UNSTRUCTURED TECHNOLOGY FOR HIGH SPEED FLOW SIMULATIONS

By

Michael Paul Applebaum

A DISSERTATION SUBMITTED TO THE FACULTY OF

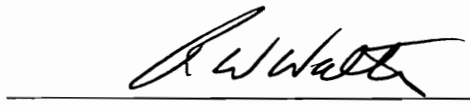
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

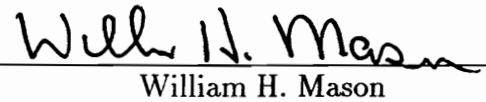
AEROSPACE ENGINEERING



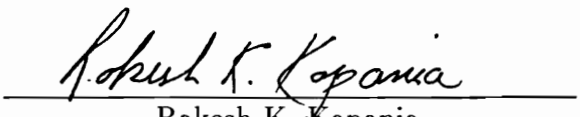
Robert W. Walters, Chairman

Bernard Grossman

Joseph A. Schetz



William H. Mason



Rakesh K. Kapania

December 1994

Blacksburg, Virginia

UNSTRUCTURED TECHNOLOGY FOR HIGH SPEED FLOW SIMULATIONS

by

Michael Paul Applebaum

Committee Chairman: Robert W. Walters
Aerospace Engineering

(ABSTRACT)

Accurate and efficient numerical algorithms for solving the three dimensional Navier Stokes equations with a generalized thermodynamic and chemistry model and a one equation turbulence model on structured and unstructured mesh topologies are presented. In the thermo-chemical modeling, particular attention is paid to the modeling of the chemical source terms, modeling of equilibrium thermodynamics, and the modeling of the non-equilibrium vibrational energy source terms. In this work, non-equilibrium thermo-chemical models are applied in the unstructured environment for the first time.

A three-dimensional, second-order accurate *k-exact* reconstruction algorithm for the inviscid and viscous fluxes is presented. Several new methods for determining the stencil required for the inviscid and viscous *k-exact* reconstruction are discussed. A new simplified method for the computation of the viscous fluxes is also presented.

Implementation of the one equation Spalart and Allmaras turbulence model is discussed. In particular, an new integral formulation is developed for this model which allows flux splitting to be applied to the resulting convective flux.

Solutions for several test cases are presented to verify the solution algorithms discussed. For the thermo-chemical modeling, inviscid solutions to the three dimensional Aeroassist Flight Experiment vehicle and viscous solutions for the axi-symmetric Ram-II C are presented and compared to experimental data and/or published results. For the hypersonic AFE and Ram-II C solutions, focus is placed on the effects

of the chemistry model in flows where ionization and dissociation are dominant characteristics of the flow field. Laminar and turbulent solutions over a flat plate are presented and compared to exact solutions and experimental data. Three dimensional higher order solutions using the *k-exact* reconstruction technique are presented for an analytic forebody.

Acknowledgements

The research presented in this dissertation was made possible in large part due to AFOSR grant F49620-92-J-0085 under the direction of the technical monitor, Dr. Len Sakell. The author is indebted to Dr. Robert Walters, whose guidance over the last four years has been invaluable. A huge thank you goes out to Dr. William McGrory and Dr. Curtis Mitchell, whose knowledge of the unstructured fluid dynamics environment has been of tremendous value during this research. A special thank you goes out to my family for the love and support they have provided over the years. And finally, to my friends in the lab whose help and friendship have made this experience memorable.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Computational fluid dynamics (CFD) has steadily gained acceptance from the aerospace community as a beneficial tool aimed at reducing the cost of designing and maintaining aerospace vehicles. As the use of CFD grows in the aerospace industry, new applications are conceived which require more sophisticated algorithms to model the various physical processes associated with current vehicle design. Recent interest in hypersonic vehicle design and in particular the National Aerospace Plane (NASP) demonstrates one such application. One of the characteristics of hypersonic flow fields is high temperature or real gas effects, where chemical and thermal non-equilibrium are important in the modeling of the physics. To accurately solve for the flow field around such configurations, these effects must be taken into account in the solution procedure.

Finite volume fluid dynamics codes can be divided into two main groups; structured and unstructured (or general indexing) codes. Each has its limitations and benefits. Up to this point, the main focus and research effort has been in structured flow solver technology. During the past two decades, powerful numerical methods for solving the Navier-Stokes equations with real gas effects on structured meshes have been developed (see the historical perspective). However, during the same time period, little in the way of progress has been accomplished in the area of efficiently

solving the Navier-Stokes equations with real gas effects on unstructured meshes, in either two dimensions or three dimensions. Most of the recent successes in unstructured technology have been in the area of solving three dimensional inviscid perfect gas flows or two dimensional perfect gas viscous flows. While technically and theoretically simple, the implementation of the viscous fluxes along with finite rate chemistry models in practice becomes complicated by the nature of unstructured flow solvers. This research takes a step in bringing unstructured technology into the realm where structured codes have dominated for so long. However, there is still a lot of research that needs to be accomplished before unstructured flow solvers are competitive with their structured counterparts, particularly in three dimensions.

