```
     cellptr[cellid]->phys.rotavg=((cellptr[cellid]->phys.rotavg)*
                       entcount+(cellptr[cellid]->phys.rotent))/(entcount+1);
     cellptr[cellid]->phys.vibavg=((cellptr[cellid]->phys.vibavg)*
                       entcount+(cellptr[cellid]->phys.vibent))/(entcount+1);

     heatflux(cellid,nobj,Wp,monodi,pdfbins);
     cellptr[cellid]->phys.q1avg=((cellptr[cellid]->phys.q1avg)*
                       entcount+(cellptr[cellid]->phys.q1))/(entcount+1);
     cellptr[cellid]->phys.q2avg=((cellptr[cellid]->phys.q2avg)*
                       entcount+(cellptr[cellid]->phys.q2))/(entcount+1);
     cellptr[cellid]->phys.q3avg=((cellptr[cellid]->phys.q3avg)*
                       entcount+(cellptr[cellid]->phys.q3))/(entcount+1);


     entropflux(cellid,nobj,Wp,pdfbins,monodi);

     float deltat;

#ifdef OXFORD
     deltat=tstepREF*(cell->timesc);
#else
     deltat=tstepREF;
#endif

     cellptr[cellid]->phys.timederiv=timederiv(cellid,
     cellptr[cellid]->phys.entropyold,
                       cellptr[cellid]->phys.entropy,deltat);
     cellptr[cellid]->phys.tdtrans=timederiv(cellid,
                       cellptr[cellid]->phys.transentold,
                       cellptr[cellid]->phys.transent,deltat);
     if(monodi==1)
       {
 cellptr[cellid]->phys.tdrot=timederiv(cellid,
                       cellptr[cellid]->phys.rotentold,
                       cellptr[cellid]->phys.rotent,deltat);
 cellptr[cellid]->phys.tdvib=timederiv(cellid,
                       cellptr[cellid]->phys.vibentold,
                       cellptr[cellid]->phys.vibent,deltat);
       }

     int numneigh=0;
     int numsides=cellptr[cellid]->geom.nsides;
     int neighs[numsides];
```

```
    int neighsides[numsides];
    int ntest;
    int j;
    for(j=0;j<numsides;j++)
      {
ntest=cellptr[cellid]->neighbor[j];
if(ntest>0)
  {
    neighs[numneigh]=ntest;
    neighsides[numneigh]=cellptr[ntest]->geom.nsides;
    numneigh=numneigh+1;
  }
      }

    cellptr[cellid]->phys.convectentropy=spatialgrad(cellid,numneigh,
                              numsides,neighs,neighsides,0,cell->phys.sums);
    cellptr[cellid]->phys.contr=spatialgrad(cellid,numneigh,
                              numsides,neighs,neighsides,1,cell->phys.sums);
    cellptr[cellid]->phys.transentgen=cellptr[cellid]->phys.tdtrans+
      cellptr[cellid]->phys.contr;
   cellptr[cellid]->phys.transgenavg=
     (cellptr[cellid]->phys.transgenavg)*entcount+
     (cellptr[cellid]->phys.transentgen);
    if(monodi==1)
      {
cellptr[cellid]->phys.conrot=
  spatialgrad(cellid,numneigh,numsides,neighs,
      neighsides,2,cell->phys.sums);
cellptr[cellid]->phys.convib=
  spatialgrad(cellid,numneigh,numsides,neighs,
      neighsides,3,cell->phys.sums);
cellptr[cellid]->phys.rotentgen=cellptr[cellid]->phys.tdtrans+
  cellptr[cellid]->phys.contr;
cellptr[cellid]->phys.vibentgen=cellptr[cellid]->phys.tdtrans+
  cellptr[cellid]->phys.contr;
      }

    if(monodi==1)
      {
cellptr[cellid]->phys.rotgenavg=
  (cellptr[cellid]->phys.rotgenavg)*entcount+
  (cellptr[cellid]->phys.rotentgen);
