3-22-2012

Validation of the Chemistry Module for the Euler Solver in Unified Flow Solver

William C. Humphrey Jr.

Follow this and additional works at: https://scholar.afit.edu/etd

Part of the Aerodynamics and Fluid Mechanics Commons

VALIDATION OF THE CHEMISTRY MODULE FOR THE EULER SOLVER IN UNIFIED FLOW SOLVER

THESIS

William C Humphrey Jr, Civilian, USAF

AFIT/GAE/ENY/12-M24

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT/GAE/ENY/12-M24

VALIDATION OF THE CHEMISTRY MODULE FOR THE EULER SOLVER IN UNIFIED FLOW SOLVER

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

William C. Humphrey Jr., BS

Civilian, USAF

March 2012

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT/GAE/ENY/12-M24

VALIDATION OF THE CHEMISTRY MODULE FOR THE EULER SOLVER IN UNIFIED FLOW SOLVER

William C. Humphrey Jr., MS

Civilian, USAF

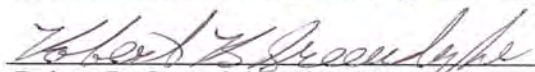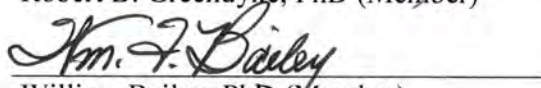Approved:

Andrew Lofthouse, Maj, USAF (Chairman)   3/8/12
                                                          Date

Robert B. Greendyke, PhD (Member)   3/9/12
                                                          Date

William Bailey, PhD (Member)   8 Mar '12
                                                          Date

AFIT/GAE/ENY/12-M24

Abstract

In the world of computational fluid dynamics (CFD) three main types of flow regimes exist; continuum, rarified, and free molecular. Of these regimes the rarified regime is the most difficult to model because the continuum equations don't apply and using the Boltzmann equation is too computationally expensive to use. Unified Flow Solver (UFS) is currently being developed to solve this problem by using the kinetic continuum Euler equations where valid and only using the Boltzmann equation where necessary, thus reducing the computational cost. The use of the kinetic Euler equations helps to aid in the coupling of the Euler equations with the Boltzmann equation. This work compares UFS with a common non-equilibrium solver, LeMANS, to attempt to validate the thermo-chemical Euler solver available in UFS. Three types of simulations were run to validate the Euler solver; perfect gas, thermal non-equilibrium, and thermo-chemical non-equilibrium. The perfect gas simulation was run using both a monatomic and two species diatomic gas. The thermal non-equilibrium simulation was run using a 2 species gas while the thermo-chemical non-equilibrium was run using 2 and 11 species. The results of the simulations show that UFS matches closely for both the monatomic and 2 species perfect gas simulations as well as the thermal non-equilibrium simulation. The thermo-chemical non-equilibrium simulations do not show the correct vibrational temperature which causes the species concentrations to not be correct. All of the simulations show that UFS is much slower than LeMANS in number of cpu hours. This makes UFS not a practical choice for a CFD solver and cannot be fully validated in its current state.

Acknowledgments

Table of Contents

List of Figures

List of Tables

VALIDATION OF THE CHEMISTRY MODULE FOR THE EULER SOVLER IN UNIFIED FLOW SOLVER

I. Introduction

The Air Force has many different areas in which it conducts research. A couple of specific areas of interest for the Air Force are upper atmospheric flight and re-entry vehicles. The reason for the Air Force's interest in upper atmospheric flight and re-entry is due to their involvement in NASA, putting up satellites, and hypersonic flight. The ability to send a vehicle or satellite into space requires the capacity to fly through the various stages of the atmosphere. Flight at hypersonic speeds requires air that is less dense, which occurs in the upper atmosphere, to reduce drag.

When traveling through the atmosphere there are three types of flow regimes that exist; the first is the continuum regime, second is the rarified regime and finally is the free molecular regime. The continuum regime occurs in the lower atmosphere close to the earth's surface and contains the densest air of the three regimes. The high density means that the air molecules are packed together tightly. When a perturbation away from equilibrium occurs, the flow returns to equilibrium so quickly that the different energy modes of the molecules can be modeled using a single energy equation. The rarified regime occurs in the middle to upper sections of the atmosphere. In this regime the density of the air is not as dense as in the continuum regime. Due to the lower density of the air, the time required to return to equilibrium after a perturbation away from equilibrium is much greater than the continuum regime. This extra time means that the different energy modes of the molecules must be modeled separately. The free-molecular

regime occurs outside the atmosphere, in space. The regime is characterized by air with such a low density that each molecule must be modeled individually, because the very low density causes the return to equilibrium, after a perturbation occurs, to take a very long time.

The distinction between each of the regimes is based on a non-dimensional number called the Knudsen number. The Knudsen number (Kn), which will be discussed in more detail in the next chapter, is a measure of how dense a gas has become relative to a give characteristic length. Figure 1 shows the ranges of Knudsen number and the equations that can be applied for each regime.
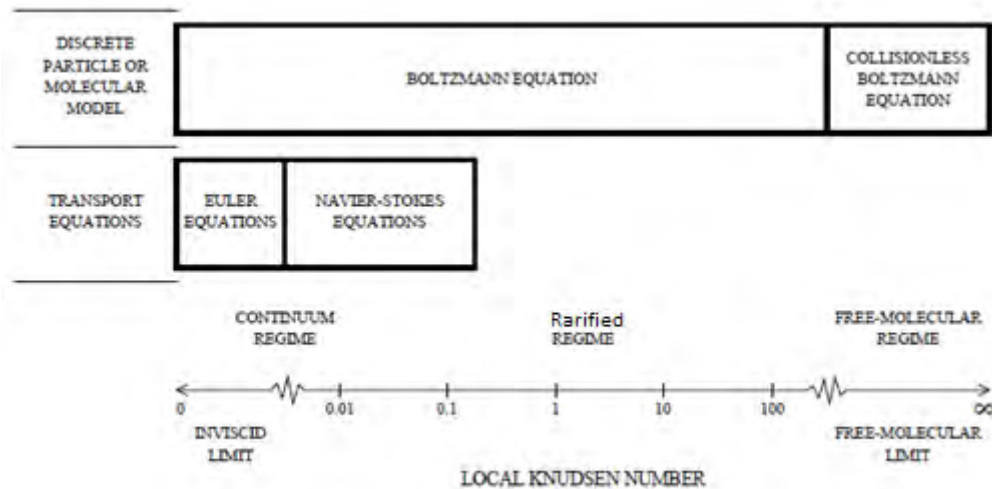
Figure 1. Flow Regimes Based on Knudsen Number [1]

More specifically at Kn≈0.1 the flow can no longer be modeled using continuum flow equations. Continuum flow solvers use the Euler equations, which are only valid for inviscid flow, and the Navier-Stokes equations. When the Knudsen number is between .01 and 100 the Boltzmann equation is applied and the flow is classified as the rarified

regime. Finally when the Kn>100 then the flow has moved in the free-molecular regime and a collisonless Boltzmann equation must be used.

Even though the Air Force has interest in and has done research on re-entry vehicle and upper atmospheric flight, it is very expensive to build a full size vehicle and do a test flight. To get around the cost of building and performing test flights the Air Force is using Computational Fluid Dynamics (CFD) to simulate the conditions analogous to upper atmosphere and re-entry. Using CFD, the Air Force can run simulations of test flights on the computer without the requirement of building a full size vehicle. The largest problem with using CFD is how to model the entire flight trajectory because the use of the Boltzmann equation to model the continuum regime is very computationally expensive. Research has been done and is currently being done to find a way around using the Boltzmann equation. The two main areas of focus are the Direct Simulation Monte Carlo (DSMC) method and reducing the computational cost of solving the Boltzmann equation directly.

DSMC was developed by G. A. Bird and "uses statistical modeling to predict the collisional behavior of a gas using a Monte Carlo scheme and then calculating the expected motion through the use of gas kinetics" [2]. DSMC can be used in both the continuum and rarified regime. The main benefit to using DSMC is that it requires less computational cost than solving the Boltzmann equation for flow in the rarified regime. The downside of DSMC is the computational cost if used in the continuum regime [2]. The high computational cost is due to the fact that DSMC was designed for the rarified regime and was also designed to simulate particles. As the density of the fluid increases

the number of particles DSMC simulates increases and this is what causes the high computational cost of DSMC in the continuum regime.

The second method has been used in a program called Unified Flow Solver (UFS). UFS is currently being developed under collaborative efforts the Air Force SBIR Phase II Project, CFD Research Corporation and Dorodnizyn Computing Center of the Russian Academy of Sciences [3]. UFS is a solver that combines a Boltzmann solver with a kinetic continuum solver to reduce the computational cost in both the continuum and rarified regimes. UFS uses the Boltzmann solver only where the flow is in the rarified regime and uses the continuum solver where the flow is in the continuum regime. UFS also reduces computational cost by implementing adaptive Cartesian mesh refinement, which refines the grid only where necessary and coarsens the grid where possible.

The purpose of this thesis is to test the progress of UFS by validating the 2 dimensional chemistry module for the Euler solver. The validation of UFS would reduce the computational expense of running transitional non-equilibrium simulations. The validation would also show the developers the strengths and weaknesses of UFS. To accomplish this, simulations will be run starting with a simple prefect gas case then moving to a thermal non-equilibrium case and finally to a thermo-chemical non-equilibrium case. The data from the simulations will be compared to a second code, LeMANS, that was previously validated for use with hypersonic simulations.

II. Background

2.1 Introduction

When trying to understand how to solve for a rarified regime flow one must first have an understanding of what happens inside a CFD code. This chapter will first cover a brief overview of the conservation equations, a discussion of kinetic theory, which will lead to a discussion of the Boltzmann equation. From there, this chapter will then go into the two codes used for this research, which include UFS and LeMANS.

2.2 Conservation Equations

The conservation equations describe the physical nature of a fluid and are the basis for all CFD codes. These equations are based on a set of three laws that give three properties (mass, momentum, and energy) that can never be created nor destroyed. The laws state that there cannot be more or less mass, momentum, or energy at the end then there was at the beginning.

The derivation of each equation starts with defining of a small fluid element of volume $d\Omega$ and a surface area of dS. The velocity of the flow through the element is equal to $\vec{v}$ and the unit normal is $\vec{n}$, where the arrow indicates a vector. The sign convention for $\vec{n}$ is flow into a surface is negative and flow out of a surface is positive due to the fact that $\vec{n}$ always points out of the control volume.

For the conservation of mass the conserved quantity is density, ρ [mass per volume], since the volume of the element is unchanging. Given the conserved quantity, the time rate of change inside the volume is

$$\frac{\partial}{\partial t} \int_\Omega \rho \, d\Omega$$

$$(2.1)$$

and the flow though a surface is

$$\rho(\vec{v} \cdot \vec{n}) dS.$$

$$(2.2)$$

The change in mass inside the volume, Equation (2.1), plus the mass leaving the volume, Equation (2.2), must come to zero so combining the two terms gives the full conservation of mass equation, Equation (2.3).

$$\frac{\partial}{\partial t} \int_\Omega \rho \, d\Omega + \rho(\vec{v} \cdot \vec{n}) dS = 0$$

$$(2.3)$$

The conserved quantity for the momentum equation is $\rho \vec{v}$ [momentum per volume] therefore the time rate of change inside the control volume is

$$\frac{\partial}{\partial t} \int_\Omega \rho \vec{v} \, d\Omega.$$

$$(2.4)$$

The transfer of momentum across the surface of the control volume is given by

$$-\oint_{d\Omega} \rho \vec{v}(\vec{v} \cdot \vec{n}) dS.$$

$$(2.5)$$

These two terms make up the left hand side of the momentum equation but unlike the conservation of mass the right hand side is not zero. For the momentum equation the right

hand side accounts for the forces acting on the fluid element. The reasons for the right

hand side being equal to the forces on the fluid are due to Newton's second law which

states that force is equal to the time rate of change of momentum. There are two types of

forces that act on the fluid element, body forces and surface forces. The body forces

include gravity or buoyancy and are described by $\vec{f_e}$. Therefore the total affect of the body

forces on the volume is

$$\int_{\Omega} \rho \vec{f_e}\, d\Omega.$$

$$(2.6)$$

The surface forces include pressure, shear stress and normal stress and the total

contribution is given in Equation 2.7. The first term accounts for the pressure and the

second term accounts for the stresses, where $\bar{\bar{\tau}}$ is the stress tensor.

$$-\oint_{d\Omega} p\vec{n}\, dS + \oint_{d\Omega} (\bar{\bar{\tau}} \cdot \vec{n})\, dS.$$

$$(2.7)$$

Equation (2.8) combines the five terms and gives the complete conservation of

momentum equation.

$$\frac{\partial}{\partial t}\int_{\Omega} \rho \vec{v}\, d\Omega - \oint_{d\Omega} p\vec{v}(\vec{v} \cdot \vec{n})\, dS = \int_{\Omega} \rho \vec{f_e}\, d\Omega - \oint_{d\Omega} p\vec{n}\, dS + \oint_{d\Omega} (\bar{\bar{\tau}} \cdot \vec{n})\, dS$$

$$(2.8)$$

The energy equation conserves the quantity E, which is the total energy, or

$e + \frac{|\vec{v}|^2}{2}$ [energy per volume], where e is the energy per unit mass. The time rate of change

inside the volume is

$$\frac{\partial}{\partial t} \int_{\Omega} \rho E \, d\Omega$$

(2.9)

and the energy leaving the surface is

$$-\oint_{d\Omega} \rho E (\vec{v} \cdot \vec{n}) dS.$$

(2.10)

These two terms complete the right hand side of the equation and the left hand side takes

into account the heat being added or removed, and the work done by the forces and

stresses. The heat being added or removed from the system is given by

$$\oint_{d\Omega} k(\Delta T \cdot \vec{n}) dS$$

(2.11)

where k is the thermal conductivity coefficient or the ability of the volume to conduct

heat and T is the temperature. The work done by the forces and stress is

$$\int_{\Omega} \left(\rho \vec{f_e} \cdot \vec{v} + \dot{q}_h \right) d\Omega - \oint_{d\Omega} \rho (\vec{v} \cdot \vec{n}) dS + \oint_{d\Omega} (\bar{\bar{\tau}} \cdot \vec{v}) \cdot \vec{n} \, dS$$

(2.12)

8

where $\dot{q}_h$ is the time rate of change of the heat transfer per unit mass. The first term in Equation (2.12) is the work done by the body forces, the second term is the work done by the pressure, and the third term is the work done by the stresses. The combination of the terms into the final form of the energy equation is shown in Equation (2.13).

$$\frac{\partial}{\partial t} \int_\Omega \rho E \, d\Omega + \oint_{d\Omega} \rho E(\vec{v} \cdot \vec{n}) dS = \oint_{d\Omega} k(\Delta T \cdot \vec{n}) dS + \int_\Omega (\rho \vec{f_e} \cdot \vec{v} + \dot{q}_h) d\Omega -$$
$$\oint_{d\Omega} p(\vec{v} \cdot \vec{n}) dS + \oint_{d\Omega} (\bar{\bar{\tau}} \cdot \vec{v}) \cdot \vec{n} \, dS$$

$$(2.13)$$

The set of conservation equations derived above are also called the Navier-Stokes equations. The Navier-Stokes equations are valid only in the continuum regime because of the assumption that the fluid is not made up of individual particles. The use of only one energy equation means that the molecules are close together that when a perturbation away from equilibrium occurs the different modes of energy, which will be talked about in the next section, return to equilibrium quickly.

Another set of equations that can be derived from the Navier-Stokes equations are the Euler equations, Equation (2.14-2.16). The Euler equations can be derived from the Navier-Stokes equations and assume viscosity and thermal conductivity do not exist in the flow field. Since the stresses in the Navier-Stokes equations are due to the viscosity in the fluid no stress terms occur in the Euler equations, which also means there is no heating due to stress. As with the Navier-Stokes equations, the Euler equations are only valid in the continuum regime but the Euler equations are even more restrictive. The

reason for the restrictiveness is due to viscosity and thermal conductivity being neglected, which causes there to be no transfer of mass, momentum or energy due to gradients.

$$\frac{\partial}{\partial t}\int_\Omega \rho \, d\Omega + \rho(\vec{v}\cdot\vec{n})dS = 0$$

(2.14)

$$\frac{\partial}{\partial t}\int_\Omega \rho\vec{v} \, d\Omega - \oint_{d\Omega} \rho\vec{v}(\vec{v}\cdot\vec{n})dS = \int_\Omega \rho\vec{f_e} \, d\Omega - \oint_{d\Omega} p\vec{n}dS$$

(2.15)

$$\frac{\partial}{\partial t}\int_\Omega \rho E \, d\Omega + \oint_{d\Omega} pE(\vec{v}\cdot\vec{n})dS = \int_\Omega (\rho\vec{f_e}\cdot\vec{v} + \dot{q}_h)d\Omega - \oint_{d\Omega} p(\vec{v}\cdot\vec{n})dS$$

(2.16)

Equations (2.14-2.16) can also be written in vector form as given in Equation (2.17)

$$\frac{\partial \mathbf{Y}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} + \frac{\partial \mathbf{H}}{\partial z} = 0$$

(2.17)

where

$$\mathbf{Y} = \{\rho, \rho u, \rho v, \rho w, E\}$$

$$\mathbf{F} = \{\rho u, P/2 + \rho u^2, \rho vu, \rho uw, u(E+P)\}$$

$$\mathbf{G} = \{\rho v, \rho uv, P/2 + \rho v^2, \rho vw, v(E+P)\}$$

$$\mathbf{H} = \{\rho w, \rho uw, \rho vw, P/2 + \rho w^2, w(E+P)\}$$

(2.18)

2.3 Kinetic Theory

When looking at individual molecules, as is done in the rarified regime, different modes of energy are available depending on whether the molecule is monatomic or diatomic. In a monatomic molecule there is translational and electronic energy in the x, y and z directions but with a diatomic molecule there is also rotational and vibrational energy along with the translational and electronic energy. When a perturbation away from equilibrium occurs, or non-equilibrium, each mode of energy requires a different number of collisions to occur before that mode returns to equilibrium.

In the continuum regime the molecules are tightly packed together and collisions occur very frequently. Since the collisions between molecules occur so frequently, the return to equilibrium is very rapid. The rapid return to equilibrium allows the different energy modes to be modeled using only one energy equation and the flow can be modeled looking at only the macroscopic properties. In the rarified regime, on the other hand, the distance between molecules is much larger and the different modes of energy return to equilibrium in different amounts of time. The difference in equilibration time between the different energy modes requires that each mode be modeled separately and the separate modeling means that each individual molecule is also important. Since each molecule is important, the rarified regime models the flow on a microscopic level in order to retrieve the macroscopic properties.

As mentioned in the previous chapter a parameter used to distinguish between the continuum and rarified regime is the Knudsen number. The Knudsen number is defined as

$$Kn = \frac{\lambda}{L}$$

(2.19)

where λ is the mean free path and L is a reference length that is based on the geometry. The mean free path of a molecule is defined as the average distance a molecule has to travel before a collision occurs. In terms of the flow regimes, the continuum regime has a very small mean free path due to the high density but the rarified regime has a large mean free path due to the lower density. The smaller mean free path in the continuum regime leads to small Knudsen numbers on the order of 0.1 or smaller while the rarified regime has a Knudsen numbers between 0.1 and 100 because of the large mean free path.

2.4 Boltzmann Equation

The Boltzmann equation is used to describe "the molecular motion of a system, which can be used to determine the overall behavior of that system"[2]. The molecular motion can be described by the use of velocity space. Velocity space is similar to physical space except that the coordinate axes are in units of velocity instead of units of length as in physical space. The coordinate axes in velocity space are labeled as c1, c2, and c3 and therefore the volume of a velocity element would be $dV_c=dc1dc2dc3$. The number of molecules with a given velocity class (c_i), assuming the velocities of the class differ by only a small amount, is:

$$dn = nf(c_i)dV_xdV_c$$

<div align="right">(2.20)</div>

where n is the number of molecules per volume, $f(c_i)$ is the velocity distribution function and dV_x is the volume of the physical space element.

A velocity function describes the probability of a molecule having a velocity, c_i, at a given position in physical space. A velocity function can only give a probability because of the impossibility of knowing the speed of a molecule and its position in physical space at the same time due to the number of molecules in a flow and the number of collisions. If the flow is in equilibrium the velocity function is called a Maxwellian distribution and is given by

$$f(c_i) = \left(\frac{m}{2\pi kT}\right)^{\frac{2}{3}} e^{-\frac{m}{2kT}(c1^2+c2^2+c3^2)}$$

<div align="right">(2.21)</div>

where m is the mass of the molecule, k is the Boltzmann constant, T is the temperature, and $f(c_i)$ is the distribution function.

Taking the derivative of Equation (2.20) gives the rate of change of the number of molecules inside the control volume.

$$\frac{\partial n}{\partial t} = \frac{\partial}{\partial t}(nf(c_i)dV_xdV_c)$$

<div align="right">(2.22)</div>

The change in the number of molecules would occur by either by molecules leaving dV_x or dV_c the or by collisions within dV_x. The flux of dV_c perpendicular to the j-direction of

<div align="center">13</div>

dV_x is given by $c_j nf(c_i)dV_c$. By applying the conservation of mass, Equation (2.3), to Equation (2.22) the equation for the net inward flux across the six surfaces of dV_x is given by:

$$-\frac{\partial}{\partial x_1}[c_1 nf(c_i)]dV_c dV_x - \frac{\partial}{\partial x_2}[c_2 nf(c_i)]dV_c dV_x - \frac{\partial}{\partial x_3}[c_3 nf(c_i)]dV_c dV_x$$
$$= -c_j\frac{\partial}{\partial x_j}[nf(c_i)]dV_c dV_x$$

(2.23)

where x1, x2, and x3 are the three directions in physical space.

Next is the convection across the surfaces of dV_c. An acceleration F_i is caused because of an external force mF_i per molecule and will alter the number of molecules in the velocity class. The density of the molecules can be defined as $(n\, dV_x)f(c_i)$ and the flux of molecules across the surfaces perpendicular to the velocity class is $F_j nf(c_i)\, dV_x$. Combining terms results in

$$-\frac{\partial}{\partial c_i}[F_j nf(c_i)]dV_x dV_c$$

(2.24)

and

$$\{\frac{\partial}{\partial t}[nf(c_i)]\}_{coll}dV_x dV_c$$

(2.25)

14

which gives the rate of change in the number of molecules of the velocity class resulting from collisions. Finally combining Equations 2.23-2.25 and doing some algebra gives

$$\frac{\partial}{\partial t}[nf(c_i)] + c_j\frac{\partial}{\partial x_j}[nf(c_i)] + \frac{\partial}{\partial c_i}[F_j nf(c_i)] = \{\frac{\partial}{\partial t}[nf(c_i)]\}_{coll}$$

(2.26)

which is the final result for the Boltzmann equation. The Boltzmann equation represents the entire behavior of a molecule within a given velocity class, c_j. The reason the Boltzmann equation is so computationally expensive is because of the term on the right hand side, which is referred to as the collision term. To solve the collision term, knowledge of the velocity states of the molecules before and after the collision are needed.

Looking at two different velocity classes, designated c and ζ, collisions will cause molecules to leave and enter the two velocity classes. These types of collisions are called depleting and replenishing collisions, respectively. By taking the sum of all of these collisions, the collision term can now be expressed as an integral:

$$\{\frac{\partial}{\partial t}[nf(c_i)]\}_{coll} = \int_{-\infty}^{\infty}\int_{0}^{2\pi}\int_{0}^{\pi/2} n^2[f(c_i')f(\zeta_i') - f(c_i)f(\zeta_i)]gd^2 \sin\varphi\cos\varphi\, d\varphi d\epsilon dV_\zeta$$

(2.27)

where c_i' and ζ_i' are the replenishing velocities, c_i and ζ_i are the depleting velocities, g is the relative velocity between the molecules, d is the radius of the sphere of influence, n is the number density, and ϵ and ψ are angles and define the location on the sphere where the collision occurred [4].

The assumptions that are made to derive the Boltzmann equation are that the density is low enough that only binary collisions occur and that the intermolecular forces are zero. The first assumption simply states that the molecules are far enough apart that only two molecules will ever collide at one time. This assumption limits the Boltzmann equation to being used in a dilute gas, such as the atmosphere, because in anything other than a dilute gas the fluid is too dense and binary collisions would not be the only type collisions that occur. The second assumption states that only collisions can change a molecules path or velocity and not just another molecules presence. This assumption limits the Boltzmann equation to temperatures above approximately 100K along with a dilute gas because at temperatures lower than this, the molecules would be moving very slow and the intermolecular forces would affect the path and velocity of a molecule. Also, if the fluid is too dense the molecules would be close enough for the intermolecular forces to make a difference.