At this point, you are probably wondering why we need to bring unstructured fluid dynamics codes in line with the structured technology if so much progress has been made in this area. The answer to this question lies in the relationship between the grid and solution. This relationship has played a vital role in the success and failure of structured fluid dynamics packages. From a basic understanding of CFD, we know that the quality of the spatial discretization of the physical domain has a direct and significant bearing on the accuracy of the solution. Not only is the accuracy affected, but the stability of the numerical algorithm is strongly affected by the quality of the spatial discretization. Numerical algorithms for solving the Navier Stokes and even the inviscid Euler equations are very sensitive to irregularities in the mesh. In particular, skewed and/or distored cells adversely affect the robustness of these numerical algorithms. In many situations, these mesh irregularities cannot be avoided. This is particularly true of structured packages which rely solely on quadrilateral and hexahedral cell types for the discretization. This lack of flexibility in the spatial discretization has been a large limiting factor in structured computational fluid dynamic packages.

By its very nature, unstructured technology is better equipped to handle these situations. To see this, we need to discuss the differences between structured and unstructured meshes. A structured mesh is defined by its indexing. That is to say, given any cell in the domain and its indices i, j, and k, one can determine where every other cell in the domain resides in relation to this cell. Figure 1.a shows a

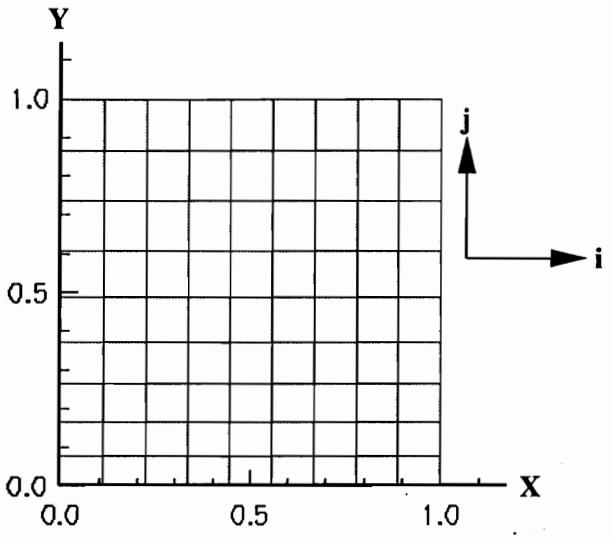


Figure 1.a: Sketch of a typical structured mesh

simple two dimensional structured mesh with the indices i and j defined as shown. In this case, a cell is defined by the values of i and j as are the nodes and faces. For $i = 0$ we know that we are on a boundary of the computational domain. The same is true for $j = 0$, $i = imax$ and $j = jmax$. The fact that we can determine where any cell resides based on its i and j index restricts structured grids to quadrilateral and hexahedral cell types. As a result of the indexing scheme, structured grids are inherently directional. In other words, the i index has a direction associated with it.

In contrast, unstructured grids are characterized by a lack of the need for directionality. In an unstructured environment, the domain may be broken down into cells of any type or shape. The only restrictions being that the cells do not overlap and the entire computational domain is discretized. In other words, gaps or holes are not allowed in the discrete representation of the physical space. For unstructured meshes in two dimensions, the domain is typically sub-divided into triangles and/or quadrilaterals. Triangles allow the greatest flexibility in the discretization of complex surfaces and are dominant in two dimensional unstructured research. In three dimensions, the domain is typically sub-divided via tetrahedra, hexahedra, and prisms. Figure 1.b shows a representation of the same computational domain using triangles.

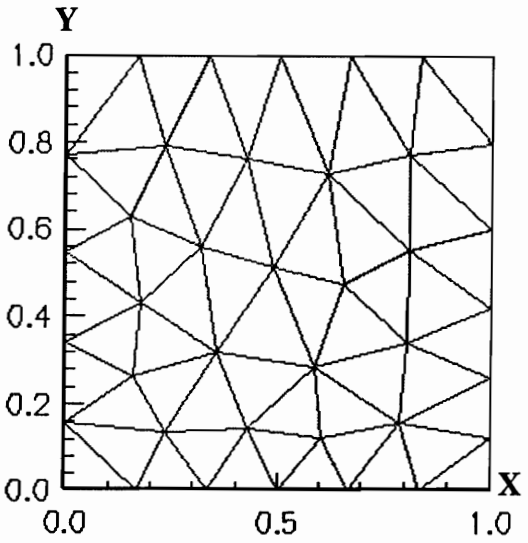


Figure 1.b: Sketch of a typical unstructured mesh

Obviously, i and j have no meaning here. No direct method exists for determining where one cell resides in relation to another without more information. This information comes in the form of pointers which tell a fluid dynamics code which faces and nodes surround which cells. Obviously, it would be trivial to take a structured grid and convert it into a format acceptable for an unstructured solver. In fact many of the solutions presented here were obtained in this manner. However, the opposite is not possible. There is no method which would allow the unstructured grid in figure 1.b to be represented in a format acceptable to a structured flow solver. Therefore, we may think of the structured environment as a simplification of the unstructured environment. The added flexibility from the unstructured flow solver allows the domain to be sub-divided in any manner the user desires.

For bodies with high curvature, the discretization limitations associated with the structured technology directly affects the solution quality regardless of how advanced the solution algorithms become. Therefore, if we are going to be able to accurately predict flows about complex geometries, the unstructured technology needs to be brought up to current structured capabilities. In addition to improved control over the discretization process, unstructured technology offers an optimum setting for

solution adaptation and problems with moving boundaries.