cellptr[cellid]->phys.vibgenavg=
```

```
   (cellptr[cellid]->phys.vibgenavg)*entcount+
   (cellptr[cellid]->phys.vibentgen);
       }


     cellptr[cellid]->phys.entropgen=cellptr[cellid]->phys.timederiv+
       cellptr[cellid]->phys.convectentropy;

     /*Increment entropy counter*/
     entcount=entcount+1;
     cellptr[cellid]->phys.nument=entcount;
     /*Update averages*/
     cellptr[cellid]->phys.entgenavg=
       (cellptr[cellid]->phys.entgenavg)/entcount;
     cellptr[cellid]->phys.transgenavg=
       (cellptr[cellid]->phys.transgenavg)/entcount;

     if(monodi==1)
       {
 cellptr[cellid]->phys.rotavg=
   (cellptr[cellid]->phys.rotavg)/entcount;
 cellptr[cellid]->phys.vibavg=
   (cellptr[cellid]->phys.vibavg)/entcount;
 cellptr[cellid]->phys.rotgenavg=
   (cellptr[cellid]->phys.rotgenavg)/entcount;
 cellptr[cellid]->phys.vibgenavg=
   (cellptr[cellid]->phys.vibgenavg)/entcount;
       }






     if(istep==VDFstep && cellid==VDFcell)
       {
 printf("Outputting VDF at iteration %i and cell %i",
istep,cellid);
 FILE *xvdffile,*yvdffile,*zvdffile,*transfile;
 FILE *rotfile,*vibfile,*energyfile;
 float xmean=0,ymean=0,zmean=0,transmean=0,rotmean=0,vibmean=0;
 int i;
 for(i=0;i<nobj;i++)
   {
```

```
    xmean += v_x[i]/nobj;
    ymean += v_y[i]/nobj;
    zmean += v_z[i]/nobj;
    transmean += e_trans[i]/nobj;
    rotmean += e_rot[i]/nobj;
    vibmean += e_vib[i]/nobj;
  }

xvdffile=fopen("xVDF.dat","w");
yvdffile=fopen("yVDF.dat","w");
zvdffile=fopen("zVDF.dat","w");
        transfile=fopen("TRPDF.dat","w");
rotfile=fopen("ROTPDF.dat","w");
vibfile=fopen("VIBPDF.dat","w");
energyfile=fopen("energy.dat","w");
float f,c;
for(i=0;i<pdfbins;i++)
  {
    f=cellptr[cellid]->phys.xvdf[i];
    c=cellptr[cellid]->phys.xmid[i];
    fprintf(xvdffile,"%e %e\n",c,f);
  }
fclose(xvdffile);
for(i=0;i<pdfbins;i++)
  {
    f=cellptr[cellid]->phys.yvdf[i];
    c=cellptr[cellid]->phys.ymid[i];
    fprintf(yvdffile,"%e %e\n",c,f);
  }
fclose(yvdffile);
for(i=0;i<pdfbins;i++)
  {
    f=cellptr[cellid]->phys.zvdf[i];
    c=cellptr[cellid]->phys.zmid[i];
    fprintf(zvdffile,"%e %e\n",c,f);
  }
fclose(zvdffile);
/*  for(i=0;i<pdfbins;i++)
  {
    f=cellptr[cellid]->phys.transpdf[i];
    c=cellptr[cellid]->phys.transmid[i];
    fprintf(transfile,"%e %e\n",transmid[i],transpdf[i]);
  }
```

```
    fclose(transfile);*/
 for(i=0;i<rotstates;i++)
   {
     f=cellptr[cellid]->phys.rotpdf[i];
     c=cellptr[cellid]->phys.rotmid[i];
     fprintf(rotfile,"%e %e\n",c,f);
   }
 fclose(rotfile);
 for(i=0;i<pdfbins;i++)
   {
     f=cellptr[cellid]->phys.vibpdf[i];
     c=cellptr[cellid]->phys.vibmid[i];
     fprintf(vibfile,"%e %e\n",c,f);
   }
 fclose(vibfile);
 for(i=0;i<nobj;i++)
   {
     f=cellptr[cellid]->phys.xvdf[i];
     c=cellptr[cellid]->phys.xmid[i];
     fprintf(energyfile,"%e %e %e %e %e %e\n",v_x[i],v_y[i],
     v_z[i],e_trans[i],e_rot[i],e_vib[i]);
   }
 fclose(energyfile);

 printf("\nCell = %i\n",cellid);
 printf("# particles %i\n",nobj);
 printf("# X bins %i\n",pdfbins);
 printf("Mean X Velocity %e\n",xmean);
 printf("Mean Y Velocity %e\n",ymean);