Even with these limitations, the Boltzmann equation can be used in both the continuum and rarified regimes. The reason it is valid for both regimes is because it models the behavior of the molecules. For the rarified regime this is the way the flow must be solved because the molecules are so far apart. For the continuum regime, even though it can be solved using the Boltzmann equation, it is unnecessary and will cause the computational cost of a simulation to increase drastically.

2.5 UFS

   a. Grid

UFS uses a Cartesian grid while most other CFD solvers use structured or

unstructured grids. Structured grids are defined as a grid that even though the cells may

not be orthogonal in physical space they become orthogonal when mapped in

computational space. Figure 2 shows an example of how the mapping works with the

physical space on the right and the computational space on the left. The cells for a

structured 2D grid are always quadrilaterals and are good for boundary layers [5]. The

downside to a structured grid is that it is difficult to use with a complex geometry because

the cells become too skewed, which leads to incorrect results in simulations [5].
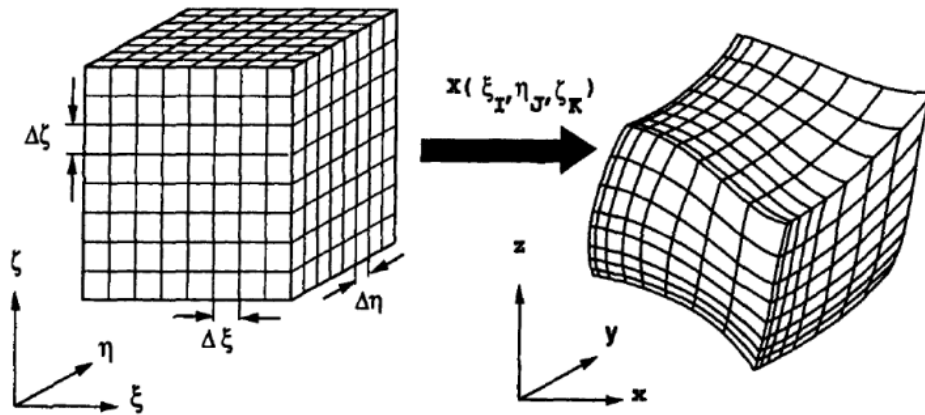
Figure 2. Example of Structured Grid Mapping [5]

Unstructured grids are not required to map to Cartesian coordinates and are

typically have the shape of a triangle for 2D. Figure 3 shows an example of an

unstructured mesh. The benefit to using an unstructured grid is that it is much easier to

use with a complex geometry and the downside to an unstructured grid is the larger

number of cells required to capture areas of high gradients due to highly skewed cells [5].

The Cartesian grid that UFS uses is defined by having the same mapping in computational and physical space and that the mapping is in Cartesian coordinates. A Cartesian grid is used for automatic grid refinement but is does not capture a viscous boundary layer well [5].
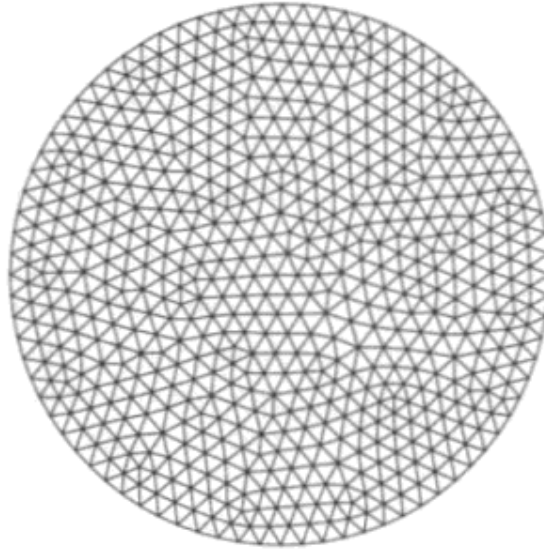
Figure 3. Example of Unstructured Grid [5]

Figure 4. Example of Cartesian Grid

b. Kinetic Euler Solver

Most CFD methods use the discretization of the Euler and Navier-Stokes equations to solve for a given flow field but a kinetic solver uses the Boltzmann Transport Equation (BTE) to develop the numerical solutions [6]. The idea for the kinetic Euler equations has been suggested by [7] and later on by [8][9] and [10][11]. It was first used by Deshpande et al [12]and then further developed by [13][14][15][16].

The kinetic Euler scheme used in UFS follows the equilibrium flux method that was set forth by Pullin [8], which is shown below. First consider the Euler equations for a monatomic gas

$$\frac{\partial \mathbf{Y}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} + \frac{\partial \mathbf{H}}{\partial z} = 0$$

(2.28)

where

$$\mathbf{Y} = \{\rho,\ \rho u,\ \rho v,\ \rho w,\ E\}$$

$$\mathbf{F} = \{\rho u,\ P/2 + \rho u^2,\ \rho v u,\ \rho u w,\ u(E+P)\}$$

$$\mathbf{G} = \{\rho v,\ \rho u v,\ P/2 + \rho v^2,\ \rho v w,\ v(E+P)\}$$

$$\mathbf{H} = \{\rho w,\ \rho u w,\ \rho v w\ ,\ P/2 + \rho w^2,\ w(E+P)\}$$

(2.29)

In **Y, F, G,** and **H;** $\rho = mn$ is the gas density, m is mass, n is the number density, u, v, and w are the velocity components in the x y and z directions respectively,

19

$E=3/2\rho T+\rho(u^2+v^2+w^2)$ is the total energy, T is temperature, and P is the pressure. When using a finite volume technique the discretization of the Equation (2.28) and (2.29) gives

$$\frac{Y_{ijk}^{n+1} - Y_{ijk}^{n}}{\Delta t} = -\left(\frac{F_{i+\frac{1}{2},j,k}^{n} - F_{i-\frac{1}{2},j,k}^{n}}{\Delta x} + \frac{G_{i,j+\frac{1}{2},k}^{n} - G_{i,j-\frac{1}{2},k}^{n}}{\Delta y} + \frac{H_{i,j,k+\frac{1}{2}}^{n} - H_{i,j,k-\frac{1}{2}}^{n}}{\Delta z} \right)$$

$$(2.30)$$

where Y_{ijk}^{n} is the cell averaged value of **Y** at time t^n, $F_{i-\frac{1}{2},j,k}^{n}$, $G_{i,j-\frac{1}{2},k}^{n}$, $H_{i,j,k-\frac{1}{2}}^{n}$ are the4 fluxes on the cell faces in the x, y, and z respectively. To calculate the fluxes $F_{i-\frac{1}{2},j,k}^{n}$, $G_{i,j-\frac{1}{2},k}^{n}$, and $H_{i,j,k-\frac{1}{2}}^{n}$ the integral had to be taken over the velocity distribution function, Equations (2.31-2.33),

$$F_{i-\frac{1}{2},j,k}^{n} = \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} \int_{R^3} \varphi\zeta_x f(x_{i+\frac{1}{2},j,k}, t, \zeta)d\zeta dt$$

$$(2.31)$$

$$G_{i,j-\frac{1}{2},k}^{n} = \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} \int_{R^3} \varphi\zeta_y f(y_{i,j+\frac{1}{2},k}, t, \zeta)d\zeta dt$$

$$(2.32)$$

$$H_{i,j,k-\frac{1}{2}}^{n} = \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} \int_{R^3} \varphi\zeta_z f(z_{i,j,k+\frac{1}{2}}, t, \zeta)d\zeta dt$$

$$(2.33)$$

where ψ is the collision invariants [17]. The velocity distribution at the cell faces has the form of

$$f\left(x_{i+\frac{1}{2}}, y_j, z_k, t, \zeta\right) = H[\zeta_x] f_M^l + (1 - H[\zeta_x]) f_M^r$$

<div align="right">(2.34)</div>

where f_M^l and f_M^r are Maxwellian distributions at the left and right side of the cell face, given by Equation (2.35),

$$f_M^r = \frac{\rho_{i+1/2}}{\left(\pi T_{i+\frac{1}{2}}^n\right)^{3/2}} \exp\left[-\frac{(\zeta_x - u_{i+\frac{1}{2},j,k})^2 + (\zeta_y - v_{i+\frac{1}{2},j,k})^2 + (\zeta_z - w_{i+\frac{1}{2},j,k})^2}{T_{i+\frac{1}{2}}^n}\right.$$

<div align="right">(2.35)</div>

and $H[\zeta_x]$ is the step function, Equation (2.36)

$$H[\zeta_x] = \begin{cases} 1, & \zeta > 0 \\ 0, & \zeta < 0 \end{cases}.$$

<div align="right">(2.36)</div>

For a first order scheme the macro-parameters at the cell faces [17] are calculated for f_M^l and f_M^r using the know values of the macro-parameters at the cell centers.

c. Chemistry

The chemistry in UFS is built into the conservation equations by first defining a pre-chemistry density. Then the density is used to come up with the conservative variables, rhou and rhov. Then using conservation of momentum, the conservative variables created using the pre-chemistry density were divided by a post-chemistry density to come up with the primitive variables, u and v. Equation (2.37) shows a pseudo code example of how this works, where rho is the pre-chemistry density, rho_chem is the

post-chemistry density, energyvib is the vibrational energy, energyrot is the rotational energy, and heat_form is the heat of formation.

$$rhou = rho*u$$

$$rhov = rho*v$$

$$rhoE = P/(\gamma - 1) + rho*(u^2 + v^2) + rho* \text{ energyvib } + rho* \text{ energyrot } + rho*heat\_form$$

$$u = rhou/rho\_chem$$

$$v = rhov/rho\_chem$$

$$energyvi = rhoEv/rho\_new$$

$$energyrot = rhoEr/rho\_new$$

$$P = (\gamma-1)*(rhoE - rho\_new*(u^2+v^2) - rho\_new*energyvib - rho\_new*energyrot - rho\_new*heat\_form)$$

(2.37)

Also shown in Equation (2.37) is how the pressure is defined, for a multiple species simulation, after the chemistry has taken place. The total temperature is defined

$$Tt = (rhoEt - rhot*(ut^2 + vt^2) - rhoEvt - rhoErt - rhoht)/K\_tr$$

(2.38)

where the t means the total quantity of each variable. The rate controlling temperature for the chemistry in UFS is defined by Park's two temperature model, which defines an average temperature using a combination of both the translational and vibrational temperatures [18].

22

The chemistry module in UFS uses another program called Cantera, which is an "object-oriented software for reacting flows" [19]. Cantera is a set of software tools, which can be used with several different program languages, for solving reacting flow problems. The chemical equilibrium uses an element potential method [20]. The element potential method dates back to 1959 and was used in NASA's equilibrium program in the early 60's [19]. During the 70's the idea was popularized in the combustion community by STANJAN code of Reynolds [19][21].

In the element potential method the element potentials are the chemical potentials of the atomic vapor species. Once the element potentials are given, any of the other chemical potentials can be computed using the equation of reaction equilibrium for the atomization reactions[19]. The partial pressures and total pressure are computed from the element potentials. The element potentials are adjusted until the pressure and the molecule compositions have the required values. This process requires solving a system of nonlinear algebraic equations. Variations of the Newton method work for solving the system of equations if the initial estimates are close[19].

2.6 LeMANS

The program that will be used to verify the results of UFS is called "Le" Michigan Aerothermodynamics Navier-Stokes Solver (LeMANS). The code was developed by Leonardo C. Scalabrin at the University of Michigan for the purpose of "the simulation of weakly ionized hypersonic flows in thermo-chemical non-equilibrium around entry configurations" [22]. LeMANS was chosen because of its ability to solve thermal and thermo-chemical non-equilibrium for either 2D or 3D continuum regime flows. LeMANS

uses a second order in time and space modified Steger-Warming flux vector splitting

scheme to solve the Navier-Stokes equations.

Since LeMANS is being used to validate UFS, LeMANS must have already been

validated. LeMANS has been compared to DSMC in many different conference papers

and for many different flow conditions. Lofthouse et al[23][24] validates LeMANS using

a cylinder at speeds of Mach 10 and Mach 25 for a flow of argon[24] and nitrogen[23].

The simulations run by Lofthouse et al[23][24] showed that LeMANS was within 8% of

DSMC. Other research by Schwartentruber et al [25][26][27] uses a 2D cylinder[25][27]

and a hollow cylinder flare[26] at various Mach numbers to compare LeMANS to

DSMC.

Table 1 shows a comparison of the capabilities of UFS and LeMANS. Even

though LeMANS uses the Navier-Stokes equations, LeMANS has the capability

off viscous effects and to also turn on an adiabatic boundary condition, which

equations equivalent to the Euler equations. A couple of other important notes

from

Table 1 are that first: LeMANS uses implicit time integration instead of explicit, which means LeMANS should be able to run faster than UFS because an implicit scheme allows for a larger time step without changing the solutions ability to converge. The second note is that UFS is non-dimensional, which means that each variable will have to be re-dimensionalized at the post-processing step.

Table 1. Capabilities of UFS and LeMANS

	UFS	LeMANS
Equations	Euler, Navier-Stokes, Boltzmann	Euler, Navier-Stokes
Time Integration	Explicit	Explicit, Implicit
Max Number of Species	11	11
Units	Non-Dimensional	Metric
Cases	Perfect Gas Thermal Non-Equilibrium Thermo-chemical Non-Equilibrium	Perfect Gas Thermal Non-Equilibrium Thermo-chemical Non-Equilibrium
Order	First and Second	First and Second
Flow Speed	Hypersonic	Hypersonic
Vibrational Relaxation	Millikan and White	Millikan and White
Rate Controlling Temperature	Park's Two Temperature model	Park's Two Temperature model
Grid	Cartesian	Structured

For the thermal and thermo-chemical non-equilibrium, LeMANS incorporates the assumption that the translational and rotational temperatures can be grouped together under a single temperature, and that the vibrational and electron translational temperatures can be combined under a single temperature as well. The translational and

rotational temperatures can be combined because both modes of energy equilibrate after only a few collisions. The vibrational and electron translational temperature can be combined because the transfer of energy between the electron translational mode and the vibrational mode is very fast in air [28], different molecules have very similar vibrational temperatures [29], and a single Maxwellian distribution can model both the electronic energy and the electron translational energy [22][30].

The chemistry solver uses Park's two temperature model [18] to account for vibrational non-equilibrium when calculating the forward and backward chemical rates. The forward rates are calculated using Arrhenius curve fits

$$k_{fk} = C_{fk} T_c^{\eta k} \exp{(-\theta_k / T_c)}$$

(2.39)

where Tc is the controlling temperature, and C_{fk}, $T_c^{\eta k}$, and θ_k are constants[18]. The subscript f means the rate is a forward rate and the k represents the given reaction. The backward rates are defined as

$$k_{bk}(T_{bc}) = \frac{k_{fk}(T_{bc})}{K_{eq}(T_{bc})}$$

(2.40)

where T_{bc} is the backward controlling temperature and K_{eq} is the equilibrium constant. The equilibrium constant is found by either using curve fits [18] or by using Gibbs free energy [22]. The normalized enthalpy and entropy are also obtained using curve fits [22].

The vibrational energy added or removed by chemistry is modeled using a preferential or non-preferential model [22]. The non-preferential model assumes that molecules are created or destroyed at an average vibrational energy. While the preferential model assumes that molecules are created or destroyed at higher vibrational

26

energy levels [22]. Both models are simplifications of a physical process that has no

models.

III. Methodology

3.1 Introduction

As stated in the introduction, the purpose of this thesis is to test the progress of UFS by validating the chemistry module for the Euler solver. The validation is done by comparing UFS with LeMANS with three different types of problems. The first type problem is a simple perfect gas simulation, the second is thermal non-equilibrium, and the third is full thermo-chemical non-equilibrium. The geometry for the validation process is a blunted wedge, Figure 5.
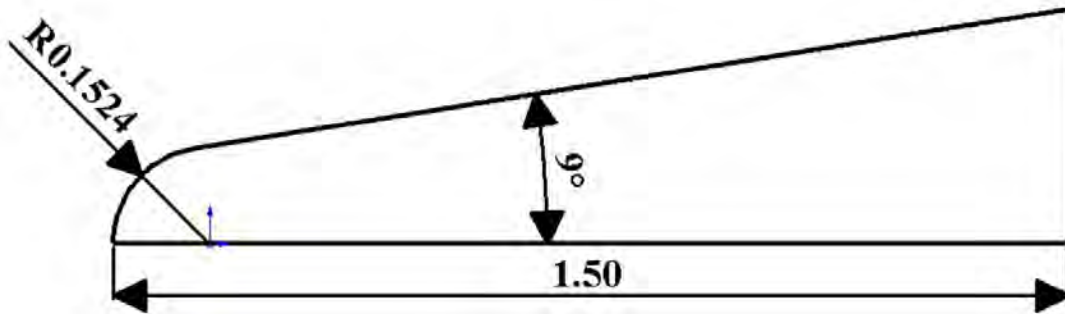
Figure 5. Blunted Wedge Geometry

All of the cases will be run at a speed of Mach 10. The conditions in each simulation are: a temperature of 300 K, a density of $2.816e^{-4}$ kg/m^3 and the pressure for each case will be calculated by UFS and LeMANS based on the species in the flow and the other initial conditions. The Knudsen number will change slightly depending on what species are included in the flow, but for the cases that will be run the Knudsen number is around a value of Kn≈0.002. This value of the Knudsen number is within the section of the continuum regime where the Euler equations are valid. An example input file for both LeMANS and UFS is in Appendix A.

There are some problems that arise when trying to run the single species diatomic

cases for both the perfect gas and the thermal non-equilibrium cases. The problem is UFS

does not have the capability to run a single species diatomic case. The reason for this is

UFS has gamma, which is the ratio of specific heats $\gamma = \frac{c_P}{c_v}$, hard coded to five thirds,

which is the gamma of a monatomic gas and UFS has no way of adjusting the gamma.

The only way UFS has to account for a diatomic species is to enable multiple species but

when multiple species are activated the vibrational and rotational energies must be

entered into the input file. Requiring the addition of the vibrational and rotational

energies causes problems for the perfect gas case because the vibrational energy should

not be activated. The pseudo code, Equation (3.1), below shows how UFS uses the

vibrational and rotational energies when multiple species are enabled, where energyvib is

the vibrational energy, energyrot is the rotational energy and heat_form is the heat of

formation.

if multiple_species=1

then $\rho E = P/(\gamma - 1.) + \rho*(u^2 + v^2) + \rho*$energyvib + rho*energyrot + rho*heat_form;

else $\rho E = P/(\gamma - 1.) + \rho*(u^2 + v^2)$

(3.1)

The use of multiple species only ended up causing problems for the diatomic perfect gas

simulation and the solution used will be discussed in the perfect gas section below,

section 3.5b.

3.2 Gridding and Grid Independence Study

      a. **LeMANS**

The gridding for LeMANS is done using a separate grid generation software program called Gridgen. The grid is created using a structured grid with constant cell spacing in the direction normal to the geometry, because of uncertainty in where the shock will be located in the domain. For the cell spacing along the wall the gridding criteria required that there was a high enough concentration of cells to capture the shock and stagnation region in front of the geometry.
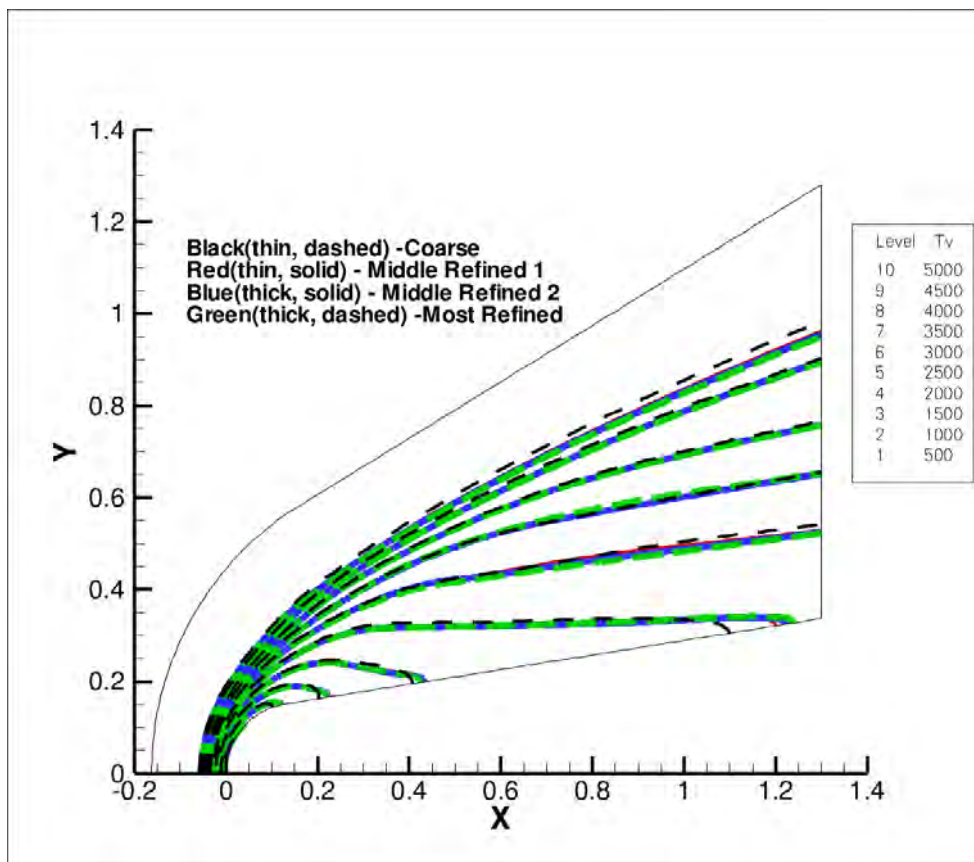
Figure 6. LeMANS Grid Study Contour Lines

A grid independence study was done to make sure the grid would not affect the solution. Grid independence is done by creating three more grids that are exactly like the initial grids except with different cell spacing in the direction normal to the geometry. One grid has larger cell spacing, decreasing the total number of cells, while the other two

30

grids have smaller spacing, increasing the total number of cells. The reason the streamwise direction was not changed is because the focus of the results are on the shock location and the shock location would not be affected by the streamwise cell spacing. Also the streamwise direction had been refined to remove highly skewed cells around the blunted portion of the geometry. Each of the grids is used in a thermal non-equilibrium simulation and compared against each other using pressure, temperature and density. For succinctness only the vibrational temperature is shown. Figure 6 shows the contour lines while Figure 7 shows the stagnation and surface line, where the stagnation point is at x=0 and is indicated by the line.
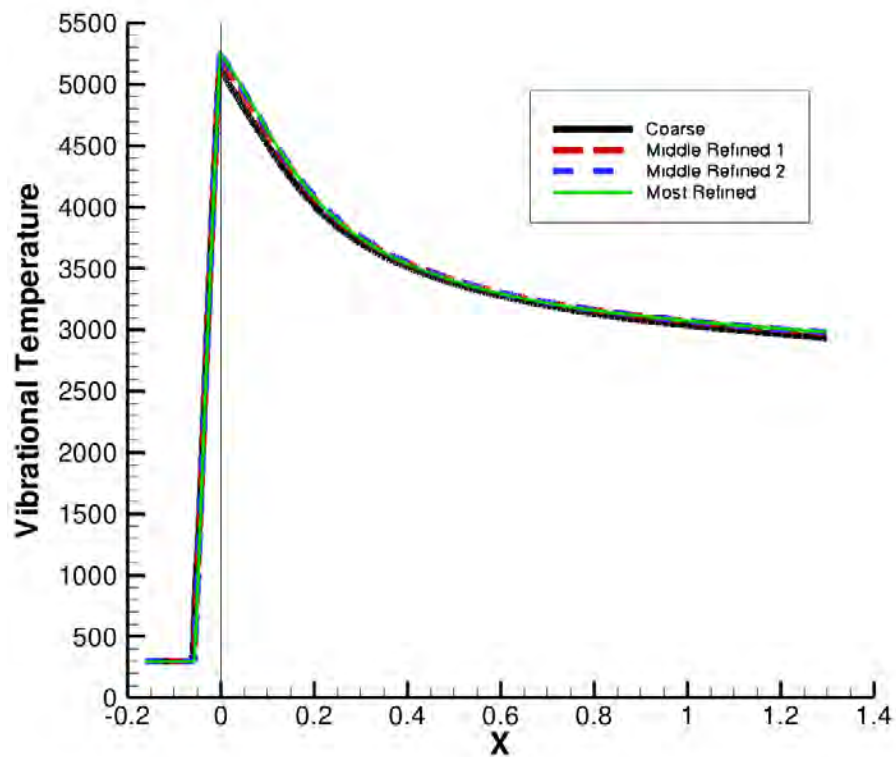
Figure 7. LeMANS Grid Study Stagnation Line

The results from each grid are relatively close with almost no change between the two most refined grids. Since there is such close agreement between the two most refined

grids, the coarser of the two grids was chosen so as to reduce computational cost. Figure 8 shows the final grid, which has 225 cells in the direction normal to the wall and 184 along the wall and a total cell count of almost 41,000 cells.
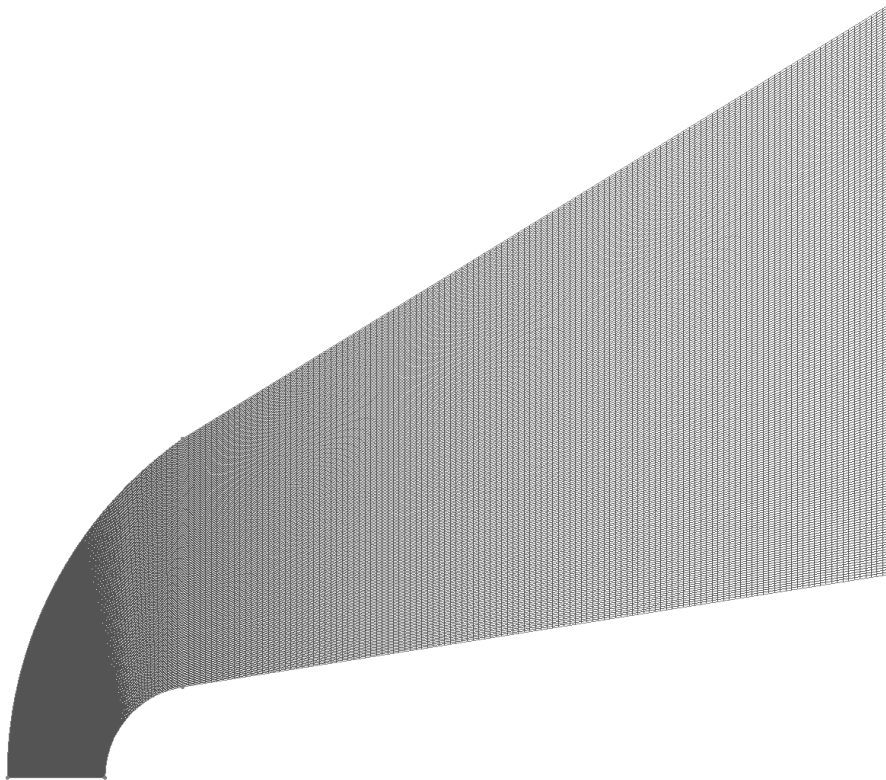
Figure 8. LeMANS Final Grid

b. UFS

The grid generation in UFS is started by adding blocks of length one unit, shown in Figure 9 as the largest cell, together until the grid is large enough to capture geometry and all the flow features. Once the initial grid layout is set, the grid is then refined by using the command Refine in the input file. Refine works by taking the initial blocks and dividing then into four new blocks then dividing each of the new smaller blocks into four more blocks until it reaches the user specified level of refinement. An example of this is shown for an initial two block grid with a refinement level of three, Figure 9. The final

step in the initial grid setup is the grid refinement around the geometry and the command that is used is RefineSolid. For the test cases, the initial level of refinement was set to 5 for the grid and 11 around the body.