However, at the current time, unstructured flow solvers are lacking in the ability to predict problems of interest today. This is partially to blame on the lack of confidence in the unstructured grid generators available today. But the fact remains the same, improvements to unstructured flow solvers were at a stand still until the early 1990's. Even today, the current state of the art production level unstructured flow solvers are essentially limited to perfect gases and typically only capable of solving the Navier Stokes equations in two dimensions. Turbulence modeling on unstructured grids has gone relatively unnoticed until very recently and is still in its infancy.

1.2 Historical Perspective

Many of the topics which will be discussed in the dissertation have a long and rich history. These topics include upwind algorithms for flows in chemical and thermal nonequilibrium, reconstruction, and turbulence modeling.

One of the most important advances in computational fluid dynamics is the development of upwind methods for the modeling of the inviscid convective fluxes. The origin of upwind methods can be traced back to Courant, Isaacson and Rees [1]. Their work laid the foundations for the upwind methods applied to today's fluid dynamics code. The Courant, Isaacson and Rees scheme is based on a finite difference discretization of the first order, linear, advection equation $u_t + au_x = 0$. From a Von Neumann stability analysis it is easy to show that applying a central difference for the spatial derivative with Euler explicit time integration leads to an unstable scheme. With a forward difference for the spatial term, the scheme is stable for negative values of a but unstable for positive values of a. With a backward difference for the spatial derivate, the opposite is true. The scheme becomes stable for positive values of a but unstable for negative values of a. Therefore, by discretizing the spatial derivative based on the sign of a, one can arrive at a stable scheme for both positive and negative values of a. In the discretization process we have introduced the physical characteristics of the equation into the discretization. This forms a basis for the family of upwind schemes for the Euler equations, commonly referred to as flux-splitting

techniques. Flux split algorithms were originally developed for perfect gas flows. A survey of these methods can be found in Harten, et.al. [2] and Roe [3]. The upwind algorithms for the Euler equations can be broken down into two classes; flux vector splitting and flux difference splitting or Godunov-type methods. Two of the more popular flux vector splitting algorithms for perfect gases are the algorithms proposed by Steger and Warming [4] and Van Leer [5]. In 1959, Godunov [6] introduced the exact solution of the Riemann (shock tube) problem into the discretization of the Euler fluxes. Flux difference schemes are based on the approximate solution to the Riemann and are appropriately referred to as Godunov-type methods. One of the most widely used flux differencing schemes for perfect gases is due to Roe [7].

The flux vector splitting and flux difference splitting algorithms have been extended to flows in which the perfect gas assumption is invalid. Extension of these algorithms to flows in chemical equilibrium have been accomplished by several researchers. Most notable among these are the methods developed by Colella and Glaz [8], Vinokur and Liu [9] [10] [11], Glaister [12] [13], and Grossman and Walters [14] [15]. The flux splitting algorithms have also been extended to flows in chemical and thermal non-equilibrium. Most notable among these efforts are the works of Liu and Vinokur [16], Walters et al [17], Candler and MacCormick [18], and Grossman and Cinnella [19] [20] [21] [22].

The works mentioned up to this point have one common characteristic. They were all applied in structured fluid dynamics codes. Perfect gas upwind algorithms have been applied to unstructured meshes. Most notable among these are the works of Batina [23], Barth et al [24] [25], and Mitchell [26].

The spatial accuracy of the upwind methods are dependent on the accuracy of the methods used to determine the left and right states at the cell interface. Most structured upwind codes in use today incorporate a variable extrapolation method commonly referred to as MUSCL (Monotone Upstream Schemes for Conservation Laws). The MUSCL variable extrapolation method was first introduced by Van Leer [27]. The traditional $\phi - \kappa$ MUSCL formulation [28] [29] cannot be applied to the unstructured environment without additional (and questionable) interpolation. An often overlooked but underlying principle of the traditional $\phi - \kappa$ MUSCL formulation

is the idea of reconstruction. Reconstruction generates a multi-dimensional pointwise distribution function from the known cell averages with the stipulation that integration of the distribution function exactly recovers the cell averages. The reconstruction principle has been used by several authors for producing second order accurate unstructured upwind codes. The most notable research efforts include the linear based gradient methods proposed by Barth et. al. [24] and Frink et. al.[30], and the *k-exact* reconstruction algorithms proposed by Barth et. al. [31] [32] and later extended by Mitchell and Walters [33] [26].

The modeling of the viscous terms in the Navier Stokes equations in unstructured flow solvers has been progressing slowly to this point. Notable among these works are Barth [34], Frink [35], and Mitchell [26]. Frink's work is especially notable due to its application for the three dimensional Navier Stokes equations. The other two works cited are limited to two dimensions.

Like the modeling of the viscous fluxes, turbulence modeling has been progressing at a snail's pace. Obviously, without adequate methods for modeling the viscous terms, turbulence modeling has had to take a back seat. The history of turbulence modeling on structured grids is very rich and controversial. The history is too long to provide here, however some good survey papers are available. Schetz [36] provides an excellent introduction and survey of the different classes of turbulence models, Wilcox [37] provides a good description of the more popular models typically implemented in CFD codes, Coakley [38] provides a good survey of turbulence models for compressible flows, and Rumsey [39] provides a good comparison of the predictive capabilities of several of the turbulence models in use today. Turbulence models for unstructured meshes have just recently become available. Two models have been developed for use in unstructured fluid codes. These are the models proposed by Baldwin and Barth [40] and Spalart and Allmaras [41] [42].