 printf("Mean Z Velocity %e\n",zmean);
 printf("Mean Translational Energy %e\n",transmean);
 printf("Mean Rotational Energy %e\n",rotmean);
 printf("Mean Vibrational Energy %e\n",vibmean);
 printf("Entropy/mass in cell %e\n",
cellptr[cellid]->phys.entropy);
 printf("dsdt/mass = %e\n",cellptr[cellid]->phys.timederiv);
 printf("conv = %e\n",cellptr[cellid]->phys.convectentropy);
 printf("entropy gen/mass %e\n",cellptr[cellid]->phys.entropgen);
 printf("VDF file written");
       }

     /*printf(" %i %i %i %i %i\n",xboxes,yboxes,zboxes,rotboxes,
       vibboxes);*/
```

```
    /*      float xi=0,xj=0,yi=0,yj=0;
    for(j=0;j<numsides;j++)
      {
xi=xi+cellptr[cellid]->geom.x0[j];
yi=yi+cellptr[cellid]->geom.y0[j];
      }
    xi=xi/numsides;
    yi=yi/numsides;

    if((istep>=VDFstep)&&(cellid==3759))
      {
printf("Cell = %i\n",cellid);
printf("# particles %i\n",nobj);
printf("Cell center location %e,%e\n",xi,yi);
printf("Entropy/mass in cell %e\n",
cellptr[cellid]->phys.entavg);
printf("Old Entropy/mass in cell %e\n",
cellptr[cellid]->phys.entropyold);
printf("conv = %e\n",cellptr[cellid]->phys.convectentropy);
printf("entropy gen/vol %e\n",cellptr[cellid]->phys.entropgen);
printf("entropy gen %e\n",(cellptr[cellid]->phys.entropgen)/
(cellptr[cellid]->geom.volinv));
printf("u vel %e\n",cellptr[cellid]->phys.sums[0][SUM_U]/
cellptr[cellid]->phys.sums[0][SUM_N]);
printf("v vel %e\n\n",cellptr[cellid]->phys.sums[0][SUM_V]/
cellptr[cellid]->phys.sums[0][SUM_N]);


printf("***************NEIGHBOR DATA*******************\n");
int k,m;
for(k=0;k<numneigh;k++)
  {
    xj=0;
    yj=0;
    for(m=0;m<neighsides[k];m++)
      {
xj=xj+cellptr[neighs[k]]->geom.x0[m];
yj=yj+cellptr[neighs[k]]->geom.y0[m];
      }
    xj=xj/neighsides[k];
    yj=yj/neighsides[k];

    printf("Cell = %i\n",neighs[k]);
```

```
    printf("# particles %i\n",nobj);

    printf("Cell center location %e,%e\n",xj,yj);
    printf("Entropy/mass in cell %e\n",
    cellptr[neighs[k]]->phys.entavg);
    printf("Old Entropy/mass in cell %e\n",
    cellptr[neighs[k]]->phys.entropyold);

    printf("conv = %e\n",
    cellptr[neighs[k]]->phys.convectentropy);
    printf("entropy gen/vol %e\n",
    cellptr[neighs[k]]->phys.entropgen);
    printf("entropy gen %e\n",
    (cellptr[neighs[k]]->phys.entropgen)/
    (cellptr[neighs[k]]->geom.volinv));
    printf("u vel %e\n",
    cellptr[neighs[k]]->phys.sums[0][SUM_U]/
    cellptr[neighs[k]]->phys.sums[0][SUM_N]);
    printf("v vel %e\n\n",
    cellptr[neighs[k]]->phys.sums[0][SUM_V]/
    cellptr[neighs[k]]->phys.sums[0][SUM_N]);
    }
    }*/
  }


else
  {
    cellptr[cellid]->phys.entropyold=cellptr[cellid]->phys.entropy;
    cellptr[cellid]->phys.transentold=cellptr[cellid]->phys.transent;
    cellptr[cellid]->phys.rotentold=cellptr[cellid]->phys.rotent;
    cellptr[cellid]->phys.vibentold=cellptr[cellid]->phys.vibent;

    cellptr[cellid]->phys.entropy=0;
    cellptr[cellid]->phys.transent=0;
    cellptr[cellid]->phys.rotent=0;
    cellptr[cellid]->phys.vibent=0;
    cellptr[cellid]->phys.F1=0;
    cellptr[cellid]->phys.F2=0;
  }
      }
```