Figure 9. Example of UFS Grid Refinement

Since UFS has an automatic grid refinement that runs while the solution is being calculated, after the initial grid refinement is set the grid adaptation parameters will be set. The grid adaptation parameters are the min and max level of refinement, the equation that controls the refinement, and when and how often the refinement will be done. For the test cases, the min level of refinement was set to 0 so there are not a larger amount of cells where they are not needed and the max level of refinement is set to 11 around the stagnation region but is set to 10 everywhere else in the grid. The reason behind this is the most important features are in the shock and stagnation region, higher refinement anywhere but the stagnation region will add significant computational cost. The first refinement is done after 100 iterations and will be done after every 100 iterations. Finally the equation used is

$$\log(\rho) + \log(V) = Cmax$$

(3.1)

33

where ρ is the density, V is the total velocity, and Cmax is a constant. The value of Cmax is a threshold value that initiates grid refinement. If the value of Equation (3.1), in a cell, is greater than the value of Cmax then the cell will be refined. On the other side if the value of Equation (3.1) is lower than Cmax, in four cells that share a corner, then those four cells will be reduced to one.

Even though UFS has automatic grid refinement a grid independence study will be done because of the user specified parameters required for grid refinement. The two important parameters are the equation and the max level of refinement. For the equation the value of Cmax is set to 0.1, 0.05, and 0.01 and for the max level of refinement is set to 10, 11, and 12.

Figure 10 and Figure 11 show the temperature results of the Cmax variation and Figure 12 and Figure 13 show the temperature results of the max refinement level variation. The variation of Cmax shows that a value of 0.01 has the best results while the max refinement level shows no differences between each level except for in the shock region. Figure 14 shows a close up of the shock region and shows that as the level of refinement increases the shock moves closer to the body of the geometry, where the front of the geometry is located at zero. The arrows in Figure 14 shows the location of the halfway point of the shock and the brackets show the thickness of each shock. The fact that the shock is still changing means the solution is not grid independent. To reach a grid independent solution the max level of refinement would need to be increase until the solution no longer changed. A fully independent grid was not found because of the computational resources required to run at the higher refinement levels were not

available. The level 12 refinement ran for 1521.8 cpu hours and had a total cell count of

almost 649,000 cells, which is a order of magnitude higher than the final LeMANS grid.
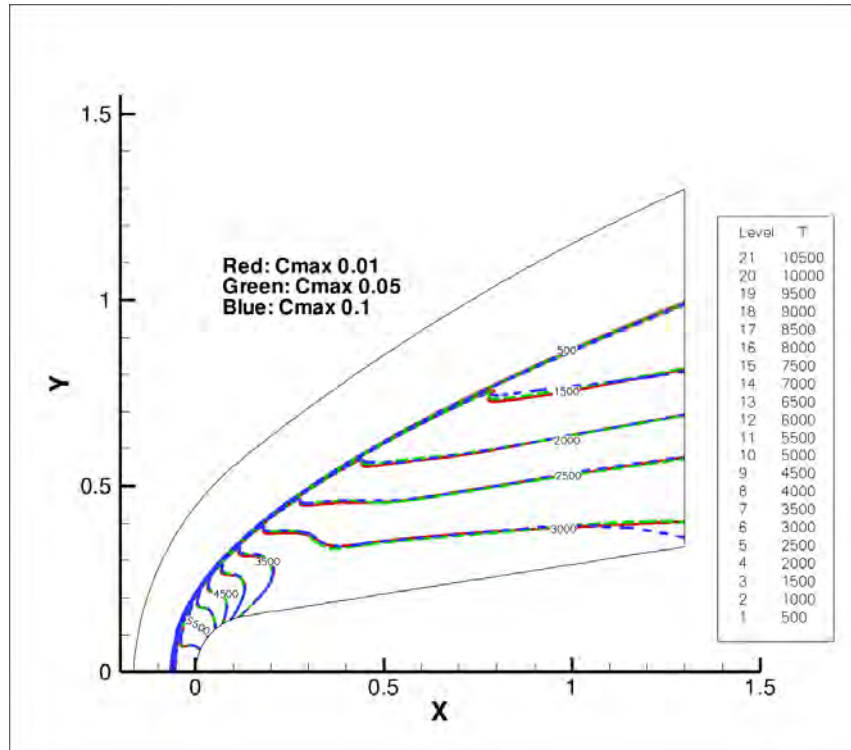
Figure 10. Contour Line Cmax Variation
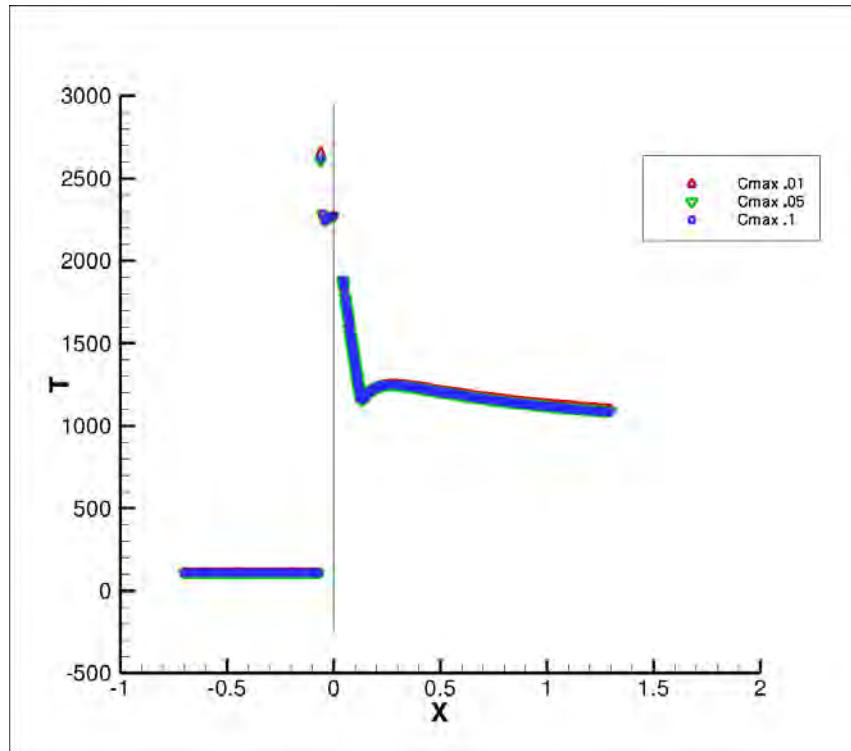
Figure 11. Stagnation Line Cmax Variation
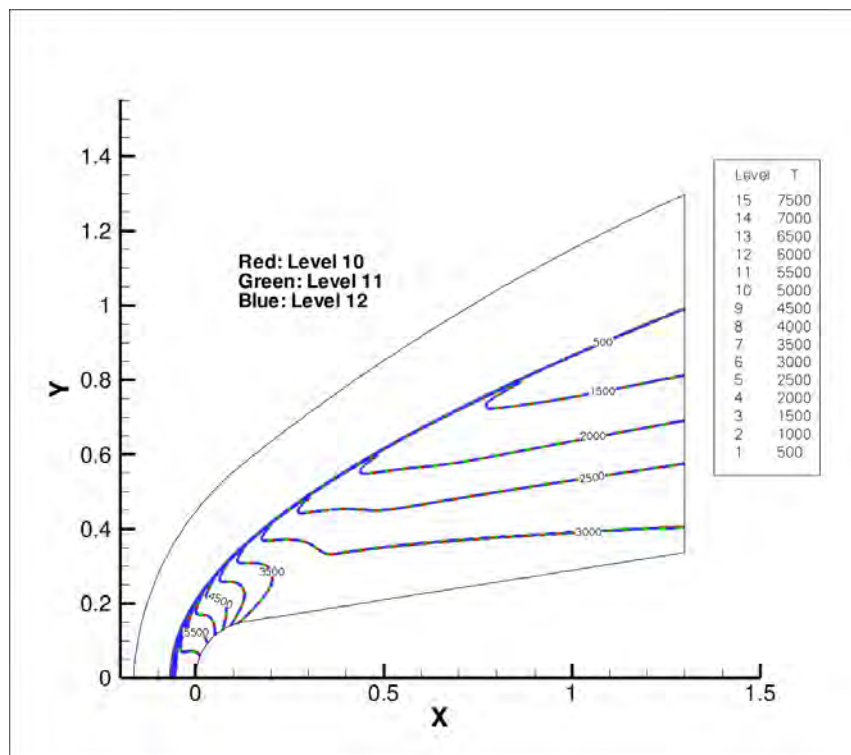
Figure 12. Contour Line Refinement Level Variation

36
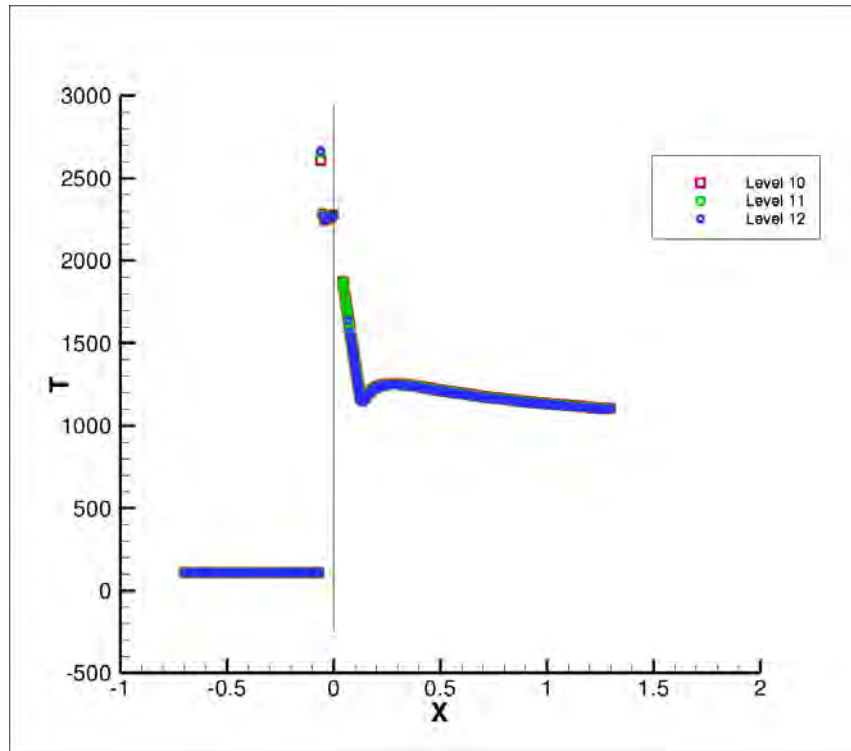
Figure 13. Stagnation Line Refinement Level Variation
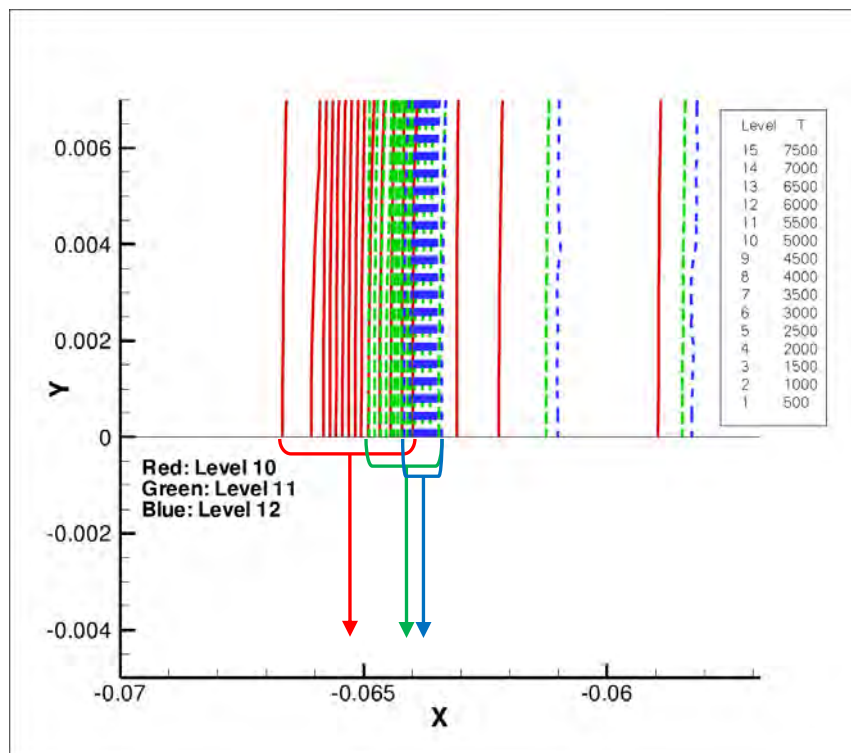
Figure 14. Shock Region Close for Refinement Level Variation

3.3 Convergence

 a. **LeMANS**

   For most CFD solver the residuals show convergence. The residuals are the error in the simulation. To have a converged solution the residuals should be as low as possible and the most used value for a residual is on the order of 10^{-5} or 10^{-6}. The method of determining convergence for LeMANS is the use of residuals. Once the residuals have either reached a specified level or have reach a point where a change is not noticeable the solution can be considered converged. For all of the cases the residuals reached a point where the change was not noticeable before the solution is considered converged.

 b. **UFS**

   The convergence of UFS is not like most CFD solvers. UFS does not output residuals; instead UFS uses user defined points in the solution. UFS then tracks the solution at each point and outputs the solution for each iteration. Once the solution at each point reaches steady state the overall solution is said to be converged. For the given geometry two points are placed in the solution, one at the front of the geometry in the stagnation region and the second on top of the geometry at the back.

3.4 General Simulation Settings

   There are many input parameters to be set in both LeMANS and UFS for each simulation. Some of the inputs did not change, some were left as the defaults, and others changed depending on which simulation was being run. For LeMANS, Table 2 shows the inputs that did not change and the settings for each input. Table 2 also has a brief description of each of the inputs. Two important things to note from Table 2 are that

viscosity is turned off and an adiabatic boundary condition is turned on because UFS uses

the Euler equations and these are the two assumptions for the Euler equations. For UFS,

Table 3 shows the unchanging inputs with settings and descriptions. The important thing

to note from Table 3 is use of a mirror reflection boundary condition. The reason this is

important is because mirror reflection gives an adiabatic boundary condition while a

diffuse reflection does not. For the inputs that are left as defaults refer to each programs

user manual [31][32]. The inputs that vary depending on the simulations are described

later in this chapter in the appropriate sections.

Table 2. LeMANS Input Settings and Descriptions

Parameter	Setting	Description
IS_VISCOUS	0	Value of 1 makes the solution viscous
IS_ADIAB	1	Value of 1 set an adiabatic wall boundary condition
IMPLICIT	2	Value of 0 sets explicit time integration Value of 1 sets point-implicit time integration Value of 2 sets a line-implicit time integration
IS_SECOND_ORDER	1	Value of 1 makes the solution second order accurate

Table 3. UFS Input Setting and Descriptions

Input	Setting	Description
SolverType	0	Value of 0 set Euler solver Value of 1 sets Navier-Stokes solver
SteadyState	1	Value of 0 makes the solution time dependent
SolverOrder	0	Value of 0 makes solution first order accurate Value of 1 makes solution second order accurate
SurfaceBcType	0	Value of 0 uses a mirror reflection boundary condition Value of 1 uses a diffuse reflection boundary condition
RefMass	10	Sets the value of the reference mass
RefTemperature	300	Sets the value of the reference temperature
RefMassDensity	$2.816e^{-4}$	Sets the value of the reference density
RefLength	1	Sets the value of the reference length

3.5 Perfect Gas Cases

For the perfect gas case two different cases are run, one with a monatomic gas and one with a 2 species diatomic gas. The monatomic molecule that is used is argon and the diatomic case used monatomic and diatomic nitrogen. The concentration of each species for the diatomic case was 99.5% N2 and .5% N.

a. LeMANS

In LeMANS there were some specific settings that were required for the case to be perfect gas. The most important settings were to turn off the chemical reactions and to make sure there was no thermal non-equilibrium as well. Along with those two setting the vibrational temperatures for the freestream and wall had to be set to 0 K. Table 4 shows the actual parameters with the setting to reach the above conditions.

Table 4. Perfect Gas Input Parameters and Settings

Parameter	Setting	Description
IS_CHEM_REAC	0	Turns on and off chemical reactions
IS_NON_EQ	0	Turns on and off thermal non-equilibrium
TV_INF	0	Sets freestream vibrational temperature
TV_WALL	0	Sets wall vibrational temperature

b. UFS

The monatomic case is run without any problems using the initial conditions stated above and the results are shown in the next chapter. Since UFS cannot run a single species diatomic case a multiple species case using two species was created. Normally when running multiple species vibrational non-equilibrium occurs but for a perfect gas simulation the vibrational, rotational, and translational energies are treated as the same. To fix this problem, the normalized characteristic temperature for vibration is set high

enough that the vibrational mode will not be excited. For the diatomic simulation the normalized characteristic temperature was set to 50. Along with the normalized characteristic temperature the other is the use of the Millikan and White vibrational relaxation model [33], which is set using VTRelaxModel equal to one in the input file. The combination of these two inputs minimizes the vibrational mode, which is shown in the next chapter. The other data needed for the gasdy_species file is: species name, species mass, species diameter, rotational degrees of freedom, the number of collisions required for the rotational and vibrational modes to reach equilibrium, and the heat of formation. The required data was taken from an example file or could be found online if necessary.

3.6 Thermal Non-equilibrium Cases

For the thermal non-equilibrium there is just a two species case run. The two species used in the simulation are N2 and N with a concentration of 99.5% and .5% respectively.

a. LeMANS

For the simulation run in LeMANS there are four settings that must be changed from the perfect gas cases. First, the thermal non-equilibrium must be turned on to activate the vibrational mode. Since there is thermal non-equilibrium then the vibrational relaxation model must be set to Millikan and White model. The other settings are the vibrational temperatures for the freestream and wall, which both must be set to 300 K.

Table 5. Thermal Non-Equilibrium Parameters and Settings

Parameter	Setting	Description
MOD_MILLIKAN	1	Turns on and off Millikan and White vibrational relaxation
IS_NON_EQ	1	Turns on and off thermal non-equilibrium
TV_INF	0	Sets freestream vibrational temperature
TV_WALL	0	Sets wall vibrational temperature

b. UFS

The only changes from the perfect gas cases is changing the normalized vibrational characteristic temperature and VTRelaxModel. The normalized vibrational characteristic temperature must be set using the correct characteristic temperature. The change to the normalized vibrational characteristic temperature is made in both the input file and the gasdy_species file. Setting VTRelaxModel to zero uses a generic relaxation model instead of the Millikan and White model.

3.7 Thermo-chemical Non-equilibrium Cases

The thermo-chemical non-equilibrium cases have a total of two different simulations that will be run; a two-species, and an eleven-species.

Table 6 shows each case with all of the species and the species concentration used for each simulation, where the plus signifies an ionized molecule. The 11 species concentrations are the values from a test simulation because simulations with other concentrations gave an error with Cantera.

Table 6. Thermo-chemical Non-equilibrium Cases with Species

Simulation	Species	Concentrations (respectively)
2 Species	O2, O	0.995, 0.005
11 Species	N2, NO, O2, O, O+, N2+, N+, O2+, e NO+, N	0.50035, 0.29002, 0.190085, 0.019858, 1.0007E-7, 1.0007E-7, 1.0007E-7, 1.0007E-7, 4.00028E-7, 6.56561E-16, 2.76532E-17

a. **LeMANS**

The only change to make in the input file for the LeMANS thermo-chemical non-equilibrium cases is to change IS_CHEM_REAC from 0 to 1. All of the other settings from the thermal non-equilibrium cases remain that same.

b. **UFS**

Since UFS uses Cantera to solve the chemistry the Cantera module is referenced in the UFS input file along with turning on the chemistry solver in UFS. Along with the changes in the input file, another file is created for Cantera and is called chemistry.cti. Inside the chemistry.cti file is the species being used in the simulation, the initial pressure and temperature, and information on the reactions between the species. The information for the reactions comes from an article written by Eswar Josyula and William Bailey [34]. The information for the species data came from the Cantera data banks.

3.8 Post Processing

The post processing for LeMANS was very straight forward and there was very little extra that need to be done because everything was already dimensional. There were only two things that did need to be done. The first was that a macro had to be created to extract date along the stagnation line, Appendix B. Once the macro has extracted the

data, in order to get the concentration values for the chemistry simulations equations had to be created in the post-processing program. The number of equations depended on how many different species, for example Equation (3.2) and (3.3) are for a two species case, where CN2 is the concentration of the diatomic nitrogen, CN is the concentration of the monatomic nitrogen, rho is the total density, rho_N2 is the density of N2 and rho_N is the density of N.

$$CN2=rho\_N2/rho$$

(3.2)

$$CN=rho\_N/rho$$

(3.3)

The post-processing for UFS was more difficult and required more equations because of the UFS outputs being dimensionless. Equations (3.4-3.15) show the equations necessary in order to dimensionalize the contour data from UFS. The most important thing about each equation is that the outputs must match the outputs that LeMANS has in order to directly compare the two programs. In Equations (3.4) and (3.5) the x and y coordinates had to be adjusted in order to set the stagnation point at (0,0) and match up with LeMANS. The reason for the negative in front of the y is in Equation (3.5) is to flip the solution upside for comparison to LeMANS as in Figure 16. Next in Equation (3.9) the partial pressures, p_c_1 and p_c_2, have to be multiplied by the non-dimensional mass, of the corresponding species, before being added to help dimensionalize the total pressure. This dimensionalization is only required when using multiple species because of how UFS sets up the initial condition for multiple species.