1.3 Significance of the Research

During the research into unstructured technology, some time was spent helping develop a structured flow solver capable of solving problems of interest in today's environment. In particular, this meant adequate thermo-chemical modeling, modeling of the viscous fluxes, and turbulence modeling. All three items are vital to the accurate simulation of the complex problems of interest today. From this early work, it was painfully evident that the current state-of-the-art unstructured flow solvers were incapable of solving these problems. At the time when this research began, all three were entirely missing in the unstructured environment. Fortunately, research in the unstructured environment has been increasing over the last several years and progress is being made in these areas.

Of the three missing parts, thermo-chemical modeling has witnessed the slowest growth in the unstructured environment. Application of upwind algorithms for flows in chemical and thermal non-equilibrium have been applied successfully to structured flow solvers for many years [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22]. However, application of these upwind algorithms in unstructured codes has been largely neglected. Two solutions are presented here in which thermo-chemical effects are significant. To our knowledge these are the first unstructured solutions presented for the aeroassist flight vehicle experiment and RamII-c. In both solutions, dissociation reactions (whereby chemical combinations are converted into simpler constituents) and ionization reactions (whereby constituents are converted to positively and negatively charged ions) have a significant effect on the flow field.

The second part lacking in unstructured work was adequate methods for interpolation of the cell averaged quantities to the faces required for the upwind methods. This is typically referred to as reconstruction. Most of the reconstruction work accomplished up to that point had been limited to two dimensions. Walters and Mitchell [26] [33] derived a robust and simple method for applying the *k-exact* reconstruction method to perfect gas upwind solvers in two dimensions. Their work has been extended to the real gas upwind solvers in three dimensions in the present research.

The third part missing from unstructured solvers was the ability to solve the Navier

Stokes equations in three dimensions. Before the viscous fluxes could be computed, adequate methods for the reconstruction of the cell averaged solution variables needed to be developed. As a result, algorithms for the modeling of the viscous fluxes have only recently been developed. Mitchell and Walters [26] [33] did some early work in this field in two dimensions. In this work, results for the axi-symmetric RamII-c are presented which use this viscous formulation.

Another part missing from the picture was turbulence modeling. Most of the flows of concern to current vehicle design are not laminar, but are turbulent. Therefore, application of an acceptable turbulence model is important to modeling problems of interest in todays environment. As of this writing, two turbulence models have been developed with unstructured flow solvers in mind. Neither model has been extensively tested in actual unstructured flow solvers. The unstructured environment presents new problems for the turbulence model investigator. Typically these models require information not available in the unstructured environment. For example, models such as the Baldwin and Lomax model rely on surveying the velocity or vorticity profile on a smooth grid line, roughly orthogonal to the surface. Something which is either impossible or infeasible in the unstructured environment. Spalart and Allmaras [41] [42] developed a one equation turbulence model with unstructured flow solvers in mind. In this research, their model has been implemented and tested in two dimensions. While the formulation theoretically should work in three dimensions, validation of their model in three dimensions has not been performed.

1.4 Scope of the Investigation

The objective of this research was to bring unstructured technology within reach of the current capabilities of structured technology. In other words, to at least partially bridge the gap that has existed between the two technologies. Emphasis in the research was focused on three areas of immediate need. The first was the implementation of a generalized thermo-chemical model. This is accomplished in three stages. The first stage is the implementation of acceptable upwind algorithms for the inviscid fluxes. Here we discuss three upwind algorithms which have been extended

to generalized thermo-chemical models. These include the Steger and Warming flux vector splitting, Van Leer flux vector splitting, and Roe's flux difference splitting techniques. The formulation used for the unstructured fluid dynamics code follows the work of Grossman and Cinnella [20] [21] [22]. Their work is briefly reviewed here. The second stage is adequate modeling of the chemical source terms. These source terms represent the rate of conversion of one species to another via chemical reactions. In this discussion, the source terms are derived from empirical relations for the forward and backward reaction rates. The computation of the forward and backward reaction rates are also discussed. The last stage is the modeling of the thermodynamics. Three equilibrium thermodynamic models are discussed here. The first is a simple perfect gas formulation where the species are assumed to be calorically perfect. The second approach assumes the species are thermally perfect gases and is based on statistical mechanics for the computation of the vibrational contribution to the internal energy. The third approach is via experimental curve fits provided by the Lewis Research Center [43]. In addition to the equilibrium thermodynamic models, the source terms which account for species considered to be in thermal non-equilibrium are derived from statistical mechanics. The implementation of the source terms into the unstructured code is presented.

The second research topic in which we focus our attention is the modeling of the viscous fluxes. A topic related to this and the upwind algorithms is reconstruction. Here we focus primarily on the *k-exact* reconstruction technique originally proposed by Barth [31] and later extended by Walters and Mitchell [33]. The extension of the Walters and Mitchell method to three dimensions is presented. In addition to *k-exact* reconstruction, linear gradient-based reconstruction methods are briefly reviewed. Following the reconstruction discussion, the modeling of the viscous fluxes is presented. The viscous fluxes require the computation of gradients in the solution variables. Derivatives of the reconstruction polynomial obtained in the *k-exact* reconstruction step are used to determine these gradients. Issues such as modeling the transport properties for multi-component mixtures is also presented.