Vita

Christopher R. Schrock was raised in Ypsilanti, Michigan. He graduated from Willow Run High School, Ypsilianti, Michigan in 1998. Upon graduation he attended Washtenaw Community College and soon transferred to the distinguished University of Michigan College of Engineering to pursue a degree in Aerospace Engineering. During his senior year at the University of Michigan he served as an undergraduate research assistant in the Combustion Synthesis Laboratory, Department of Mechanical Engineering. In 2002 he graduated from the University of Michigan with a Bachelor of Science in Engineering, Aerospace Engineering with an academic minor in Mathematics.

In June 2002 Christopher accepted a position with the Aerospace Systems Design and Anyalysis Branch of the USAF Aeronautical Systems Center at Wright-Patterson AFB under the Palace Acquire program. In 2003 he returned to full time study at the Air Force Institute of Technology where he would pursue a Master of Science in Aeronautical Engineering. Upon graduation he will return to the Design Branch.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2005	Master's Thesis	Sep 2003 — Mar 2005

4. TITLE AND SUBTITLE

Entropy Generation as a Means of Examining Continuum Breakdown

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Schrock, Christopher R.

5d. PROJECT NUMBER

2004-144

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GAE/ENY/05-M20

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/VAAC
Dr. Jose Camberos
2210 8th Street Room 225
Wright-Patterson AFB, OH 45433-7512

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

To quantify the validity and breakdown of the continuum equations of fluid flow, the concept of entropy generation is examined. This parameter is formulated utilizing statistical mechanics and kinetic theory to avoid the use of equilibrium assumptions. This analysis leads to expressions in terms of energy distribution functions. These results are applied to monatomic and diatomic molecules. A numerical procedure for computing these values using the Direct Simulation Monte Carlo Method (DSMC) is presented. Normal shock waves in argon and nitrogen were simulated at Mach numbers ranging from 1.2 to 10. Results are compared to Navier-Stokes predictions. The Navier-Stokes equations are shown to delay the onset of nonequilibrium and diminish the magnitude of nonequilibrium though the shock. Because of this, breakdown parameters based on continuum data will fail to capture the initial nonequilibrium and will not provide good measures of continuum breakdown. Error in flow variables is shown as a strong function of entropy generation, suggesting this parameter is a good indicator of continuum breakdown and onset when computed using kinetic approaches.

15. SUBJECT TERMS

Continuum Breakdown, Entropy Generation, Nonequilibrium Kinetic Theory, Rarefied Gas Dynamics, Statistical Mechanics, Direct Simulation Monte Carlo (DSMC), Shock Waves, Quantum Mechanics, Translational Nonequilibrium, Rotational Nonequilibrium, Vibrational Nonequilibrium, Navier-Stokes Equations

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Richard J. McMullan, Maj, USAF (ENY)
U	U	U	UU	137	19b. TELEPHONE NUMBER *(include area code)* (937) 255–3636, ext 4578