44

$$X = X - .201172 + .000488 \tag{3.4}$$

$$Y = -Y - .5 \tag{3.5}$$

$$rho\_N2 = rho\_1 * RefDensity \tag{3.6}$$

$$rho\_N = rho\_2 * RefDensity \tag{3.7}$$

$$rho = rho\_N2 + rho\_N \tag{3.8}$$

$$P = (p\_c\_1 * mass\_1 + p\_c\_2 * mass\_1) * Tref * RefDensity * R \tag{3.9}$$

$$T = ((3 * T\_c\_1 + 2 * Tr\_c\_1)/5) * Tref \tag{3.10}$$

$$a = sqrt(P/rho * gamma) \tag{3.11}$$

$$mt = 1/nt\_c \tag{3.12}$$

$$V = sqrt((ut\_c/sqrt(1/mt))**2 + vt\_c * vt\_c) * Vref \tag{3.13}$$

$$M = V/a \tag{3.14}$$

$$Tv = Tv\_c\_1 * Tref \tag{3.15}$$

45

Equation (3.10), the equation to dimensionalize the temperature, required a little extra because in LeMANS the translational and rotational temperature are combined into one temperature. For UFS they are kept separate so an average of the two temperatures was required. The final thing to notice is the dimensionalization of the x-component of velocity in Equation (3.13) before being multiplied by the reference velocity has to undergo more dimensionalization. The extra dimensionalization is only done again when running multiple species, but when running single species only the reference velocity is required. The equation for the reference velocity can be seen in the UFS user's manual [32].

There are also equations used for the stagnation line plot but most of the equations are exactly the same as Equation (3.4-3.15) except for format because UFS outputs the stagnation data differently. There are two different equations needed only for the stagnation line and shown in Equations (3.16) and (3.17). Both equations are different because UFS does not output vibrational and rotational temperature but instead outputs vibrational and rotational energies. In the equations the VibEn is the non-dimensional vibrational characteristic temperature, Tv' is the non-dimensional vibrational temperature, T' is the non-dimensional translational temperature, Tr' is the non-dimensional rotational temperature, and the mass is the non-dimensional mass.

$$\{Tv\}=(VibEn)/(\log((VibEn)/(Tv'*mass)+1))*Tref$$

3.16)

$$\{T\}=((3*\{T'\}+2*\{Tr'\}*2.8)/5)* Tref$$

(3.17)

IV. Analysis and Results

4.1 Introduction

The results for each simulation show the comparison between UFS and LeMANS using a flooded contour comparison, a contour line comparison, and a stagnation line comparison. For ease of reference the flooded contour will always have the LeMANS solution on the top while the UFS solution will be underneath. In the contour line plot LeMANS is always black and UFS is red and for the stagnation line plot LeMANS is the lines and UFS is the circles.

4.2 Perfect Gas

As the previous chapter mentions there are two different simulations run using the perfect gas assumption, a monatomic and 2 species diatomic simulation. The flow conditions for each simulation, which are also mentioned in the previous chapter, are a flow of Mach 10, temperature of 300 K, and density of 2.816E-4 kg/m^3. The results from the perfect gas cases have a UFS grid with a body refinement of level 10 and an initial grid refinement of level 5. Figure 15 shows the comparison of the grid spacing between UFS and LeMANS for the two perfect gas simulations. The automatic grid refinement is set to a max level of 10. The grid spacing in the stagnation region is very similar. In order reach that level of refinement UFS had final cell count around 500,000 cells while LeMANS had around 41,000 cells, which would increase the computational expense of UFS.
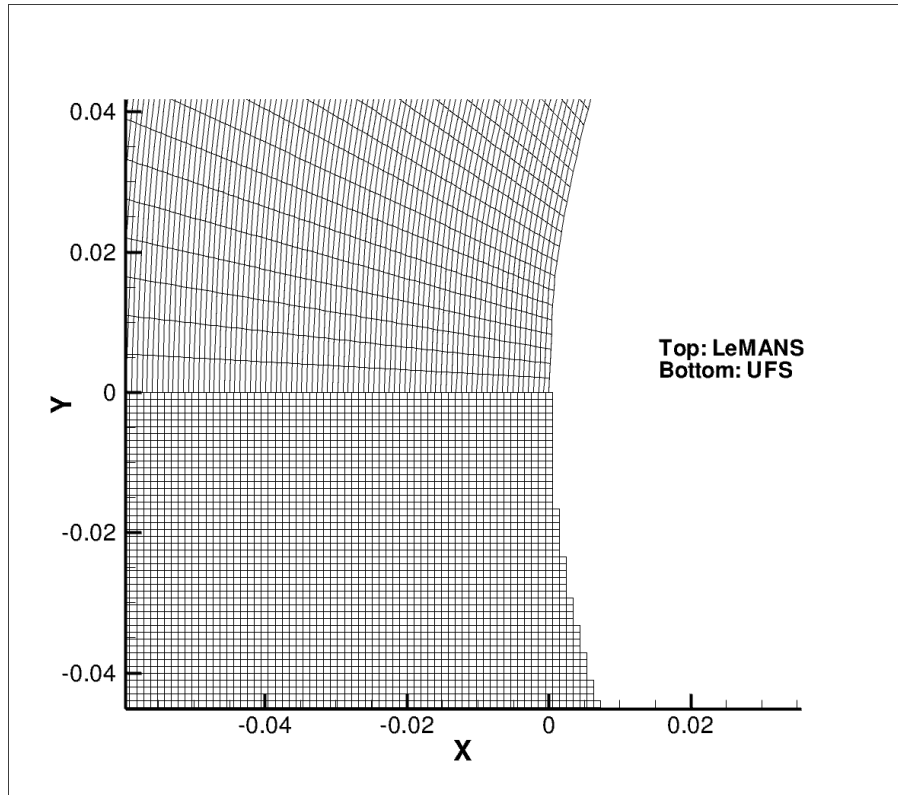
Figure 15. Perfect Gas Stagnation Region Grid Comparison

a. Monatomic Gas

For the monatomic simulation UFS ran for 40,000 iterations at a speed of 11.7 sec/iteration for a total time of 129.91 cpu hours. LeMANS ran for 6000 iterations at a speed of 4.6 sec/iteration for a total time of 7.66 cpu hours. These results show that LeMANS is much less computationally expensive than UFS. LeMANS takes fewer iterations, less time per iteration, and less overall time. The reason for the difference is most likely because LeMANS uses implicit time integration, which allows for a larger time step for cells of the same size, while UFS uses explicit.

The flooded contour, Figure 16, shows in a qualitative way how close LeMANS and UFS agree. The shape of the shocks for each program is similar along with the

48

coloring of the contours. There is a slight difference in the lower level contour color

shapes but how much of a difference is difficult to tell from this view and will be

discussed more with the contour line plot, Figure 18. A closer look at the stagnation

region in Figure 17 shows a slight difference in the shock standoff distance but the

percent difference is only 1.36%, which is low enough to be considered negligible.
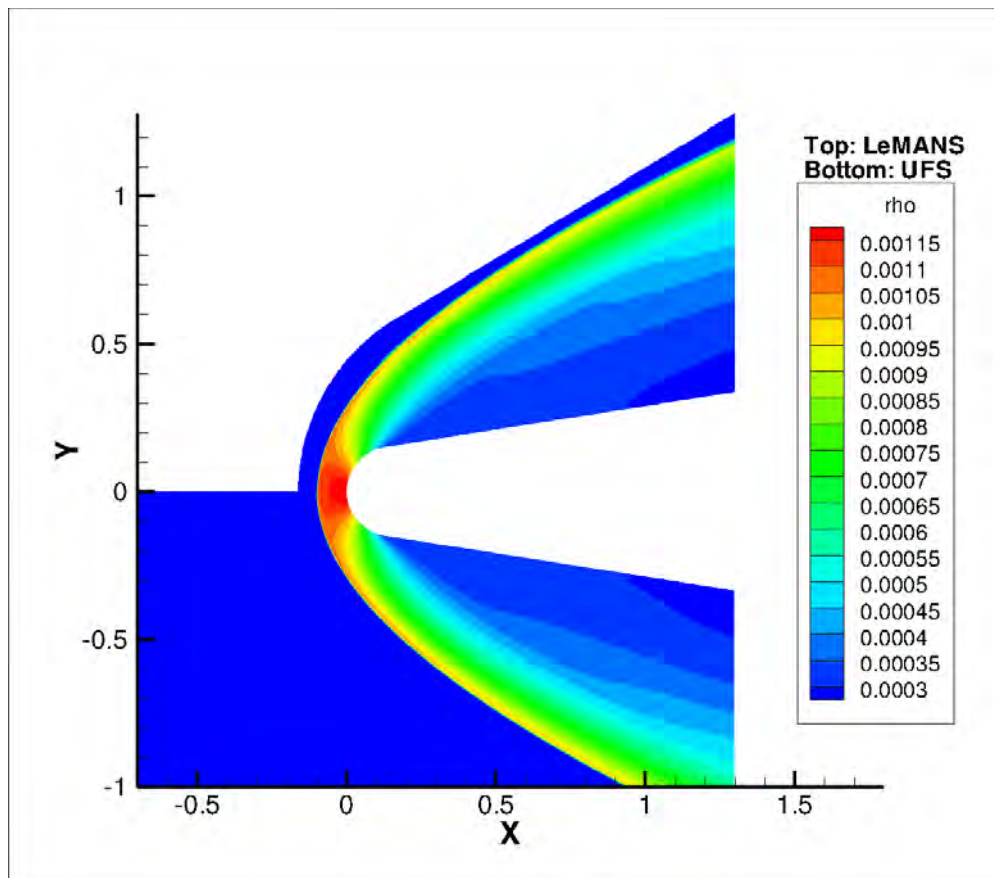
Figure 16. Monatomic Perfect Gas Flooded Contour Comparison
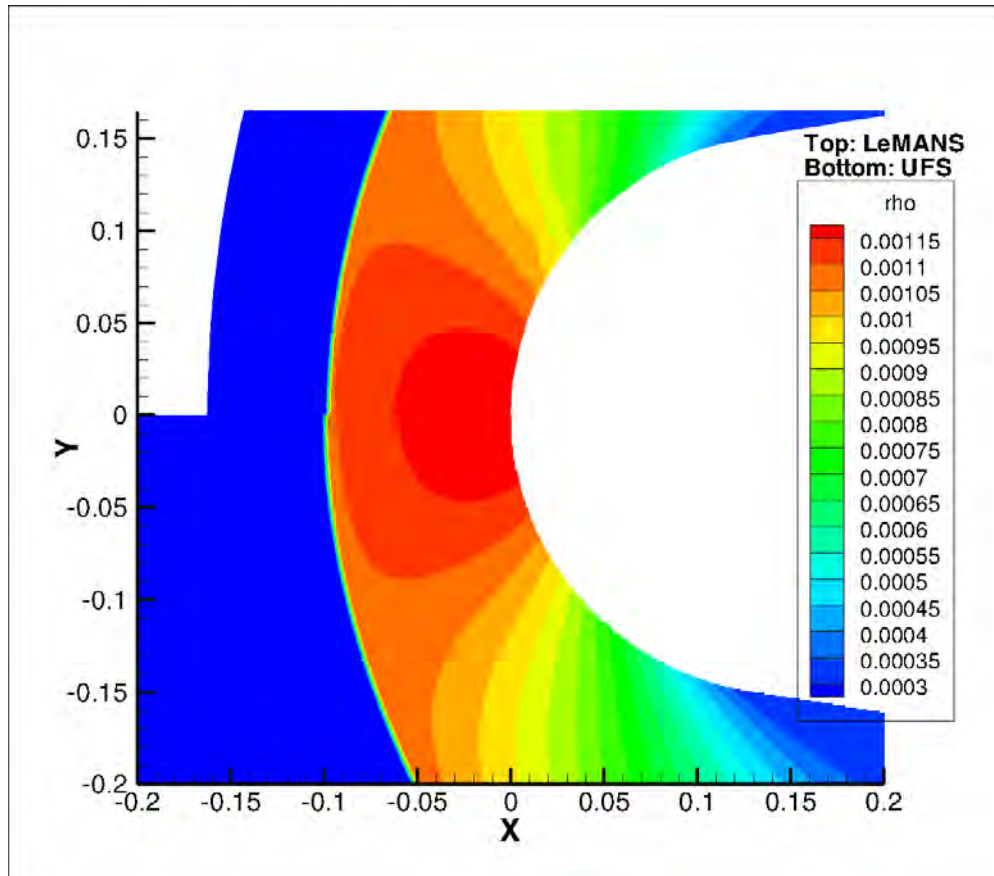
Figure 17. Monatomic Perfect Gas Stagnation Region Flooded Contour Comparison

Figure 18 shows the how the contour lines compare between the two programs. As mentioned above there are some slight difference between a few of the contour line. The max percent difference in height between UFS and LeMANS is 4.25%, whcih is a still within the bounds of acceptable. The difference could be decreased by increasing the grid refinement.
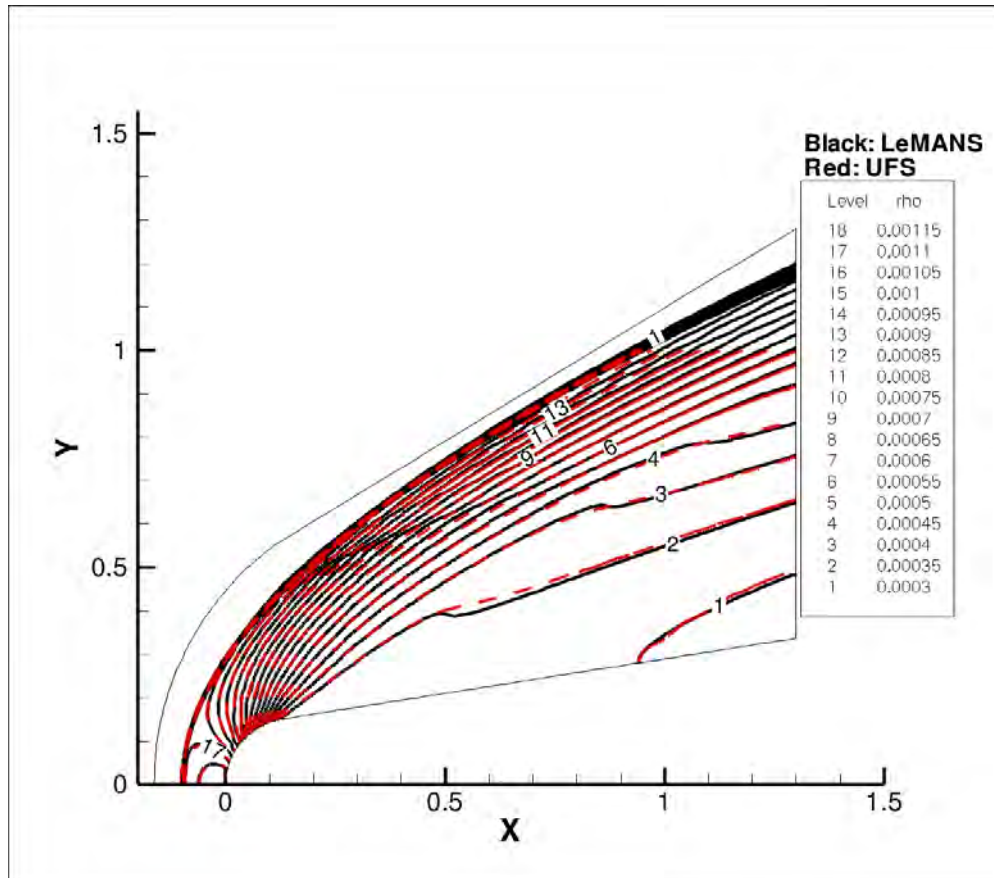
Figure 18. Monatomic Perfect Gas Contour Line Comparison

Figure 19 shows the stagnation line data comparison between UFS and LeMANS. This data shows that both UFS and LeMANS have very similar values for density through the shock and in the stagnation region. This data also makes sense from what is known about perfect gas flow through a shock in that the density increases through the shock and continues to increase up to the stagnation point where the density decrease as the flow expands around the geometry. Table 7 is a comparison of pressure, density and temperature at the stagnation point for both UFS and LeMANS along with the percent difference between the two values for each property. The data shows that UFS is in very close agreement with LeMANS.
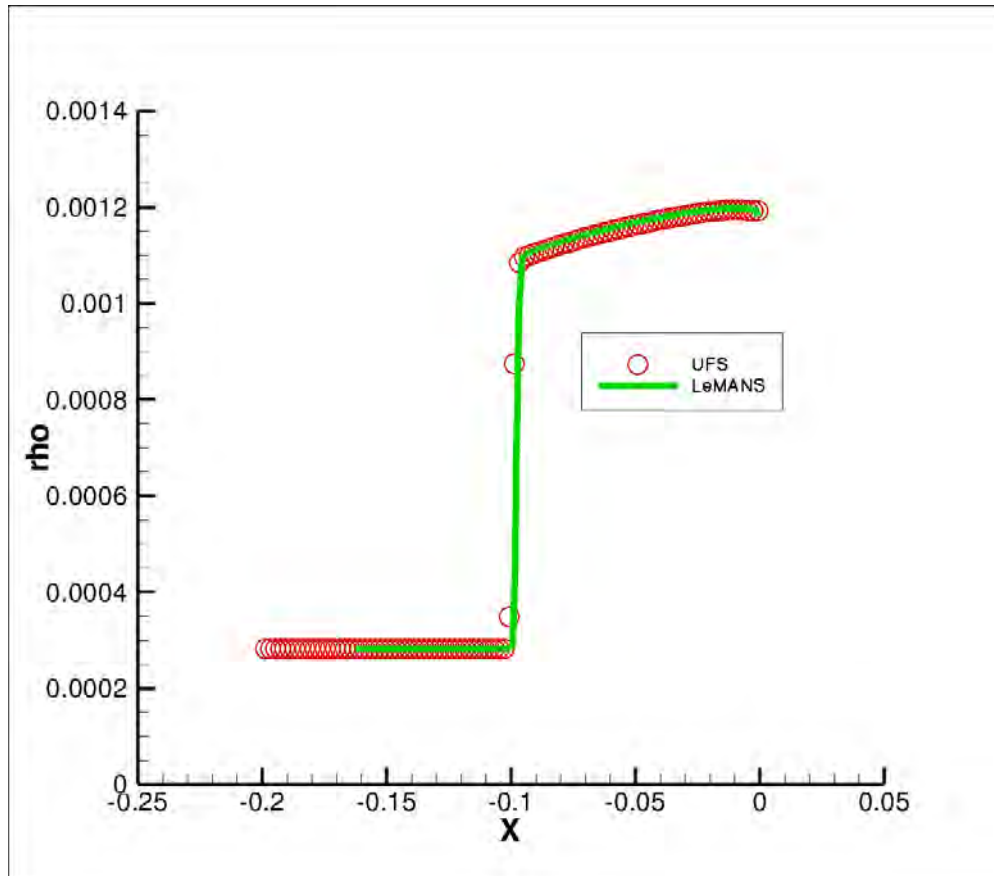
Figure 19. Monatomic Perfect Gas Stagnation Line Comparison

Table 7. Monatomic Perfect Gas Stagnation Point Property Comparison

	Pressure (N/m^2)	Density (kg/m^3)	Temperature (K)
UFS	2,576	1.194E-03	10,368
LeMANS	2,604	1.198E-03	10,474
Percent Difference	1.07%	0.36%	1.01%

b. Diatomic Gas

For the diatomic simulation UFS ran for 50,000 iterations at a speed of 30 sec/iteration for a total time of 416.8 cpu hours. LeMANS ran for 6000 iterations at a speed of 6.8 sec/iteration for a total time of 11.4 cpu hours. These results show that LeMANS is much faster than UFS.

In the results for the diatomic perfect gas simulation the pressure, density, and temperature were compared and the temperature results are shown. The temperature is shown because the vibrational energy mode had to be initialized since UFS cannot run a single diatomic species simulation and when multiple species is used the vibrational mode must be initialized.
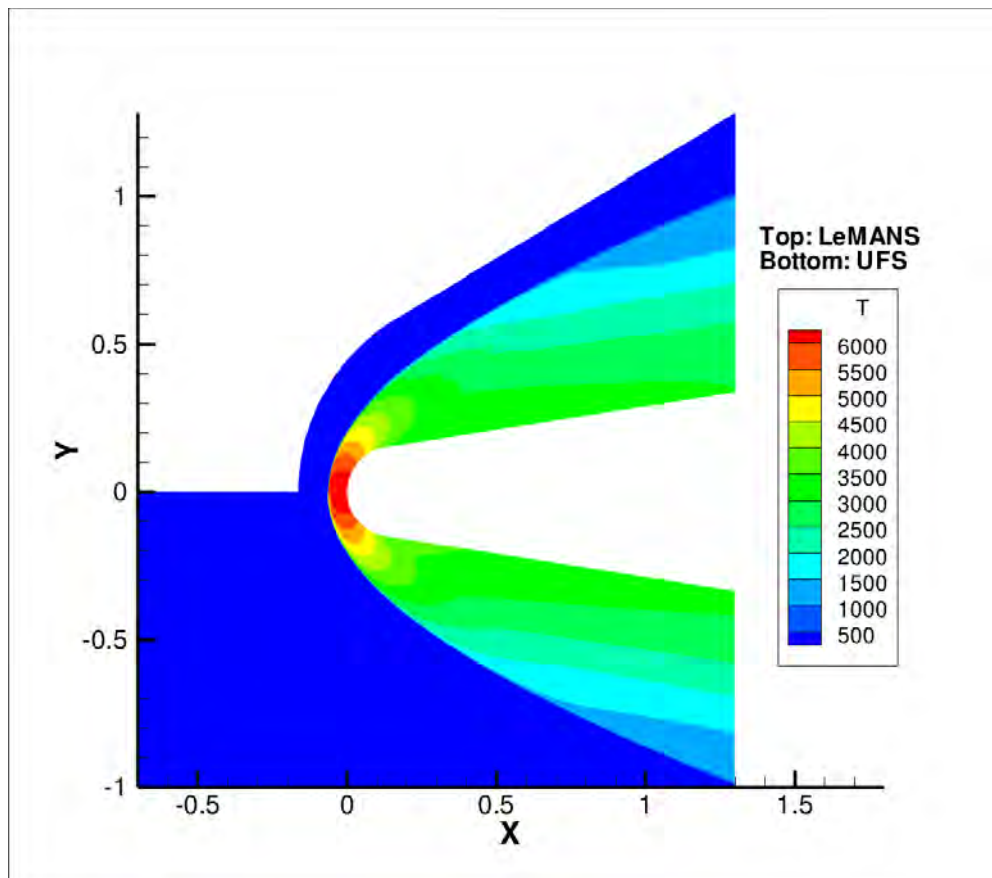
Figure 20. Diatomic Perfect Gas Flooded Contour Comparison