The last research topic presented is turbulence modeling. Here we have taken the differential transport equation of the one equation Spalart and Allmaras [41]

[42] model and transformed it into an integral equation consistent with the other governing equations. The convective flux is modeled using a Van Leer type flux vector splitting technique. Gradients in the diffusive flux are modeled using derivatives of the reconstruction. This discussion presents an in depth view of the implementation into the current unstructured code.

To round out the discussion, the methods used for advancing the solution in time are presented. Here we discuss three time integration schemes. The time integration schemes discussed include an explicit M-stage Jameson-style Runge-Kutta [44], Euler implicit with Jacobi relaxation [44], and Euler implicit with the generalized minimum residual (GMRES) method [45]. To speed up convergence to the steady state, nested iteration also known as mesh sequencing has been implemented into the unstructured code. A brief discussion on the implementation of the mesh sequencing is provided here. This was vital in obtaining solutions to some of the more complex flow fields, such as the Aeroassist Flight Experiment Vehicle.

Although this research goes a long way towards bridging the gap between current unstructured and structured fluid dynamic packages, there is still much work that needs to be accomplished in the unstructured area. The efficiency of structured flow solvers is still an order or magnitude ahead of current unstructured flow solvers for complex, viscous three dimensional problems. In this research, little has been done in this area.

Chapter 2

Governing Equations

2.1 Introduction

In this chapter, the integral formulation of the Navier-Stokes equations with a generalized thermo-chemical model are given. Many of the terms used throughout the dissertation are introduced in this chapter. The actual implementation of the different elements of the governing equation are discussed in detail in later chapters. In addition, the following sections do not discuss turbulence modeling. This will be introduced in a separate chapter where its governing equation will be developed.

2.2 Integral Conservation Equations

We begin the development of our numerical algorithm with the integral form of the three dimensional Navier Stokes equations,

$$\frac{\partial}{\partial t}\iiint_V Q\ dV + \oint_S \vec{F}\cdot\hat{n}\ ds - \oint_S \vec{G}\cdot\hat{n}\ ds = \iiint_V W\ dV \tag{2.1}$$

where Q represents the vector of dependent variables, W is the source term due to chemical reactions and non-equilibrium thermodynamics, $\vec{F}\cdot n$ and $\vec{G}\cdot n$ represent the inviscid and viscous flux of mass, momentum, and energy out of the control volume V through the surface S with $\hat{n}$ an outward pointing unit normal vector from S. In order to include real gas effects, the standard system of five equations has been augmented

to include the effects of a generalized thermo-chemical model due to Grossman et.al. [19]. For our purposes in this chapter, the effect is to replace the global continuity equations by N species continuity equations, where N is the number of species in the chosen chemistry model. In addition, to account for M species considered to be in thermal non-equilibrium, the system has been augmented to include M non-equilibrium energy conservation equations. In light of this, the vector of dependent variables and thermo-chemical source terms are given by:

$$Q = \begin{pmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_N \\ \rho u \\ \rho v \\ \rho w \\ \rho_1 e_{n_1} \\ \rho_2 e_{n_2} \\ \vdots \\ \rho_N e_{n_M} \\ \rho e_o \end{pmatrix}, \qquad W = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_N \\ 0 \\ 0 \\ 0 \\ \dot{Q}_1 \\ \dot{Q}_2 \\ \vdots \\ \dot{Q}_M \\ 0 \end{pmatrix} \tag{2.2}$$

where ρ_i is the i^{th} species density, ρ is the mixture density, u, v, and w are the components of the velocity vector in the $\hat{i}$, $\hat{j}$, and $\hat{k}$ directions respectively, e_{n_i} represents the i^{th} species non-equilibrium energy contribution, e_o is the total energy, ω_i represents the source terms due to chemical reactions, and Q_i represents the source terms due to non-equilibrium thermodynamics. In equation (2.1) we assume body forces are negligible. The governing equations are closed by an equation of state relating the species densities, pressure and energy. The chemical source terms and their implementation are discussed in chapter 3. The equation of state and equilibrium and non-equilibrium thermodynamics are discussed in chapter 4.

The inviscid flux vector $\vec{F}$ consists of components from the three principal directions,

$$\vec{F} = f\hat{i} + g\hat{j} + h\hat{k} \tag{2.3}$$

where the flux components f, g, and h are given by:

$$f = \begin{pmatrix} \rho_1 u \\ \rho_2 u \\ \vdots \\ \rho_N u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ \rho_1 e_{n_1} u \\ \rho_2 e_{n_2} u \\ \vdots \\ \rho_M e_{n_M} u \\ \rho h_0 u \end{pmatrix}, \quad g = \begin{pmatrix} \rho_1 v \\ \rho_2 v \\ \vdots \\ \rho_N v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ \rho_1 e_{n_1} v \\ \rho_2 e_{n_2} v \\ \vdots \\ \rho_M e_{n_M} v \\ \rho h_0 v \end{pmatrix}, \quad h = \begin{pmatrix} \rho_1 w \\ \rho_2 w \\ \vdots \\ \rho_N w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ \rho_1 e_{n_1} w \\ \rho_2 e_{n_2} w \\ \vdots \\ \rho_M e_{n_M} w \\ \rho h_0 w \end{pmatrix}. \tag{2.4}$$

The discretization of the inviscid fluxes is accomplished by means of either flux-vector or flux difference splitting techniques. These algorithms have been extended from their original perfect gas formulation to include the effects of equilibrium and non-equilibrium chemistry and thermodynamics (see historical perspective in the Introduction).

Two of the more popular flux-vector split algorithms have been implemented into the unstructured flow solver. These are the algorithms developed by Steger and Warming [4] and Van Leer [5]. The basic idea of both methods is to split the inviscid flux vector in one dimension into two parts, one which contains the information that propagates downstream and one which contains the information that propagates upstream. The two parts are then constructed using extrapolation formulas consistent with the direction of propagation. Extension of the algorithms to more spatial dimensions is accomplished by superimposing pseudo-one dimensional problems. The flux vector splitting techniques have proven to be accurate and robust when used for hypersonic flows [22] and are fully compatible with the conservative finite-volume approach.

In addition to the flux-vector splitting techniques, the flux difference splitting technique originally developed by Roe [7] has been implemented into the unstructured

flow solver. While this method has proven to be less robust for hypersonic flows [22], it has been shown to be more accurate than the flux-vector split techniques. The flux difference splitting technique consists of an approximate Riemann solver, where an arbitrary discontinuity is supposed to exist between the left and right state. An approximate solution is written for this situation in terms of waves propagating upstream and downstream. In this case, the flux is not split, but reconstructed from the upstream and downstream contributions that constitute the left and right state of the Riemann problem.

The flux-vector splitting algorithms of Steger and Warming and Van Leer and the flux-difference splitting algorithm of Roe are discussed in chapter 5. A subject related to the upwind algorithms is the interpolation of the cell averaged solution variables to the faces. In the present formulation, this is accomplished using several reconstruction techniques. These include the *k-exact* polynomial reconstruction and a linear gradient based reconstruction. These will be discussed in detail in chapter 6.

Similar to the inviscid flux vector, the viscous flux vector $\vec{G}$ may be written in component form as:

$$\vec{G} = f_v\hat{i} + g_v\hat{j} + h_v\hat{k} \tag{2.5}$$

where the viscous flux components f_v, g_v, and h_v are given by:

$$f_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_x - q_x \end{pmatrix}, g_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_y - q_y \end{pmatrix}, h_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_z - q_z \end{pmatrix}. \tag{2.6}$$

The stresses in viscous fluxes are written in terms of the velocity gradients via:

$$\tau_{xx} = \mu\left(2u_x - \frac{2}{3}(u_x + v_y + w_z)\right) \quad (2.7)$$

$$\tau_{yy} = \mu\left(2v_y - \frac{2}{3}(u_x + v_y + w_z)\right) \quad (2.8)$$

$$\tau_{zz} = \mu\left(2w_z - \frac{2}{3}(u_x + v_y + w_z)\right) \quad (2.9)$$

$$\tau_{xy} = \mu(u_y + v_x) \quad (2.10)$$

$$\tau_{xz} = \mu(u_z + w_x) \quad (2.11)$$

$$\tau_{yz} = \mu(v_z + w_y) \quad (2.12)$$

and

$$\bar{\tau}_x = u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \quad (2.13)$$

$$\bar{\tau}_y = u\tau_{xy} + v\tau_{yy} + w\tau_{yz} \quad (2.14)$$

$$\bar{\tau}_z = u\tau_{xz} + v\tau_{yz} + w\tau_{zz} \quad (2.15)$$

In these equations μ is the laminar viscosity coefficient, and the subscript x, y, and z on the velocity terms represents the gradient of the velocity component with respect to the subscript. The heat fluxes in equation (2.6) are written in terms of the temperature gradients via:

$$q_x = -kT_x \quad (2.16)$$

$$q_y = -kT_y \quad (2.17)$$

$$q_z = -kT_z \quad (2.18)$$

where k is the thermal conductivity and T is the translational temperature. The subscripts on the temperature represent the temperature gradients. The molecular viscosity and thermal conductivity are calculated using one of several transport property curve fits available to the user. The bulk viscosity, $\lambda = -2\mu/3$ is based on Stokes

hypothesis, which is valid when Newtonian fluids are considered and bulk viscosity effects are neglected. Bulk viscosity effects should be accounted for when rotational non-equilibrium is present [46]. In addition bulk viscosity effects exhibit a significant influence on sound propagation and shock-wave structure [46].

Chapter 3

Chemistry

3.1 Introduction

From the beginning, the design of aircraft has been largely driven by the urge to fly faster and higher than ever before. In recent years, emphasis has shifted to the spectrum of flight known as hypersonics. A case in point, the design of the National Aerospace Plane (NASP) which has received a lot of attention in recent years from both the aerospace researcher and the national media.

The conventional "rule of thumb" definition of hypersonic flow is a flow where the Mach number is greater than 5. However, hypersonic flows are best characterized as the flow regime where certain physical phenomena become increasingly more important as the Mach number is increased. In some cases, the physical phenomena may become increasingly important at lower Mach numbers while other phenomena only become important at higher Mach numbers. These phenomena include, but are not limited to thin shock layers, entropy layers, viscous interaction, low density flow, and high temperature flows. In this chapter and the next, the physical phenomena associated with high temperature flows is considered.