The results of the flooded contour, Figure 20, shows the general shape of the shock for both solutions is the same, as is the level of the temperature around the stagnation region. The only difference is shock in front of the stagnation region and Figure 21 shows a better view of the stagnation area. This closer view shows that the

shock standoff distance is larger in UFS than in LeMANS. The percent difference between the two distances is 6.4%. The shock thickness in UFS also appears to be larger then LeMANS. The reason for the differences in UFS is most likely due to the grid being too coarse but a max refinement of 10 was used in order to reduce run time and computational expense of a higher refined grid.
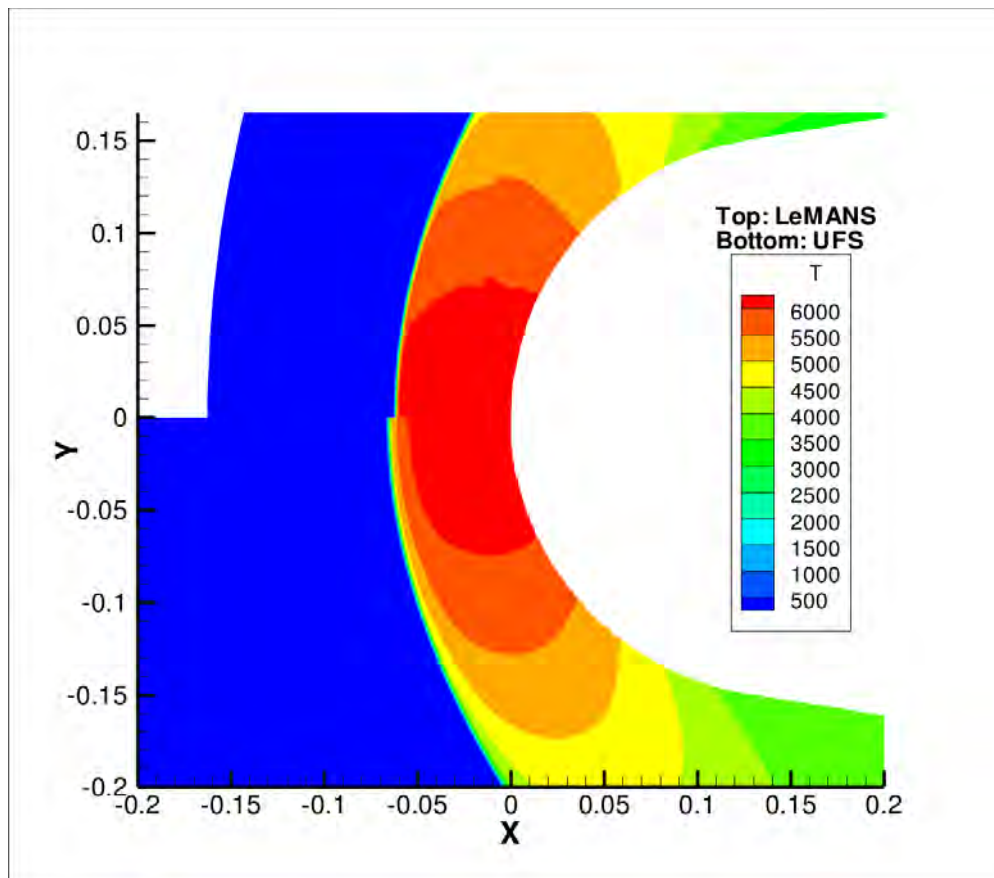
Figure 21. Diatomic Perfect Gas Stagnation Region Flooded Contour Comparison
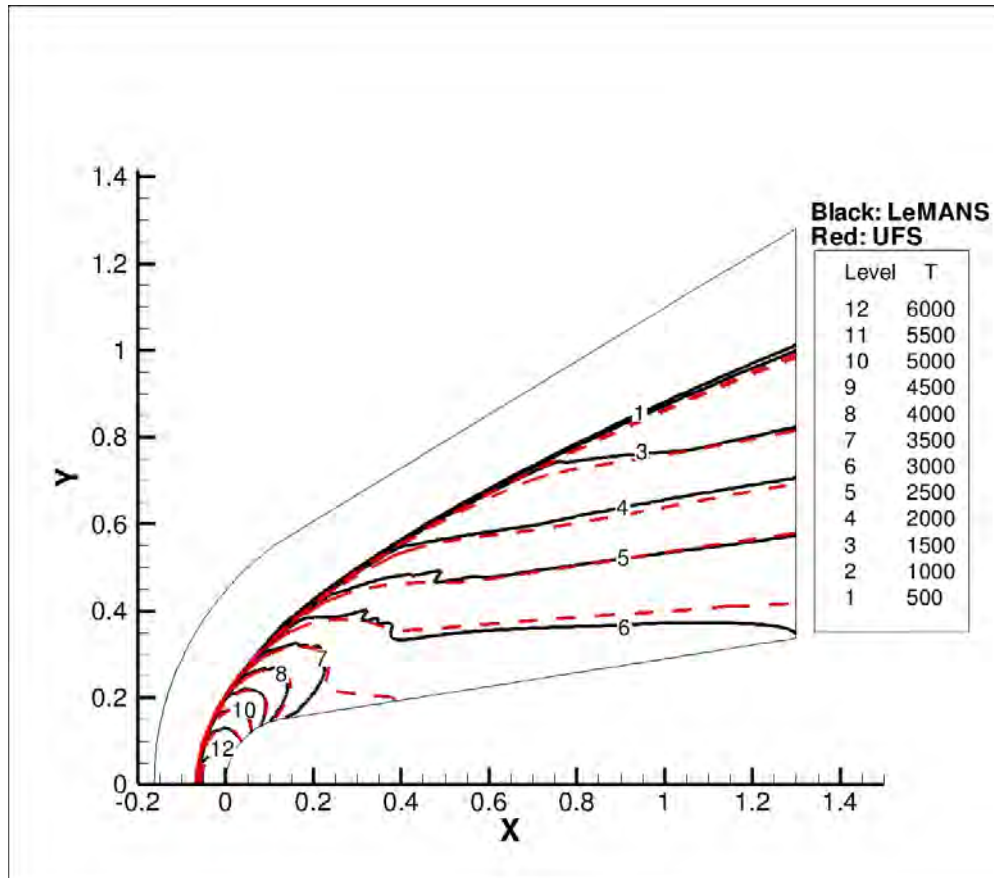
Figure 22. Diatomic Perfect Gas Contour Line Comparison

The contour line comparison, Figure 22, shows again that the general shape of the shock is similar between UFS and LeMANS but the not all of the individual lines in the UFS solution match up to the LeMANS solution. Some of the lines match well with LeMANS, as in the 2000K line, but others are very different, as in the 3500K line. Again this is most likely due to the coarseness of the grid.

The stagnation line plot for the translational/rotational temperature, Figure 23, shows the UFS and LeMANS solutions are relatively close. The only difference is right at the stagnation point where UFS is a little lower than LeMANS. Figure 24 shows the vibrational temperature which as LeMANS shows should be zero but in UFS the vibrational mode has been activated and this is due to the need to use multiple species

and having to initialize the vibrational energy. The fact that the vibrational mode is active affects the temperature throughout the solution because the vibrational energy will take energy away from the rest of the flow. This effect will be almost negligible since the vibrational temperature is about two orders of magnitude different than the translational/rotational temperature.
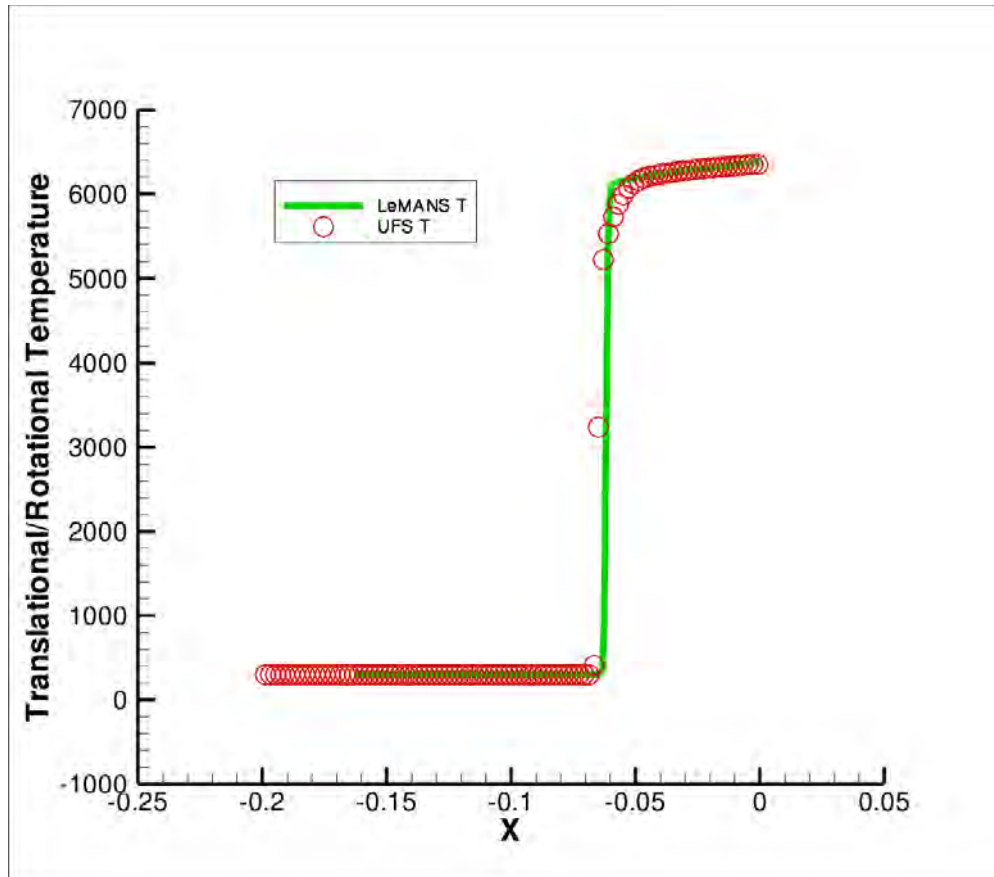
Figure 23. Diatomic Perfect Gas Stagnation Line Comparison (Translational/Rotational Temperature)

Table 8 shows that the stagnation point property values are very close. The results show that the vibrational temperature did not have much of an impact on the overall flow but the temperature may have been closer if the vibrational mode had not been required

to be initialized. Another reason for the differences is again the grid is not independent of the solution.
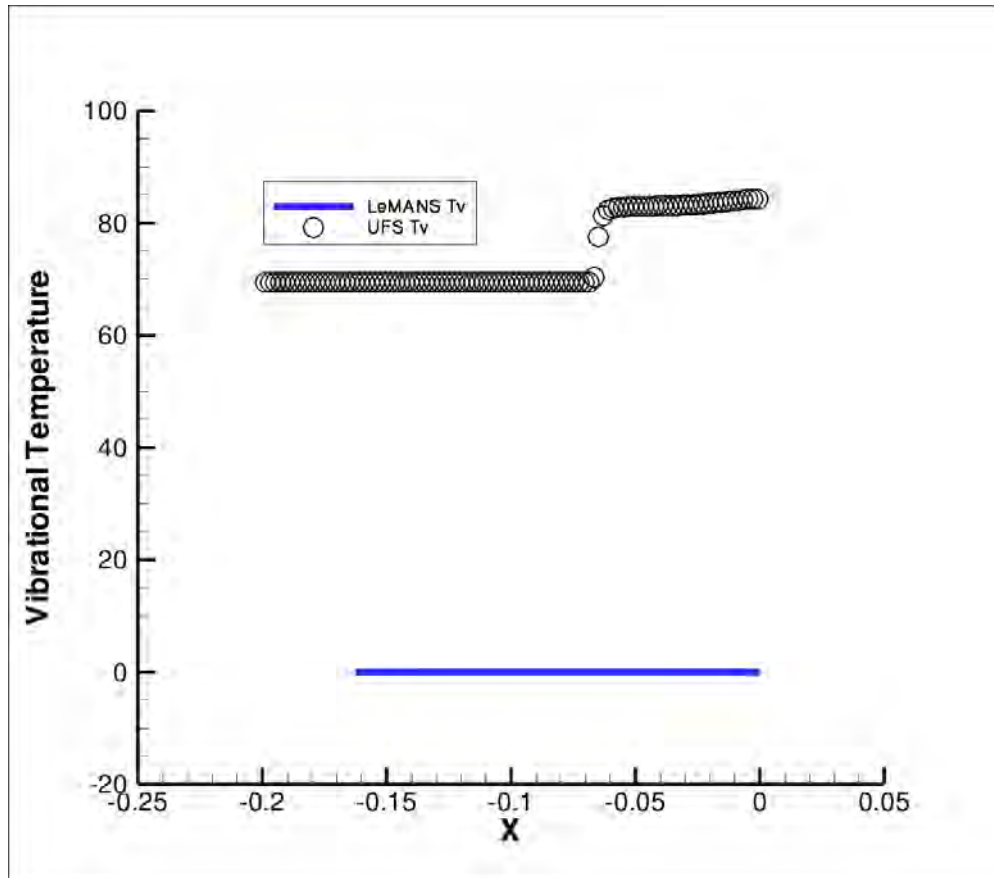
Figure 24. Diatomic Perfect Gas Stagnation Line Comparison (Vibrational Temperature)

Table 8. Diatomic Perfect Gas Stagnation Point Property Comparison

	Pressure (N/m^2)	Density (kg/m^3)	Temperature (K)
UFS	3,199	1.70E-03	6,354
LeMANS	3,257	1.71E-03	6,392
Percent Difference	1.77%	0.72%	0.60%

4.3 Thermal Non-Equilibrium

The flow conditions for this simulation are a flow speed of Mach 10, reference temperature of 300K, and reference density of 2.816E-4. As with the perfect gas cases

pressure, translational/rotational temperature, and density were compared along with the vibrational temperature but only the vibrational and translational/rotational temperatures are shown. The reason for running a thermal non-equilibrium simulation is to look specifically at the vibrational energy mode. The results from the 2 species simulation have a UFS grid with a body refinement of level 9 and an initial grid refinement of level 5. The automatic grid refinement is set to a max level of 9 in the stagnation region and a max level of refinement of 8 everywhere else. Figure 25 shows an example of the different level of refinements, where the darkest area is what is being called the stagnation region. Figure 26 shows a comparison of the grid for both the thermal and thermo-chemical non-equilibrium simulations and shows that unlike the perfect gas simulations the grids do not compare well. UFS had final cell count around 41,000 cells while LeMANS had around 41,000 cells.
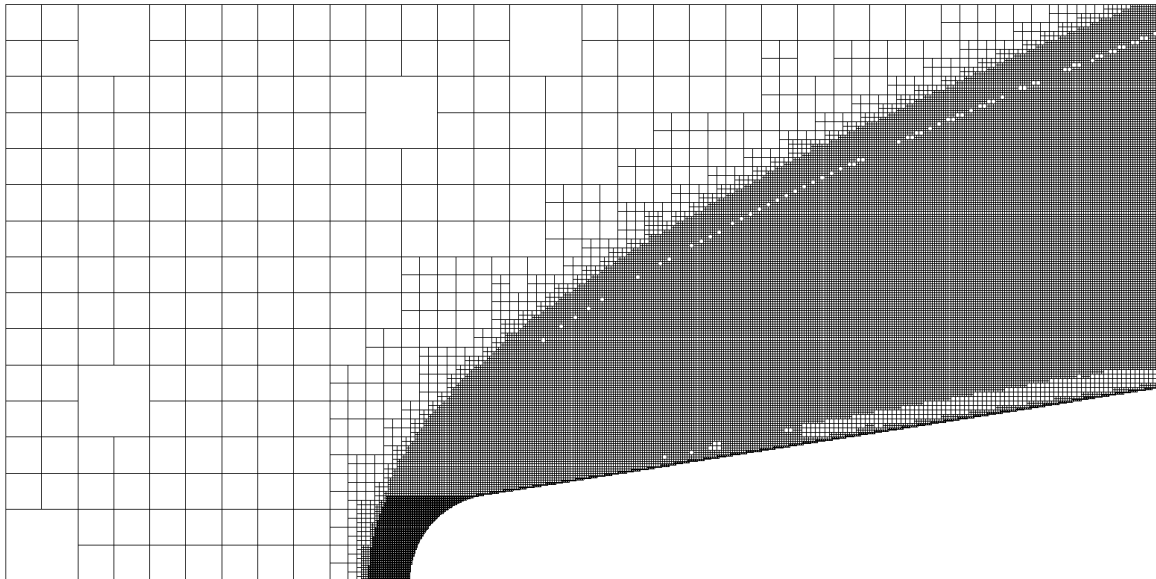
Figure 25. Example of Different Level of Refinement in UFS
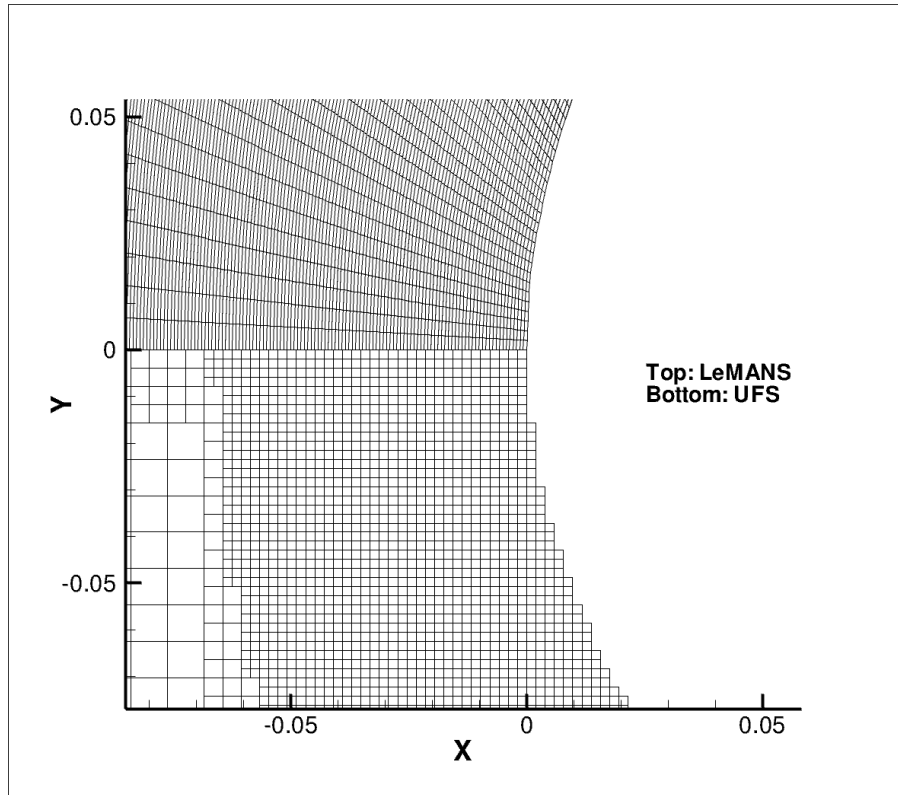
Figure 26. Thermal and Thermo-chemical Non-Equilibrium Stagnation Grid Comparison

a. 2 Species

For the 2 species thermal non-equilibrium simulation UFS ran for 50,000 iterations at a speed of 6.3 sec/iteration for a total time of 87.99 cpu hours. LeMANS ran for 6000 iterations at a speed of 12.4 sec/iteration for a total time of 20.63 cpu hours. These results show that UFS is faster per iteration for this simulation but the number of iterations UFS needs to reach steady state still makes it much slower overall than LeMANS.
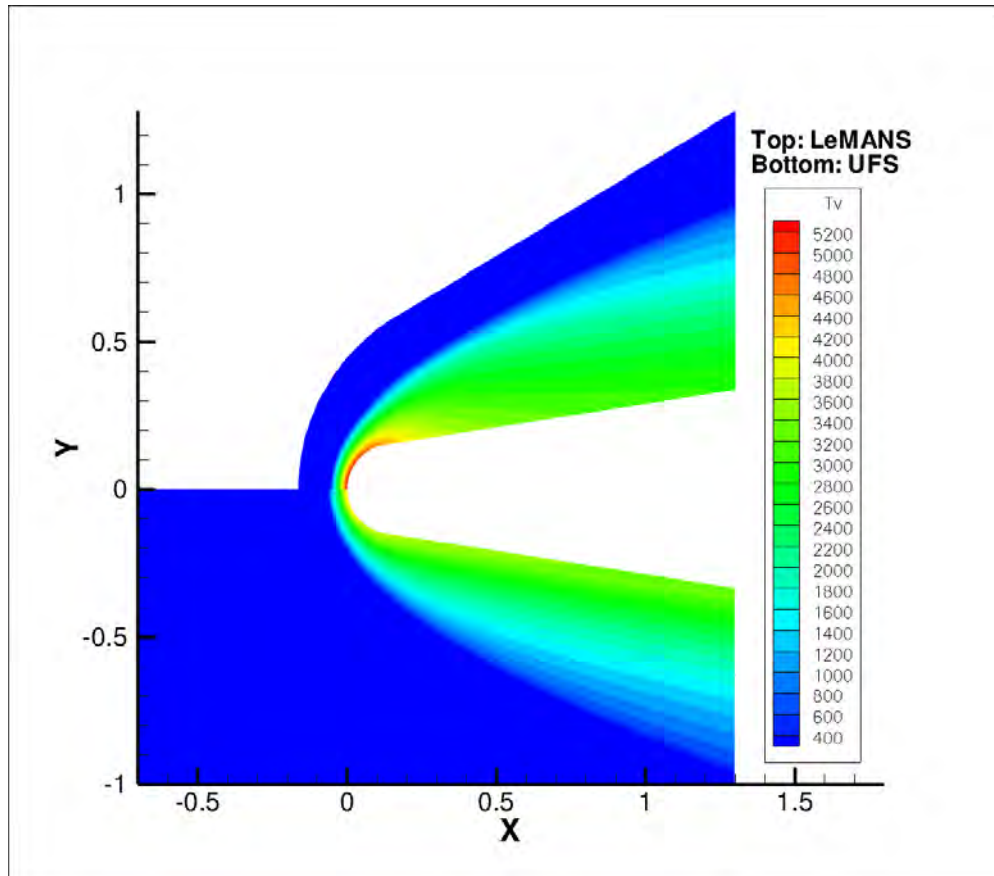
Figure 27. 2-Species Thermal Non-Equilibrium Flooded Contour Comparison

Figure 27 shows the flooded contour for the vibrational temperature and shows that the temperature in the stagnation region is lower in UFS than in LeMANS. The figure also shows there is a difference in the shock standoff distance and that the shock is not very well defined. One reason for the differences is due the grid being too coarse because of the run time and the computational cost of a more refined grid. Another reason for the difference is that that the flux scheme in UFS is too diffusive. When there is too much diffusion in a flux scheme the solution requires a smaller cell spacing to reach the same quality of solution. Since Figure 27 shows a thicker shock in UFS for the larger cells means that the flux scheme is most likely too diffusive in UFS.

60
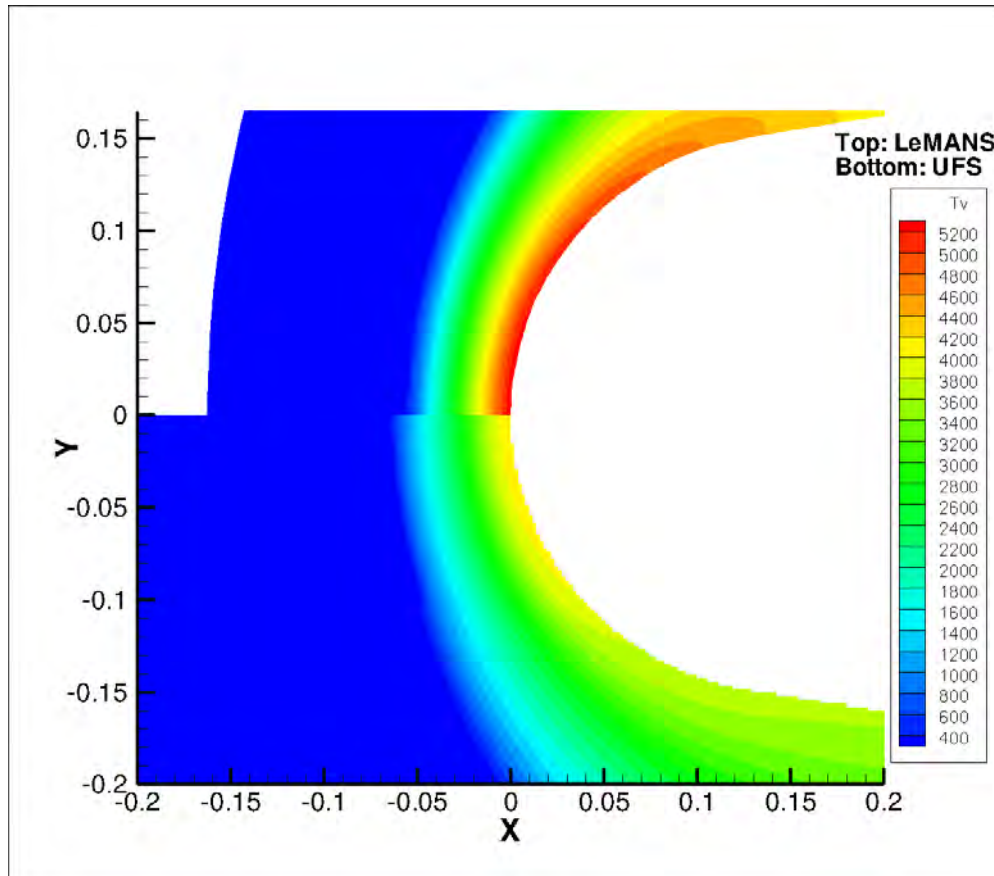
Figure 28. 2-Species Thermal Non-Equilibrium Stagnation Region Flooded Contour Comparison (Vibrational Temperature)

Figure 28 shows a closer view of the stagnation region of the vibrational temperature and shows a better view of how low the temperature is in UFS. The lower vibrational temperature means the there will be more energy in the translational and rotational modes, which will result in higher temperatures in those modes and this is confirmed in Figure 29. Figure 29 also shows better the difference in the shock standoff distance, which results in a percent difference of 10.4%.
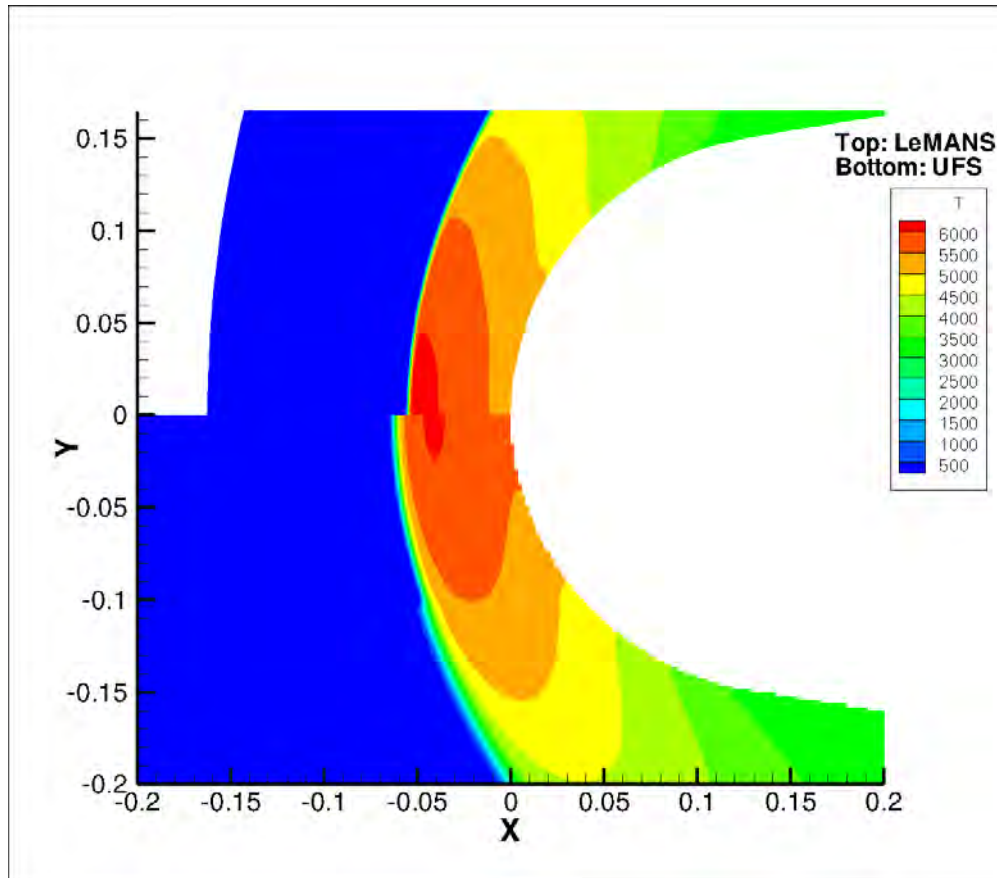
**Figure 29. 2-Species Thermal Non-Equilibrium Stagnation Region Flooded Contour
Comparison (Translation/Rotation Temperature)**

The contour line plot, Figure 30, shows that the individual contour lines between

UFS and LeMANS are not close to matching. One reason for the difference is due to the

coarseness of the grid in UFS because the area outside of the stagnation region is at an

even lower level of refinement than the stagnation region. Figure 31 is the stagnation line

data and shows both the vibration and translational/rotational temperature. The

significant points with this plot are first that the plot shows that the shock in UFS is

thicker because both temperatures start to change earlier in UFS than LeMANS and stop

changing the same time as the LeMANS solution. Second is the fact that the lower

vibrational temperature and the stagnation point leads to a higher translational/rotation

temperature. Again most likely the reason for these differences between UFS and LeMANS is due to the need to use a coarse grid in UFS due to computational expense of a more refined gird.