The effects associated with high temperature flows are commonly referred to as "high-temperature effects" or "real gas effects" [47]. For an aircraft flying in the hypersonic regime, the kinetic energy of the high speed flow is dissipated within the boundary layer through the influence of friction. The viscous dissipation can create

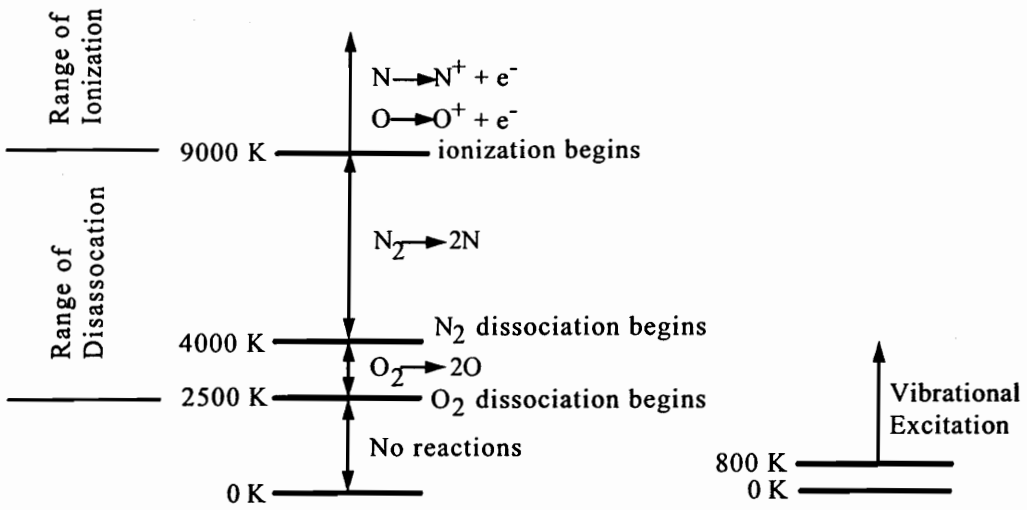


Figure 3.a: Temperature ranges for real gas effects for air at 1 atmosphere

extreme high temperatures which in turn can lead to excitation of the vibrational internal energy within the molecules, and dissociation and ionization of the molecules within the gas. Furthermore, vehicles in which ablative heat shields are employed can give rise to complex hydrocarbon reaction mechanisms. In addition to the boundary layer, the temperature on the nose region of a hypersonic vehicle can reach extraordinarily high values. Therefore, inclusion of appropriate chemical effects is vital to the calculation of these flows, where assuming a perfect gas ratio of specific heats is no longer valid.

As the gas temperature is increased, the vibrational energy of the molecules becomes "excited". From statistical mechanics, this causes the specific heats to be functions of temperature. For air, this becomes important at temperatures exceeding 800 degrees Kelvin. As the temperature is further increased, chemical reactions occur in the flow field. For air at 1 atmosphere, diatomic oxygen begins dissociation at 2000K and total dissociation occurs at 4000K [47]. Diatomic nitrogen begins dissociation at 4000 K and is typically completely dissociated at 9000K [47]. At temperatures above 9000K ionization becomes important whereby the gas becomes a partially ionized plasma. These temperature effects are illustrated in figure 3.a.

The next two chapters describe in detail the thermo-chemical model implemented

into the unstructured fluid dynamics code. The modeling of the thermodynamics and chemistry are closely linked to the formulation of the upwind solvers presented in chapter 5. The thermo-chemical modeling reviewed here closely follows the work of Grossman and Cinnella [19]. In the present chapter, the effects of non-equilibrium and equilibrium chemical reactions are investigated. We begin with a derivation of the chemical source terms, followed immediately by a discussion on the computation of the reaction rate constants.

3.2 Mass Conservation Source Terms

The source term vector was defined in the governing equations as

$$W = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_N \\ 0 \\ 0 \\ 0 \\ \dot{Q}_1 \\ \dot{Q}_2 \\ \vdots \\ \dot{Q}_M \\ 0 \end{pmatrix} \tag{3.1}$$

The first N elements of the source term vector are a result of the species continuity equations. In short, each species must obey its own conservation of mass equation. The source terms ω_i represents the rate of production of density for the ith species in the chemistry model. The Q_i terms represent the rate of change of vibrational energy for each species. These terms will be discussed in the non-equilibrium thermodynamics section of the next chapter.

In this section, we present a derivation of the source terms for the species continuity equations for a flow in chemical non-equilibrium. In the governing equations

we have a conservation of mass equation for each species. The source term for these equations represents the rate of production of the particular species with respect to time, namely $\partial\rho_i/\partial t$.

To represent the rate of production of a species, consider a general chemical reaction of the form:

$$\nu_1' X_1 + \nu_2' X_2 + \cdots + \nu_N' X_N \rightleftharpoons \nu_1'' X_1 + \nu_2'' X_2 + \cdots + \nu_N'' X_N \tag{3.2}$$

where ν' and ν'' are the stoichiometric coefficients of the reactants and products, X is the species concentration, and N is the number of species in the given chemistry model. In this discussion, a chemistry model refers to a group of species and reactions. For example, an air chemistry model may have the species N_2, O_2, NO, N, and O and a set of reactions which govern how the species react to form the other species in the model.