Table 9 shows the property comparison at the stagnation point. The most important note is the 21.8% percent difference in vibrational temperature, which is most likely due to the coarseness of the grid in UFS. All of the other percent difference values are higher than for either of the perfect gas simulations but are still within a reasonable range.
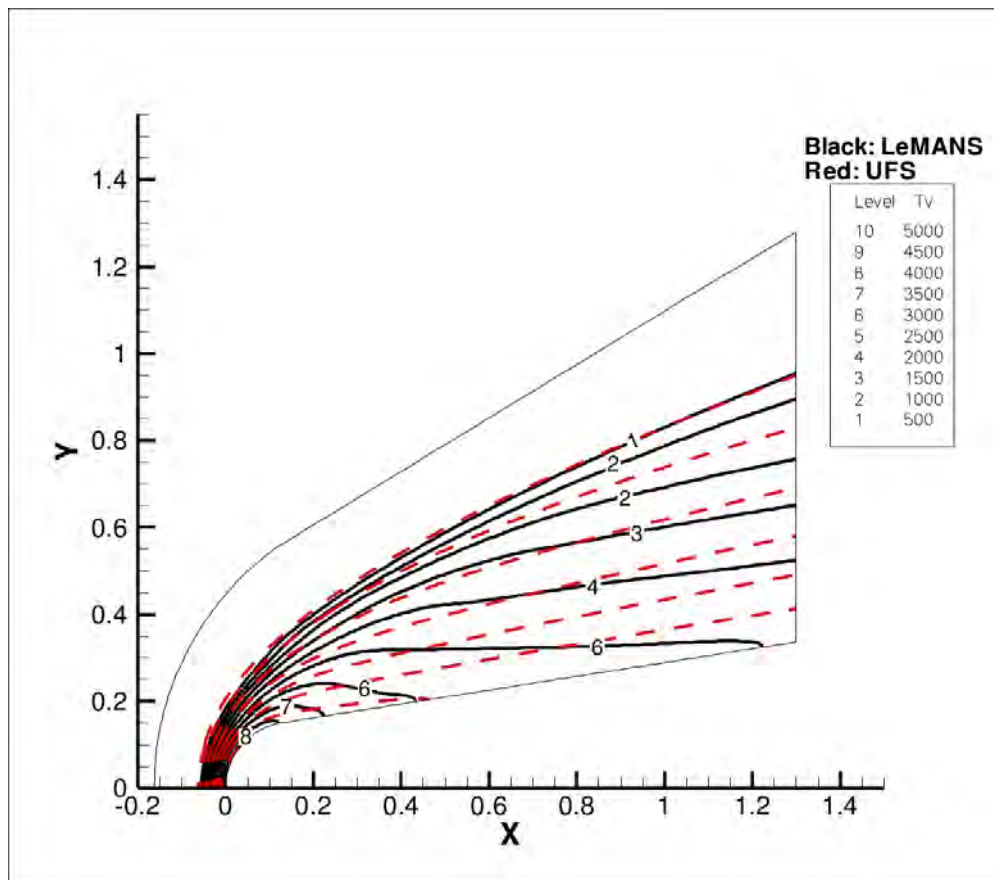
Figure 30. 2-Species Thermal Non-Equilibrium Contour Line Comparison
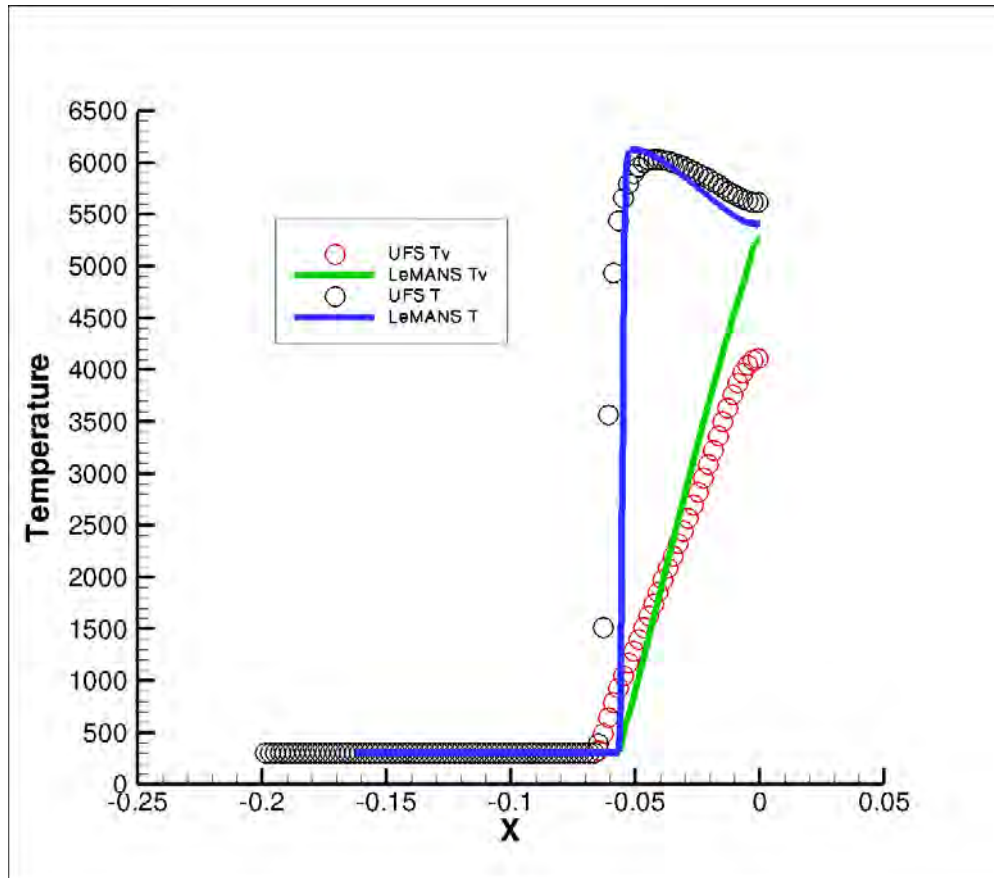
Figure 31. 2-Species Thermal Non-Equilibrium Stagnation Line Comparison

Table 9. 2-Species Thermal Non-Equilibrium Stagnation Point Property Comparison

	Pressure (N/m^2)	Density (kg/m^3)	Translational/Rotational Temperature (K)	Vibrational Temperature (K)
UFS	3,204	1.92E-03	5,618	4,114
LeMANS	3,271	2.03E-03	5,406	5,261
Percent Difference	2.04%	5.24%	3.93%	21.80%

4.4 Thermo-chemical Non-Equilibrium

The initial conditions for this simulation are a flow speed of Mach 10, a temperature of 300K, and a density of 2.816E-4 kg/m^3. As with the thermal non-equilibrium simulation pressure, density, translational/rotational temperature, and vibrational temperature were compare but the most important feature of the thermo-

64

chemical non-equilibrium is the concentration of the species. The results from the 2 species simulation have a UFS grid with a body refinement of level 9 and an initial grid refinement of level 5. The automatic grid refinement is set to a max level of 9 in the stagnation region with a max level of refinement of 8 everywhere else. Refer back to Figure 25 for an example of the difference levels of refinement. Again Figure 26 shows a comparison of the grid for the thermo-chemical non-equilibrium simulations as was mentioned previously. The figure shows that unlike the perfect gas simulations the grid does not compare well. UFS had final cell count around 44,000 cells while LeMANS had around 41,000 cells.

a. 2 Species

For this simulation UFS ran for 50,000 iterations at a speed of 8.1 sec/iteration for a total time of 112.15 cpu hours. LeMANS ran for 6000 iterations at a speed of 23.3 sec/iteration for a total time of 38.78 cpu hours. These results show that UFS is faster per iteration for this simulation but the number of iterations UFS needs to reach convergence still makes it much slower overall than LeMANS.

The most important piece of information about the chemistry simulation is the concentrations of the species because the concentrations show if the chemistry worked correctly. Figure 32 shows the comparison of the concentrations between LeMANS and UFS. As is seen, the concentrations from UFS do not match at all but Figure 32 also makes it look as if there is not any dissociation of the O2, in UFS. Figure 33 shows that in fact the there is dissociation of O2 the amount is just too small to show up on the plot with LeMANS.
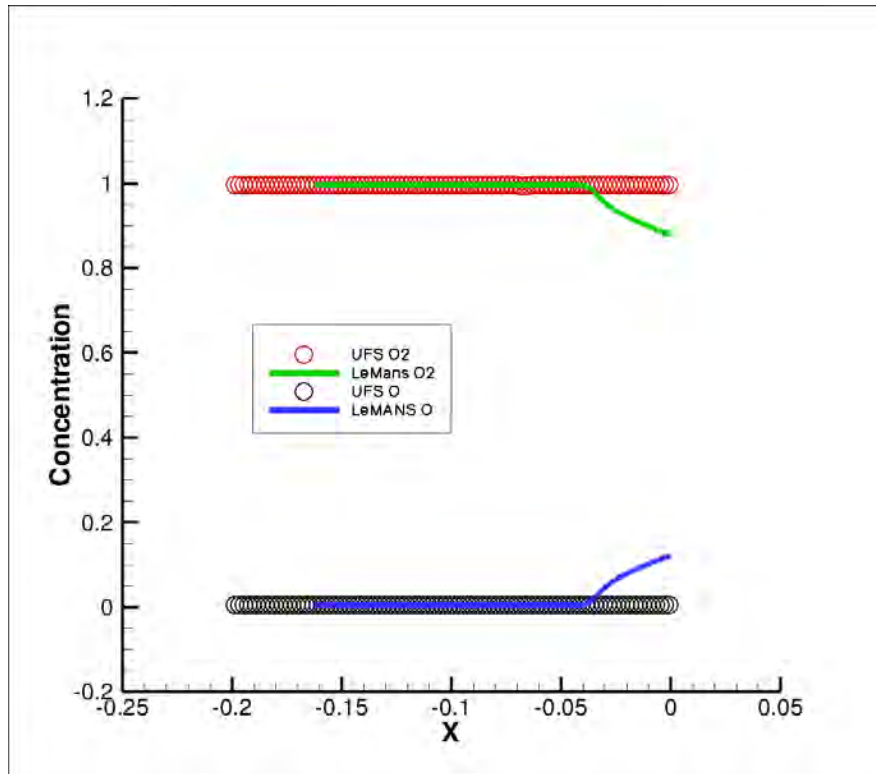
Figure 32. 2-Species Chemistry Concentration Comparison
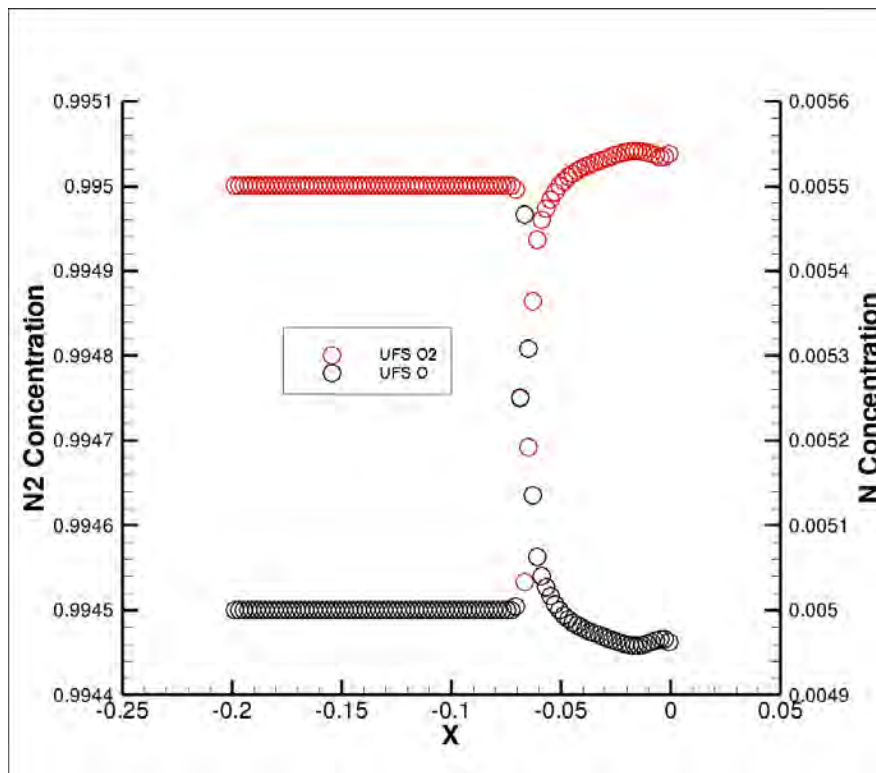
Figure 33. 2-Species Chemistry Concentration (UFS Only)

66

Figure 34 shows the main reason why there is not more dissociation occurring in UFS. Figure 34 shows the vibrational temperature comparison, which shows that the vibrational temperature in UFS quite a bit smaller than LeMANS. Figure 35 shows a closer look at the stagnation region and shows that UFS is on average about four times smaller than LeMANS. The vibrational temperature has an impact on the dissociation of the oxygen because dissociation occurs when the vibrational forces break the bond between the two molecules of a diatomic molecule. Therefore the main reason the dissociation is so small in UFS is because the vibrational energy did not get high enough to break apart the O2. The reason that the vibrational energy is not high enough is due in part to the coarseness of the grid in UFS.
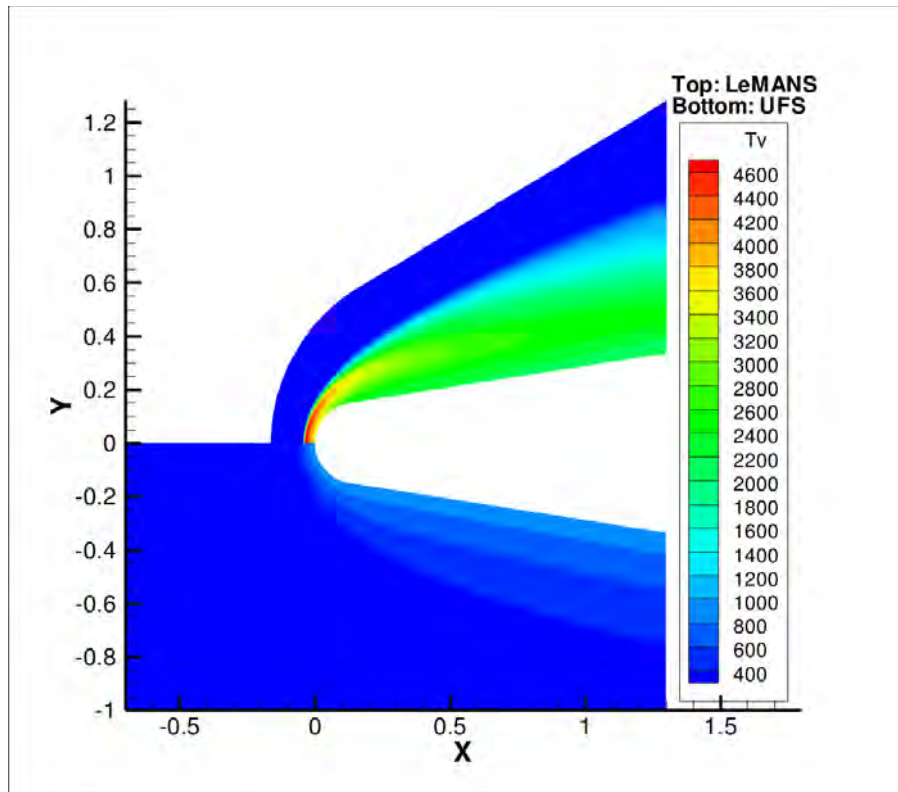
Figure 34. 2-Species Chemistry Flood Contour Comparison
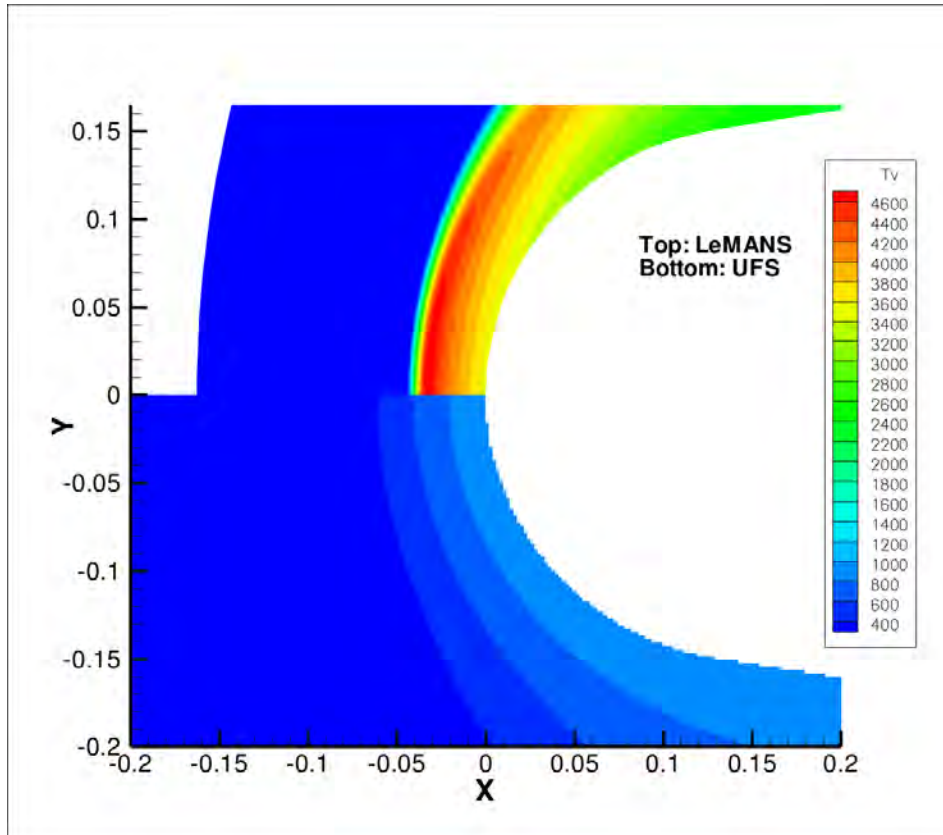
**Figure 35. 2-Species Chemistry Stagnation Region Flood Contour Comparison
(Vibrational Temperature)**

Figure 36 shows the effect the low vibrational temperature also has on the translational/rotational temperature. Since the vibrational temperature in UFS is so small the translational/rotational temperature in UFS should be a lot higher closer to the stagnation point than LeMANS, which is exactly what is seen in Figure 36. The reason for this is because there is more energy in the translational/rotational mode since not as much energy is getting used by the vibrational dissociation. Another thing to notice from Figure 36 is that the shock standoff distance in UFS is much larger than in LeMANS.
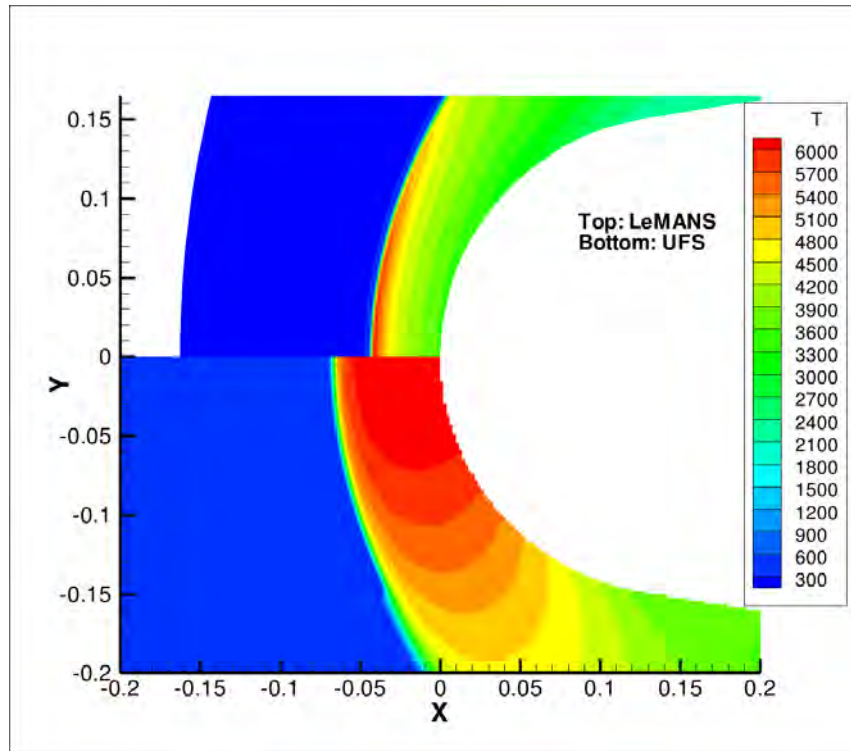
Figure 36. 2-Species Chemistry Stagnation Region Flood Contour Comparison (Translational/Rotational Temperature)

The plot of the stagnation line, Figure 37, shows the comparison of both temperatures for both programs. The first thing to note is the shock standoff distance in UFS. The standoff distance in UFS is about 0.025 meters larger than LeMANS. The next thing to note is the large difference in the temperatures. UFS has an overall higher translational/rotational temperature and a much lower vibrational temperature. The reason for the higher translational/rotational temperature is that the vibrational temperature is so low and not taking the correct amount of energy away from the translational and rotational energy modes. The reason for the low vibrational temperature has to due partially with the coarseness of the grid and since a smaller cell size is required then the flux scheme may also be too diffusive. Too much diffusion would cause the thicker shock

and also the higher shock standoff distance. There could also be parameters missing from the input file, that were not in either of the example simulations or the user's manual, that are necessary to run thermo-chemical non-equilibrium. The user's manual does not specify what input parameters required and the two example simulations have different input parameters from each other even though they both activate the chemistry module.
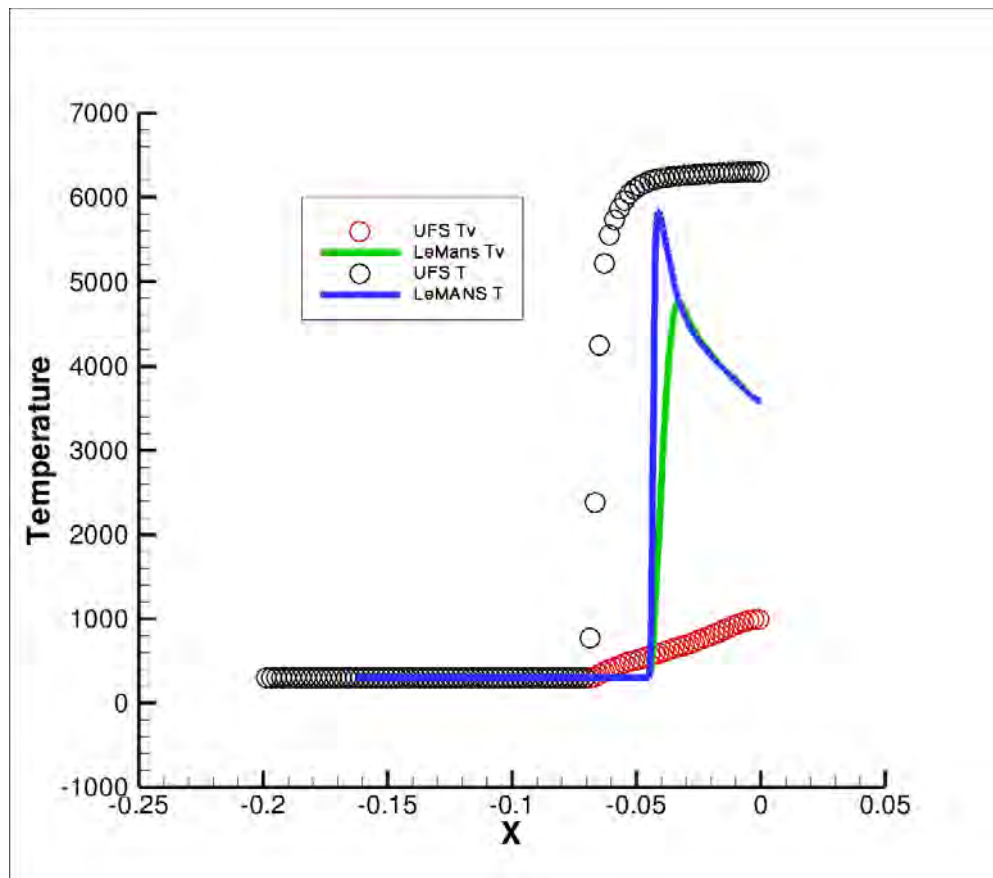
Figure 37. 2-Species Chemistry Stagnation Line Comparison

b. **11 Species**

For this simulation UFS ran for 50,000 iterations at a speed of 28.6 sec/iteration for a total time of 397.9 cpu hours. LeMANS ran for 6000 iterations at a speed of 93.9 sec/iteration for a total time of 156.5 cpu hours. These results shows that UFS is faster

per iteration for this simulation but the number of iterations UFS needs to reach

convergence still makes it much slower overall than LeMANS.

Figure 38. 11 Species Thermo-chemical Non-Equilibrium Concentrations

The results from the 11 species thermo-chemical non-equilibrium have the same

type of results as in the 2 species thermo-chemical simulation. The concentrations, Figure

38, do not compare at all between UFS and LeMANS. Only the non-ion and electron

species are shown because the ions and electrons had only negligible change and were

removed to simplify the plot. The changes in the concentration in UFS are slightly more

evident by the bump, indicated by the line, right at the shock in UFS. Figure 38 also

shows that the shocks do not line up between UFS and LeMANS. There are a few

reasons for the lack of change in concentrations; the first is the coarseness of the grid. The other reason is the low vibrational temperature seen in UFS, Figure 39. Another reason is that Cantera has not been implemented correctly or that UFS and Cantera are not communicating correctly. One of the problems that was encountered while trying to run the 11 species simulation was that the only concentrations that could be used were the concentrations from the example files. If other concentrations were used there was an error in Cantera therefore making it seem that there might not be something working correctly with Cantera.

Figure 39 shows how much lower the vibrational temperature in UFS is compared to LeMANS. If the vibrational temperature in UFS was higher, the change in the concentrations would probably match closer to LeMANS because species will not dissociate until the vibrational energy is high enough to break the bond in a diatomic species. Figure 40 shows a closer view of the vibrational temperature in the stagnation region and that UFS is about 3 times smaller than LeMANS. The plot also shows that the shock standoff distance is larger than LeMANS.
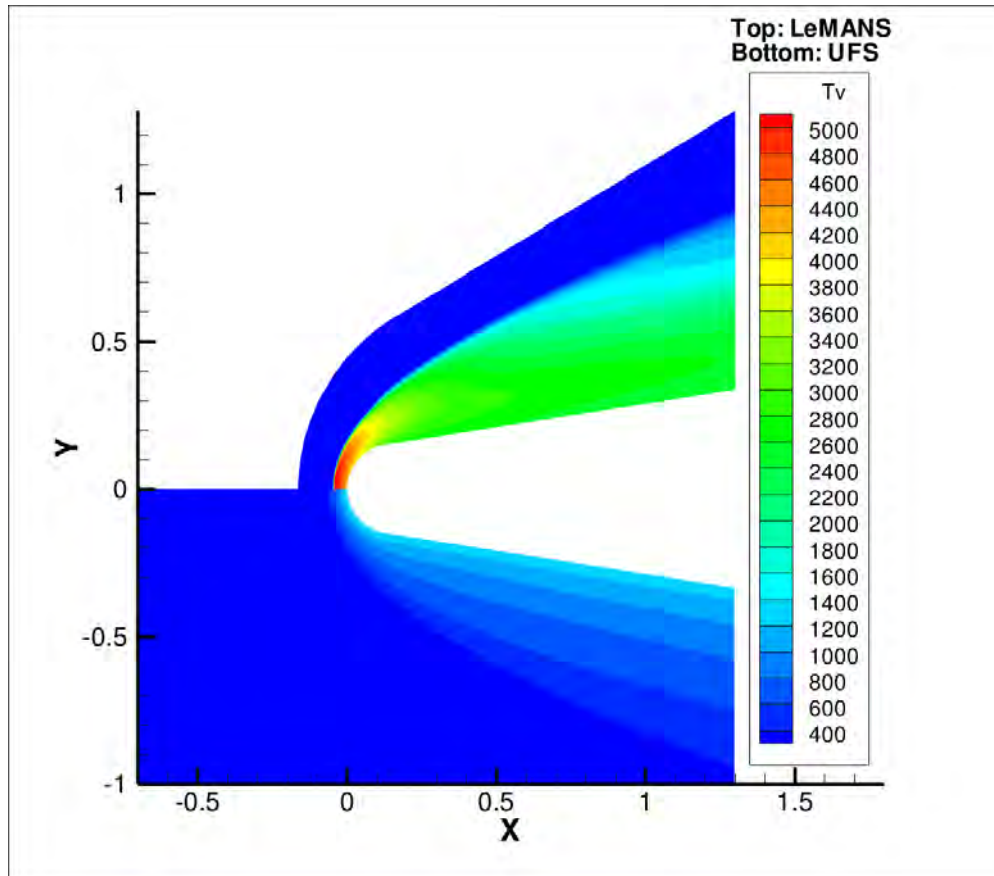
Figure 39. 11 Species Thermo-chemical Non-Equilibrium Flood Contour

The main reason for the differences is most likely due to how coarse the grid is in UFS but increasing the refinement of the grid would increase the computational expense significantly. Even with the coarse grid though, the solution should still have been closer between the two programs. Since the solutions are not closer, it suggests that the flux scheme implemented into UFS is too diffusive. The high diffusion would cause the thick shock and higher shock standoff distance. There could also be parameters missing from the input file, that were not mentioned in either the user's manual or the example simulations, that are necessary to run a thermo-chemical non-equilibrium simulation. The user's manual does not specify what input parameters required and the two example

simulations that were given have different input parameters from each other even though they both activate the chemistry module.
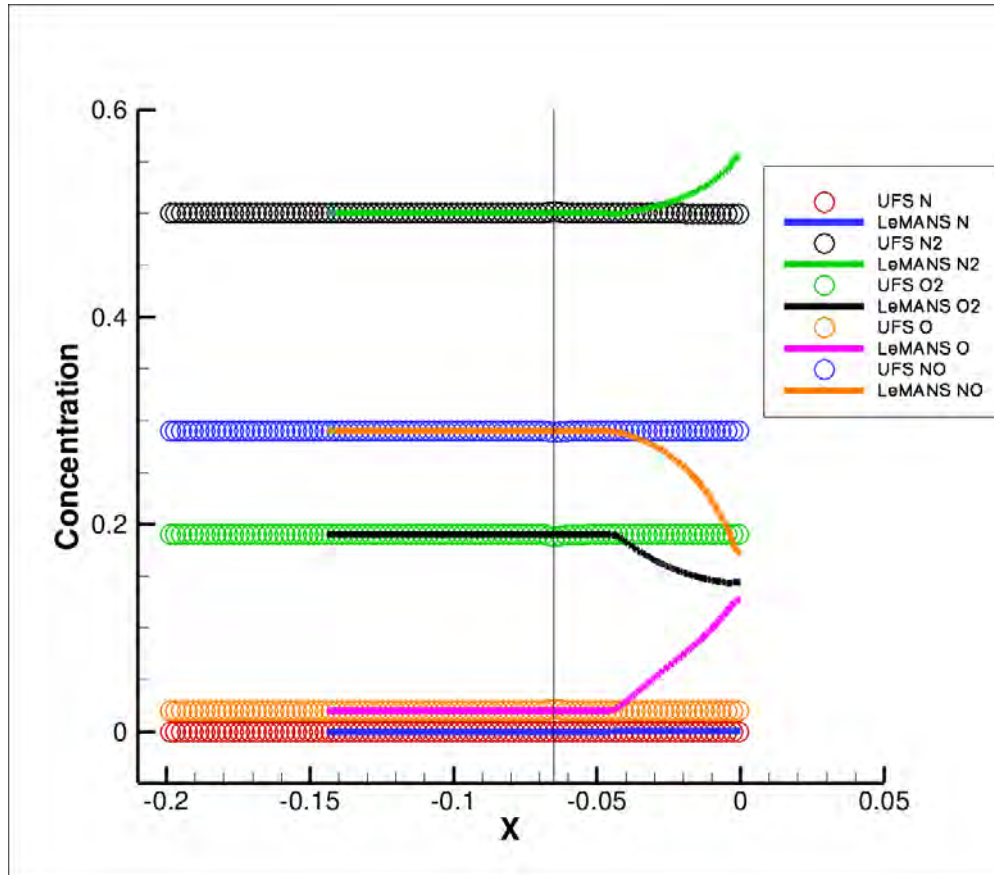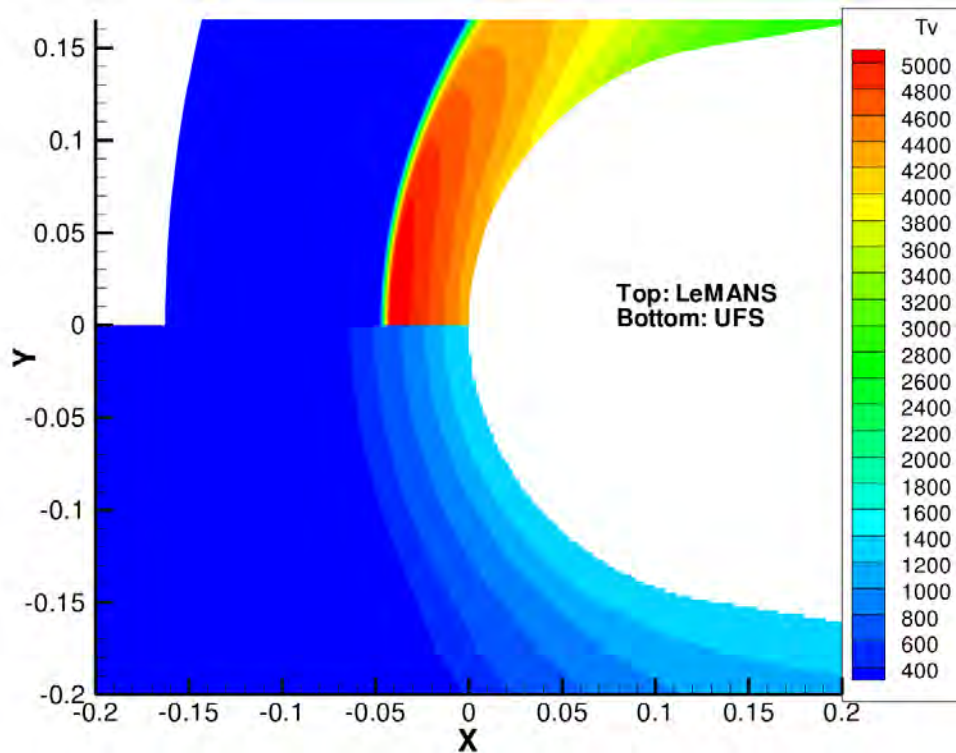

Figure 40. 11 Species Thermo-chemical Non-Equilibrium Stagnation Region Flood Contour (Vibrational Temperature)

The low vibrational temperature not only has an affect on the concentrations but also on the translational/rotational temperature as shown in Figure 42. The reason is that the low vibrational temperature corresponds to a low vibrational energy. The low vibrational energy means it did not take as much energy away from the translational/rotational mode causing the energy in the translational/rotation mode to stay high all the way up to the stagnation point. The higher energy causes the temperature to stay high as well, which is what is seen in Figure 41. Figure 41 also shows more clearly

how much of a difference there is in the shock standoff distance between UFS and

LeMANS.
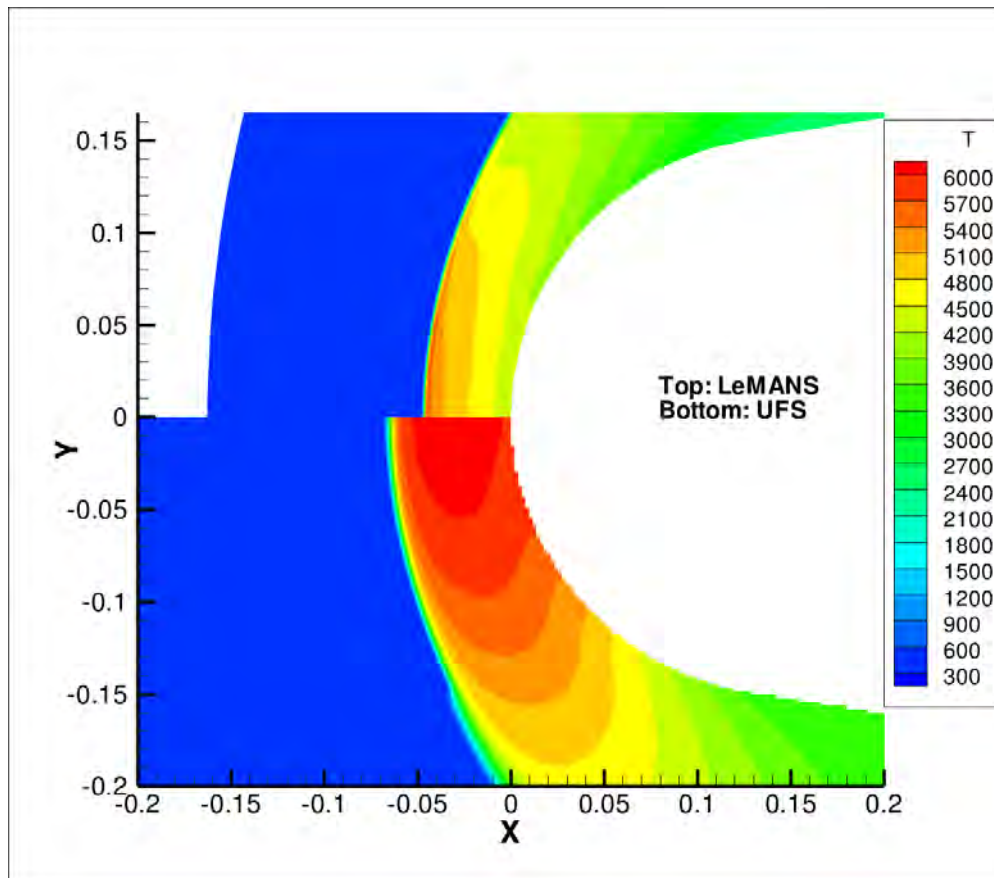
Figure 41. 11 Species Thermo-chemical Non-Equilibrium Stagnation Region Flood Contour (Translational/Rotational Temperature)

Figure 42 shows the stagnation line plot of the two temperatures. The plot also

shows the difference in the shock standoff distance along with the how the

translational/rotational temperature over shoots the translational/rotational temperatures

in LeMANS. Figure 42 also shows the difference between the stagnation temperatures

between UFS and LeMANS. Again the main reason for the difference is most likely the

coarseness of the grid used in UFS.
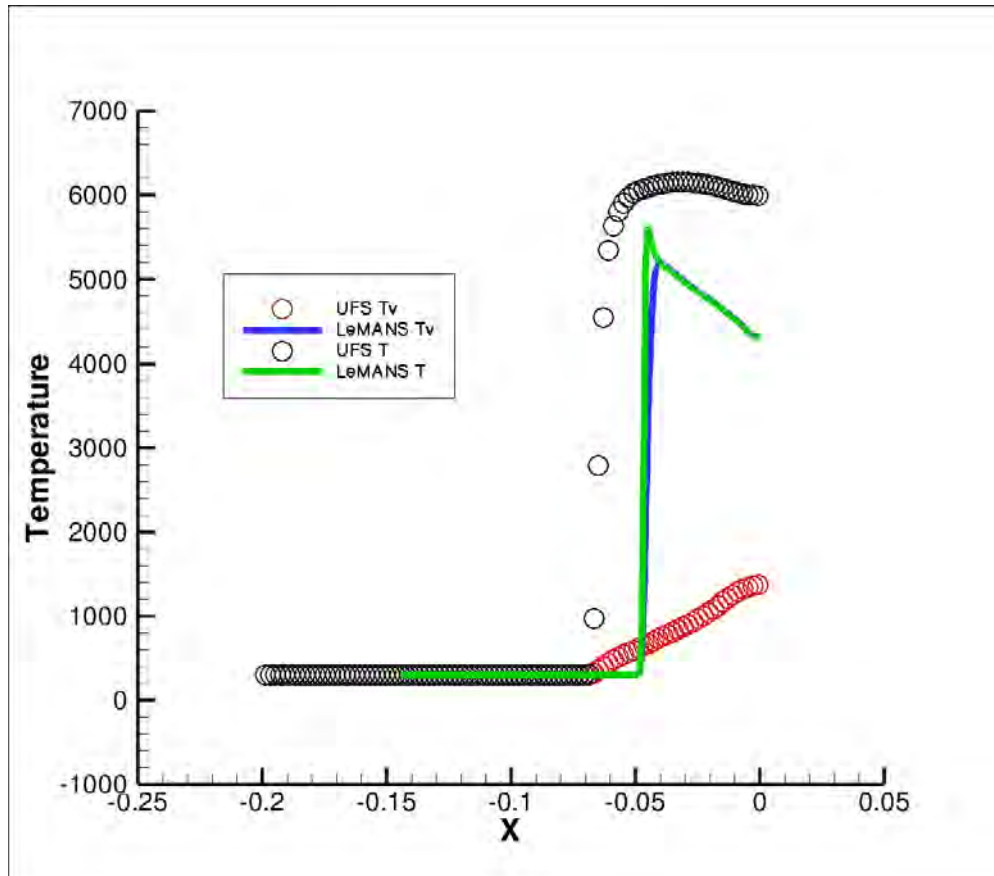
Figure 42. 11 Species Thermo-chemical Non-Equilibrium Stagnation Line Comparison

4.5 User Friendliness

Along with simulation comparisons another important aspect of validation is how easily someone can use the program. User friendliness in this context is going to be defined as how easily a user can setup a simulation, understand what settings are required, understand what the settings do, and get results that can be applied to real life. There are four man categories that are going to be focused on to compare UFS and LeMANS: setting up grid, post processing, and user's manual.

The first section and probably the most important is the user's manual because the user manual is there to explain the different aspects of a program along with the different settings. The user's manual for LeMANS is an example of a good user's manual. The reason for this is that LeMANS gives a brief explanation of the purpose of the code, how to install the code, how to create a grid, and how to actually run a simulation. From there LeMANS goes into each of the input files and gives an explanation of each of the different parameters in each of the input files. The explanation includes the different settings for the parameters and what the each setting does in the context of that parameter. The user's manual for UFS is good in the fact that it gives the purpose of the code, how to install, how to create a grid, and how to run a simulation. On the other hand though, UFS does not give an explanation of each of the parameters in the different input files. An example of this is seen in the test cases that were given by the developers of UFS. In one example file the parameters Electrons and NumberIons appear but in other example file the two parameters do not appear yet both are supposed to be examples of chemistry in UFS. Also neither parameter appears in the user's manual to explain what the two parameters do or why they might be in one example file but not the other.

The next segment is the post processing, which consists of being able to pull data from the simulation that can be applied to real life. For LeMANS the post processing is relatively simple in that all the data is outputted in metric units. The only down side is when trying to extract data along a line. In order to do this a macro was required, which can be difficult if the user has no prior experience with creating a macro. The extraction of data is one area where UFS does well because all that is required is the coordinates of the line along which the data is to be extracted. The other parts of post processing in UFS

are not as simple due to the fact that everything in UFS is non-dimensional. The problem with the data being non-dimensional is that in order to be able to compare to actually values the outputs must be dimensionalized. This would not be difficult if the normalizing factor for each variable was mentioned in the user's manual but some of the variables require looking through the source code to find out how the variable was non-dimensionalize. The final problem with UFS in post-processing is that UFS does not output the standard conventional variables. For example instead of outputting the vibrational or rotational temperature, UFS instead outputs the vibrational and rotational energies, which then must be converted to temperatures.

Setting up a grid in UFS is the best part of using UFS. All that is required to set up a grid is say how many initial boxes are needed and how they connect and then specify the level of refinement around the body and for the rest of the grid. The adaptive grid settings are the most difficult because the current user's manual does not have the current syntax, which the correct syntax can be found in the test cases, but does have the same variables and explains them well. Once the syntax is setup it is only a matter of varying the variables to match what the users wants. Even doing grid refinement study is only requires varying two different variables and then doing a comparison. For LeMANS it is more time intensive to create a grid as it requires another program that has grid generation capabilities and the outline of the grid must be created followed by putting the nodes of the grid along the boundary. After the initial grid has been created and a grid independence study is being done the grid generation program must be opened and the number of nodes along the boundary of the grid must be changed. Another downside for LeMANS in grid generation is that there are more refined cells where it is not necessary

because it was not known beforehand where the shock or other flow features would be located.

V. Conclusions and Recommendations

The purpose of this project was to validate the chemistry module of the Euler solver in the program Unified Flow Solver (UFS). UFS was compared to Le Michigan Aerothermodynamics Navier-Stokes Solver (LeMANS) using three different types of cases. The first case was a perfect gas case that tested both a monatomic and diatomic gas. The second case tested a diatomic gas in thermal non-equilibrium and the third case tested the full chemistry module of UFS using a 2-species and 11-species simulation.

The results from the perfect gas simulations showed that UFS could match very well with LeMANS for both the monatomic and diatomic simulations. Even though UFs did match well to LeMANS there were still some problems. First, the computational expense of UFS was more than 10 times greater than LeMANS. For the diatomic simulation specifically the activation of the vibrational mode, which was due to UFS not having the ability to run a single species diatomic simulation, affected the results.

The results from the thermal non-equilibrium matched closely between UFS and LeMANS except for the vibrational temperature. Again, as with the perfect gas simulations, the grid was limited due to computational expense, which affected the results. On the positive side this simulation did show that UFS was twice as fast per iteration for the same number of cells but the higher number of iterations still means that UFS took longer overall that LeMANS.

For the thermo-chemical non-equilibrium 2-species simulation UFS proved to not be anywhere close to matching the answer given by LeMANS. The simulation showed that due to a low vibrational temperature there was very little dissociation of O2. The low

vibration also affected the translational/rotational temperature. Also as with the other simulations previous, UFS proved to be much slower overall but was faster per iteration for the same number of cells.

The results from the 11 species thermo-chemical non-equilibrium simulation showed that the concentrations from UFS did not match up to LeMANS as in the 2 species thermo-chemical simulation. The lack of change in the concentrations was due to a low vibrational temperature, which also caused the translational/rotational temperature to be higher than LeMANS. Again UFS proved to be too computationally expensive because the cpu time used was over two times larger in UFS then LeMANS and the solution was not even close to matching.

Since the results for the two UFS chemistry simulations were so different from LeMANS, the 11 species UFS test case input file was compared to the input file used in the UFS chemistry simulation. One of the differences that were noticed was that the Mach number is the test case was set at 30 while this simulation only ran at Mach 10. This simulation was set to 30 but then it was realized that the simulation would not run unless a diffuse boundary condition was set. This caused problems because a diffuse boundary condition would no longer give an adiabatic boundary condition, which is a condition that must be set in order to use the Euler equations. At this point trying to run a Mach 30 case was abandoned since the point of this thesis was focusing on the Euler solver.

Overall UFS has proved to be much more computational expensive than LeMANS to reach a high-quality solution and causes UFS to not be a practical choice for

hypersonic or re-entry simulations. The reason for the expense is due to the requirement to have a small time step for stability because of the use of explicit time integration. Switching to an implicit time integration scheme would allow for a larger time step while not affecting the stability of the solution. Another way the computational expense could be reduced is by reducing the level of refinement required to reach a grid independent solution. The lower refinement level would reduce the total number of cells in the grid and it would allow for a larger time step because smaller cells require a smaller time step due to stability. In order to reduce the overall level of refinement the flux scheme must be changed because it appeared to be too diffusive. The diffusion was shown in the thermal and thermo-chemical non-equilibrium simulation through the thickness of the shock and the difference in shock standoff distance.

UFS has also proven not to be very user friendly as the user has to guess on the function and syntax of some inputs as the user's manual does not specify the function or syntax for most of the inputs that are required for the input file. The non-dimensionalization also make UFS less user friendly as the user is required look through the source code to find how a certain variable was normalized. The best way to fix the user friendliness would be to create a user's manual that talked about the different input functions and syntaxes. Also having the user's manual describe what each input file required does and also where the values come from.

The one benefit to using UFS is its use of a Cartesian grid and adaptive mesh refinement. This function made the initial grid set up very simple because the grid did not have to be refined to fit the final solution. This function also made the grid independence

study move faster because the all that had to be changed were some parameters in the input file. Finally this function allowed for everything about the grid to be done in the input file and did not require another grid generation program.

The first step for any future work is to finish validating the Euler solver in UFS. In order to do this the grid independence study should be finished so that solutions no longer depended on the grid. From there each case should be run again with the new grid to see if there are any errors that remain. Also for the chemistry simulations the inputs need to be clearly defined to make sure they are all accounted for in the input file. Finally to finish the validation of the Euler solver the VTRelaxModel parameter needs to investigated more to find out what the differences are between the settings and what effect each setting has on the solution.

The next step in the validation process of UFS would be to validate the Navier-Stokes solver. The validation would be done by running perfect gas, thermal non-equilibrium, and thermo-chemical non-equilibrium cases. These cases could be compared to LeMANS like the Euler solver. Also the flow conditions would have to give a Knudsen number between 0.01 and 0.1. From there the Boltzmann solver would need to be validated in both the continuum and rarified regimes using the same cases as the Euler and Navier-Stokes validation. The Boltzmann solver validation would have to be compared to a rarified CFD solver since LeMANS does not have is not valid in the rarified regime.

Finally, the validation of the coupling of continuum and Boltzmann solvers would be required. This validation would test to see if UFS can choose the correct solver where

required. The flow conditions for each simulation would have to give a Knudsen number above 0.1 but probably below 1 because some of the flow would be in the continuum regime and part of the flow would need to be in the rarified regime. Before any of this proposed work is done though, the computational expense of UFS must be reduced because currently UFS is not even a practical choice when using the Euler solver.

Appendix A

Example LeMANS Input File

```
IS_AXIS=0
IS_VISCOUS=0
IS_ADIAB=1
IS_SUPER_CAT=0
IS_RAD_WALL=0
IS_CHEM_REAC=0
IS_PREF_DIS=0
IS_NON_EQ=1
IS_LAURA=1
NS=2
NDS=0
RHO_INF_0=2.80192E-4
RHO_INF_1=1.408E-6
RHO_INF_2=0.0
RHO_INF_3=0.0
RHO_INF_4=0.0
RHO_INF_5=0.0
RHO_INF_6=0.0
RHO_INF_7=0.0
RHO_INF_8=0.0
RHO_INF_9=0.0
RHO_INF_10=0.0
V_INF_0=3533.89
V_INF_1=0.
V_INF_2=0.
TT_INF=300.0
TV_INF=300.0
TT_WALL=300.0
TV_WALL=300.0
Le=1.4
MOD_MILLIKAN=1
IS_GUPTA=0
IS_CLN=1
CFL=0.1.
CONV_CRITERION=1.E-12
MAX_TIME_STEP=1.E-4
MAX_CFL_NUMBER=1E10
IMPLICIT=2
IS_SECOND_ORDER=1
IS_MSW=1;
MAX_N_ITER=6000
GRAD_TYPE_CALC=2
PRINT_ITER=100
IS_RESTART=0
INV_RELAX=1.5
VISC_RELAX=1.0
GRID_FACTOR=1000.
MESH_FILENAME=bluntcone.cas
```

Example UFS Input File

```
Define MAX_ITER 150000
Define OUTPUT_INTERVAL 30000
Define MONITOR_INTERVAL 1

Define NUMCOMP 2
Define REFMASS 10.
Define MACH 10.
Define REFTEMP 300
Define REFDEN 2.816E-4
Define REFLEN 1

GModule gasdynamics

2 1 GfsGasdy GfsBox GfsGEdge {} {

Time { iend = MAX_ITER }


Global {

    static gdouble FLOW_11SP (guint species, guint var)
    {
      gdouble RHO, UVEL, TEMP, PRES, EV_EQ, ER_EQ;
      gdouble T1, U1, T2, U2;
      guint n = species - 1, n_index;
      gdouble nt, mt, Kt, gam;

      //--------- start user input --------------------
      guint   ncomp   = NUMCOMP;
      gdouble Mach    = MACH;
      gdouble RefMass = REFMASS;//reference mass in kg/kmole

      //allocate arrays (adjust size as necessary)
      guint ncomp_max = NUMCOMP;
      gdouble mass [ncomp_max];
      gdouble Krot [ncomp_max];
      gdouble VibEn[ncomp_max];
      gdouble DENS1[ncomp_max];

      gdouble MachSq = Mach*Mach;

      //set real masses in kg/kmole
      mass[0]  = 28.;
      mass[1]  = 14.;

      for (n_index = 0; n_index < ncomp; n_index++)
          mass[n_index] /= RefMass; //normalize mass to refmass

      Krot[0]  = 2.;
      Krot[1]  = 0.;
```

```
VibEn[0]  = 11.24;
VibEn[1]  = 0;

DENS1[0]  = .995;
DENS1[1]  = .005;

//get SUM RHO = 1 for SUM DENS1 = 1
for (n_index = 0; n_index < ncomp; n_index++) {
    DENS1[n_index] /= mass[n_index];
}

T1      = 1.; //same temperature
//---------- end user input ------------------------

nt = mt = Kt = 0.;
for (n_index = 0; n_index < ncomp; n_index++) {
    gdouble m   = mass [n_index];
    gdouble n_s = DENS1[n_index];

    nt +=                     n_s;
    mt +=               m * n_s;
    Kt += Krot[n_index] * n_s;
}

mt  /= nt;
Kt  /= nt;
gam  = 1.+2./(Kt+3.);

U1    = Mach*sqrt(gam/2.*T1/mt);
U2    = U1*(2.+(gam-1.)*MachSq)/MachSq/(gam+1.);
T2                    =       (2.*gam*MachSq-(gam-1.))*(2.+(gam-
1.)*MachSq)/MachSq/(gam+1.)/(gam+1.);

RHO  = mass[n] * DENS1[n];
UVEL = U1;
TEMP = T1;
PRES = RHO/mass[n] * TEMP;

if(VibEn[n] == 0.)
    EV_EQ = 0.;
else
    EV_EQ = VibEn[n]/( (exp( VibEn[n]/TEMP ) - 1.) * mass[n]
);

    if(Krot[n] == 0.)
      ER_EQ = 0.;
    else
      ER_EQ = Krot[n]/2.*TEMP/mass[n];

      //assign according to var index
      switch ( var ) {
          case 1:
            return RHO;
```

```
                    case 2:
                      return UVEL;
                    case 3:
                      return PRES;
                    case 4:
                      return EV_EQ;
                    case 5:
                      return ER_EQ;
                    default:
                    printf( "Error in SW1D\n" );
                    exit(0);
              }
        return 0.;
      }
}


Refine 4
RefineSolid 11

GtsSurfaceFile RamC_move2.gts

Init {} {

  rho_1  = FLOW_11SP (1, 1)
  u_c_1  = FLOW_11SP (1, 2)
  v_c_1  = 0.
  p_c_1  = FLOW_11SP (1, 3)
  ev_c_1 = FLOW_11SP (1, 4)
  er_c_1 = FLOW_11SP (1, 5)

  rho_2  = FLOW_11SP (2, 1)
  u_c_2  = FLOW_11SP (2, 2)
  v_c_2  = 0.
  p_c_2  = FLOW_11SP (2, 3)
  ev_c_2 = FLOW_11SP (2, 4)
  er_c_2 = FLOW_11SP (2, 5)


}

AdaptGradient { istart = 100 istep = 100 iend = MAX_ITER } {
    minlevel = 0.0
    maxlevel = {if ( (x >= 0.0  && x <= 0.5) && y <= -0.40 )
                         return 11;
                    else
                       return 9;}
    Cmax = .01     } log (rho_1)+log(sqrt(u_c_1*u_c_1+v_c_1*v_c_1))




OutputTime     { istep = MONITOR_INTERVAL } stdout
```

```
OutputLocation { istep = MONITOR_INTERVAL } monitor_point_data_1.dat
{0.2 -0.5 0}
OutputLocation { istep = MONITOR_INTERVAL } monitor_point_data_2.dat
{1.495 -0.165 0}

OutputGasdy        {  istep = OUTPUT_INTERVAL  } CYL_UFS-Euler_75km-
%06ld.sim {}
OutputLocation  {  istep  =  OUTPUT_INTERVAL  }  monitor_curve_data-
%06ld.dat monitor_curve.dat

}{
   ##gasdynamics input:
   SteadyState            = 1
   SolverType             = 0
   SolverOrder            = 0
   Limiter                = 0
   SurfaceBcType          = 0
   NumberComponents       = NUMCOMP
   Chemistry              = 0
   RefMass                = REFMASS
   RefMassDensity         = REFDEN        #kg/m3
   RefLength              = REFLEN        #m
   RefTemperature         = REFTEMP       #K
   VTRelaxModel           = 1
}

Box {
  left = Boundary {
    BcDirichletGasdy rho_1  FLOW_11SP (1, 1)
    BcDirichletGasdy u_c_1  FLOW_11SP (1, 2)
    BcDirichletGasdy p_c_1  FLOW_11SP (1, 3)
    BcDirichletGasdy ev_c_1 FLOW_11SP (1, 4)
    BcDirichletGasdy er_c_1 FLOW_11SP (1, 5)

    BcDirichletGasdy rho_2  FLOW_11SP (2, 1)
    BcDirichletGasdy u_c_2  FLOW_11SP (2, 2)
    BcDirichletGasdy p_c_2  FLOW_11SP (2, 3)
    BcDirichletGasdy ev_c_2 FLOW_11SP (2, 4)
    BcDirichletGasdy er_c_2 FLOW_11SP (2, 5)


  }

  top = Boundary {

  }

  bottom = Boundary {
    BcSymmetryGasdy rho_1
    BcSymmetryGasdy u_c_1
    BcSymmetryGasdy v_c_1
    BcSymmetryGasdy p_c_1
    BcSymmetryGasdy ev_c_1
```

```
        BcSymmetryGasdy er_c_1

        BcSymmetryGasdy rho_2
        BcSymmetryGasdy u_c_2
        BcSymmetryGasdy v_c_2
        BcSymmetryGasdy p_c_2
        BcSymmetryGasdy ev_c_2
        BcSymmetryGasdy er_c_2


    }
}

Box {

    right = Boundary {

    }

    top = Boundary {

    }

    bottom = Boundary {
        BcSymmetryGasdy rho_1
        BcSymmetryGasdy u_c_1
        BcSymmetryGasdy v_c_1
        BcSymmetryGasdy p_c_1
        BcSymmetryGasdy ev_c_1
        BcSymmetryGasdy er_c_1

        BcSymmetryGasdy rho_2
        BcSymmetryGasdy u_c_2
        BcSymmetryGasdy v_c_2
        BcSymmetryGasdy p_c_2
        BcSymmetryGasdy ev_c_2
        BcSymmetryGasdy er_c_2


    }
}

1 2 right
```

Appendix B

```
#!MC 1000

$!READDATASET  "'|DATASETFNAME|"'
  READDATAOPTION = NEW
  RESETSTYLE = YES
  INCLUDETEXT = NO
  INCLUDEGEOM = NO
  INCLUDECUSTOMLABELS = NO
  VARLOADMODE = BYNAME
  INITIALPLOTTYPE = CARTESIAN2D
  VARNAMELIST = '"X" "Y" "rho_O2" "U" "V"  "T" "rho" "P" "H" "Tv" "rho_O"'

$!TWODAXIS XDETAIL{RANGEMIN = -1.0}
$!TWODAXIS XDETAIL{RANGEMAX = 1.5}

$!ATTACHGEOM
  ANCHORPOS
    {
    X =  0.0
    Y =  0.0
    }
  RAWDATA
1
2
0 0
-1.0 0

$!PICK ADDATPOSITION
  X = 2.51668199295
  Y = 7.34404990403

$!EXTRACTFROMGEOM
  EXTRACTLINEPOINTSONLY = NO
  NUMPTS = 5000

$!RENAMEDATASETZONE
  ZONE = 2
  NAME = '|ZONENAME|'

$!WRITEDATASET  "stag.plt"
  INCLUDETEXT = NO
  INCLUDEGEOM = NO
  INCLUDECUSTOMLABELS = NO
```

```
  ASSOCIATELAYOUTWITHDATAFILE = NO
  ZONELIST =  [2]
  BINARY = YES
  USEPOINTFORMAT = NO
  PRECISION = 9

$!ATTACHGEOM
  ANCHORPOS
    {
    X = 0.0
    Y = 0.0
    }
  RAWDATA
1
601

$!PICK ADDATPOSITION
  X = 8.53396026957
  Y = 6.43395303327

$!EXTRACTFROMGEOM
  EXTRACTLINEPOINTSONLY = YES

$!RENAMEDATASETZONE
  ZONE = 3
  NAME = '|ZONENAME|'

$!ALTERDATA
  EQUATION = '{PHI} = ATAN2({Y},-({X}-0.1524))*180/PI'

$!WRITEDATASET  "surface.plt"
  INCLUDETEXT = NO
  INCLUDEGEOM = NO
  INCLUDECUSTOMLABELS = NO
  ASSOCIATELAYOUTWITHDATAFILE = NO
  ZONELIST =  [3]
  BINARY = YES
  USEPOINTFORMAT = NO
  PRECISION = 9
```

Bibliography

[1]   A. J. Lofthouse, "Nonequilbrium Hypersonic Aerothermodynamics using the Direct Simulation Monte Carlo and Navier-Stokes Models," PhD dissertation. The University of Michigan, 2008.

[2]   S. E. Summers, "Improved Collision Modeling for Direct Simulation Monte Carlo Methods," MS thesis, AFIT/GAE/ENY/11-M29, School of Engineering and Management, Air Force Institute of Technology, Wright Patterson AFB OH, , March 2011.

[3]   V. Kolobov, R. Arslanbekov, S. Bayyuk, V. Aristov, A. Frolova, S. Zabelok and F. Tcheremissine, "Unified Kinetic Approach for Simulation of Gas Flows in Rarefied and Continuum Regimes," Final Report for the Air Force SBIR Phase II Contract: FA8650-04-C-3404, 2007.

[4]   G. A. Bird, Molecular gas dynamics and the direct simulation of gas flows, New York: Oxford : Clarendon Press, 1994.

[5]   A. Lofthouse, "Unit 2 - Grid Generation," Aero 543, Advanced Computational Modeling for Aerodynamics, Graduate School of Engineering and Management, Air Force Institue of Technology, Wright Patterson AFB OH, 2011.

[6]   Y. Sone, "Kinetic Theory and Fluid Dynamics," Burkhausen, Boston, 2002.

[7]   V. V. Potkin, "A kinetic scheme for gas-dynamic equations in Lagrange coordinates," *J. Comp. Math. Math. Phys.,* no. 15, p. 227, 1975.

[8]   D. I. Pullin, "Direct simulation method for compressible inviscid ideal gas flow," *J. Comput. Phys.,* no. 34, p. 231, 1980.

[9]   R. D. Reitz, "One dimensional compressible gas dynamics calculations using the Boltzmann equations," *J. Comput. Phys.,* no. 42, p. 108, 1981.

[10] V. V. Aristov and F. G. Tcheremissine, "Solving the Euler and Navier-Stokes equations on the basis of the operator splitting of the kinetic equations," *Doklady USSR Acad. Sci,* no. 272, p. 555, 1983.

[11] V. V. Aristov and F. G. Tcheremissine, "The kinetic numerical method for rarefied and continuum gas flows in: O.M. Belotserkovskii et al. (Eds.)," *Rarified Gas Dynamics,* Vols. 1, Plenum Press NY, p. 269, 1985.

[12] J. C. Mandal and S. M. Deshpande, "Kinetic Flux vecot splitting for Euler equations," *Comput. Fluids,* no. 23, p. 447, 1994.

[13] T. Ohwada, "On the construction of kinetic schemes," *J. Comput. Phys.,* no. 177, p. 156, 2002.

[14] T. Ohwada and S. Kobayashi, "Management of discontinuous reconstruction in kinetic schemes," *J. Comput. Phys.,* no. 197, p. 116, 2004.

[15] S. Y. Chou and D. Baganoff, "Kinetic flux-vector splitting for the Navier-Stokes equations," *J. Comput. Phys.,* no. 130, p. 217, 1997.

[16] K. Xu, "A gas-kinetic BGK scheme for the Navier-Stokes equations and its connection with artificial dissipation and Godunov method," *J. comput. Phys,* no. 171, pp. 289-335, 2001.

[17] V. I. Kolobov, R. R. Arslanbekov, V. V. Aristov, A. A. Frolova and S. A. Zabelok, "Unified solver for rarefied and continuum flows with adaptive mexh and algorithm refinement," *Journal of Computational Physic,* no. 223, pp. 589-608, 2007.

[18] C. Park, Nonequilibrium Hypersonic Aerotherdynamics, John Wiley & Sons, 1990.

[19] D. Goodwin, "Cantera Object-Oriented Software for Reacting Flows," California Institute for Technology, 2006.

[20] W. R. Smith and R. W. Missen, Chemical Reaction Equilibrium Analysis: Theory and Algorithms, John Wiley & Sons Inc, 1982.

[21] W. C. Reynolds, "The Element Potential Method For Chemical Equilibrium Analysis: Implementation In The Iteractive Program STANJAN," Stanford University, Stanford CA, January 1986.

[22] L. C. Scalabrin, "Numberical Simulation of Weakly Ionized Hypersonic Flow over Reentry Capsules," PhD dissertation. The University of Michigan, 2007.

[23] A. J. Lofthouse, L. C. Scalabrin and I. D. Boyd, "Hypersonic Aerothermodynamics Analysis Across Nonequilibrium Regime using Continuum and Particle Methods," in *39th AIAA Thermophsyics Conference*, Miami, Florida, June 2007.

[24] A. J. Lofthouse, L. C. Scalabrin and I. D. Boyd, "Velocity Slip and Temperature Jump in Hypersonic Aerotherodynamics," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2007.

[25] T. E. Schwartentruber, L. C. Scalabrin and I. D. Boyd, "Modular Implementation of a Hybrid DSMC-NS Algorithm for Hypersonic Non-Equilibrium Flows," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, January 2007.

[26] T. E. Schwartzentruber, L. C. Scalabrin and I. D. Boyd, "Hybrid Particle-Continuum Simulations of Low Knudsen Number Hypersonic Flows," in *39th AIAA Thermophysics Conference*, Miami, Florida, June 2007.

[27] T. E. Schwartzentruber, L. C. Scalabrin and I. D. Boyd, "Hybrid Particle-Continuum Simulations of Non-Equilibrium Hypersonic Blunt Body Flow Fields," in *9th AIAA/ASME Joint Thermophysics and Heat Transfer Conference*, San Francisco, June 2006.

[28] P. A. Gnoffo, R. N. Gupta and J. L. Shinn, "Conservation equations and physical models for hypersonic air flows in thermal and chemcial nonequilibirum," NASA-TP-2867, NASA Langley, Hampton, Virginia, 1989.

[29] G. V. Candler, "The Computational of Weakly Ionized Flow in Nonequilibrium," PhD Dissertation, Stanford University, California, June 1988.

[30] J. H. Lee, "Basic governing equations for the flight regimes of aeroassisted orbital transfer vehicles," *Thermal Design of Aeroassisted Orbital Transfer Vechiles, AIAA, New York,* vol. 96, pp. 3-53, 1985.

[31] J. Burt, "LeMANS Users Guide," January 2008.

[32] "Unified Flow Solver Users Manual," February 2011.

[33] R. C. Millikan and D. R. White, "Systematics of Vibrational Relaxation," *Journal of Chem. Physics,* no. 39, p. 3209, 1963.

[34] E. Josyula and W. F. Bailey, "Governing Equations for Weakly Ionized Plasma Flowfields of Aerospace Vehicles," *Journal of Spacecraft and Rockets,* vol. 40, no. 6, pp. 845-857, 2003.

[35] W. G. Vincenti and C. H. Kruger Jr., Introdution to Physical Gas Dynamics, Malabar, Florida: Krieger Publishing Company, 2002.

1. REPORT DATE *(DD-MM-YYYY)* 22-3-2012	2. REPORT TYPE Master's Thesis	3. DATES COVERED *(From – To)* Aug 2010 – March 2012

TITLE AND SUBTITLE Validation of the Chemistry Module for the Euler Solver in Unified Flow Solver	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Humphrey, William C Civilian	5d. PROJECT NUMBER If funded, enter ENR #
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
   Air Force Institute of Technology
   Graduate School of Engineering and Management (AFIT/ENY)
   2950 Hobson Way, Building 640
   WPAFB OH 45433-8865
	8. PERFORMING ORGANIZATION
	REPORT NUMBER
	
	AFIT/GAE/ENY/12-M24

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research 875 N. Randolph Street Suite 325, Room 3112 Arlington, Virginia 22203 Dr. John D. Schmisseur (703) 696-6962 John.Schmisseur@afosr.af.mil	10. SPONSOR/MONITOR'S ACRONYM(S) **AFOSR**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In the world of computational fluid dynamics (CFD) three main types of flow regimes exist; continuum, rarified, and free molecular. Of these regimes the rarified regime is the most difficult to model because the continuum equations don't apply and using the Boltzmann equation is too computationally expensive to use. Unified Flow Solver (UFS) is currently being developed to solve this problem by using the kinetic continuum Euler equations where valid and only using the Boltzmann equation where necessary, thus reducing the computational cost. The use of the kinetic Euler equations helps to aid in the coupling of the Euler equations with the Boltzmann equation. This work compares UFS with a common non-equilibrium solver, LeMANS, to attempt to validate the thermo-chemical Euler solver available in UFS. Three types of simulations were run to validate the Euler solver; perfect gas, thermal non-equilibrium, and thermo-chemical non-equilibrium. The perfect gas simulation was run using both a monatomic and two species diatomic gas. The thermal non-equilibrium simulation was run using a 2 species gas while the thermo-chemical non-equilibrium was run using 2 and 11 species. The results of the simulations show that UFS matches closely for both the monatomic and 2 species perfect gas simulations as well as the thermal non-equilibrium simulation. The thermo-chemical non-equilibrium simulations do not show the correct vibrational temperature which causes the species concentrations to not be correct. All of the simulations show that UFS is much slower than LeMANS in number of cpu hours. This makes UFS not a practical choice for a CFD solver and cannot be fully validated in its current state.

15. SUBJECT TERMS
Unified Flow Solver, hypersonic, chemistry

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Andrew Lofthouse, Maj, USAF ADVISOR
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)* (937) 674-4043 (Andrew.lofthouse@wpafb.af.mil)
A	A	A	A	108	