For a flow in chemical nonequilibrium, changes in composition take place at a definite net rate. The rate at which the reaction proceeds from left to right is called the forward reaction. The opposite, the rate at which the reaction proceeds from right to left is called the backward reaction. From empirical results, the rate of production of species i from the forward reaction is given by

$$\frac{\partial[X_i]}{\partial t} = (\nu_i'' - \nu_i')K_f \prod_{l=1}^{N} [X_l]^{\nu_l'} \tag{3.3}$$

where $[X_i]$ is the concentration of species i, and K_f is the forward reaction rate constant. In a similar fashion, the rate of production of species i from the backward reaction is given by

$$\frac{\partial[X_i]}{\partial t} = -(\nu_i'' - \nu_i')K_b \prod_{l=1}^{N} [X_l]^{\nu_l''} \tag{3.4}$$

where K_b is the backward reaction rate constant. The net rate of production of species i for a single reaction is given by the sum of the contribution from the forward reaction and the backward reaction, namely

$$\frac{\partial[X_i]}{\partial t} = (\nu_i'' - \nu_i') \left[K_f \prod_{l=1}^{N} [X_l]^{\nu_l'} - K_b \prod_{l=1}^{N} [X_l]^{\nu_l''} \right] \tag{3.5}$$

The previous discussion for a single reaction can also be applied to reaction mechanisms. The only addition is that we now sum the contribution from each reaction to obtain the overall rate of production. For a chemistry model with J reactions, the net rate of production of species i is given by

$$\frac{\partial[X_i]}{\partial t} = \sum_{j=1}^{J}(\nu''_{ji} - \nu'_{ji}) \left[K_f \prod_{l=1}^{N}[X_l]^{\nu'_{jl}} - K_b \prod_{l=1}^{N}[X_l]^{\nu''_{jl}} \right] \tag{3.6}$$

This gives the rate of production of each species in terms of the concentration. In the species continuity equation, we require the production of each species in terms of the species density. To obtain the rate of production in terms of density we apply

$$[X_i] = \frac{\rho_i}{M_i} \tag{3.7}$$

to equation (3.6) and note that the molecular weights are constant. The result of this substitution yields:

$$\omega_i = \frac{\partial \rho_i}{\partial t} = M_i \sum_{j=1}^{J}(\nu''_{ji} - \nu'_{ji}) \left[K_f \prod_{l=1}^{N} {\rho_i}^{\nu'_{jl}} - K_b \prod_{l=1}^{N} {\rho_i}^{\nu''_{jl}} \right] \tag{3.8}$$

This gives the rate of production of species i for a flow in chemical non-equilibrium. The forward and backward reaction rate constants K_f and K_b, respectively, are functions of the temperature. The computation of these rate constants will be given shortly. The backward rate constant and the forward rate constant are related via the equilibrium constant.

$$K_b = \frac{K_f}{K_e} \tag{3.9}$$

where K_e is the equilibrium constant. Rewriting equation (3.8) in terms of the equilibrium constant yields,

$$\omega_i = M_i \sum_{j=1}^{J}(\nu''_{ji} - \nu'_{ji}) K_{f,j} \left[\prod_{l=1}^{N} {\rho_i}^{\nu'_{jl}} - \frac{1}{K_{e,j}} \prod_{l=1}^{N} {\rho_i}^{\nu''_{jl}} \right] \qquad i = 1, \ldots, N \tag{3.10}$$

When equilibrium concentrations are reached, the term in brackets goes to zero since it reduces to a formulation of the Law of Mass Action,

$$K_{e,j} = \frac{\prod_{l=1}^{N} (\rho_l / M_l)^{\nu''_{jl}}}{\prod_{l=1}^{N} (\rho_l / M_l)^{\nu'_{jl}}} \tag{3.11}$$

which is valid for equilibrium concentrations.

From equation (3.10), the limiting cases of frozen and equilibrium flows should be readily apparent. When the forward rate $K_{f,j}$ goes to zero for any reaction j, the source term contribution from that reaction also goes to zero and the species mass fractions remain unchanged (if no diffusion is present). This corresponds to a frozen flow situation where the mass fractions are constant in time. At the other extreme, when the forward rate goes to infinity, the term in brackets will tend towards zero because the equilibrium concentration must be maintained. Recall from classical thermodynamics that equilibrium reactions take place at an infinite rate. In this case the source term will reach a limiting value which will balance convection, diffusion and the unsteady terms in the species continuity equation.

From a coding point of view, frozen flow may easily be implemented into the formulation by setting the forward reaction rate constant to zero or equivalently by setting the source terms to zero. Likewise, flows in chemical equilibrium may be simulated by artificially forcing the forward rate constant to infinity. From a numerical point of view, one cannot set the forward rate to infinity. It has been found through out the course of our research, that multiplying the forward rate by a large number ($1x10^6$), one can reasonably approximate equilibrium conditions.

To illustrate the application of equation (3.8) to a typical chemistry model, suppose we are given the following partial reaction mechanism, where only the reactions involving dissociation of *NO* have been provided as an example:

$$\begin{aligned}
NO + O_2 &\rightleftharpoons N + O + O_2 \\
NO + N_2 &\rightleftharpoons N + O + N_2 \\
NO + NO &\rightleftharpoons N + O + NO \\
NO + O &\rightleftharpoons N + O + O \\
NO + N &\rightleftharpoons N + O + N \\
NO + NO^+ &\rightleftharpoons N + O + NO^+ \\
NO + e^- &\rightleftharpoons N + O + e^-
\end{aligned}$$

Equation (3.8) for the production of species NO would expand out to:

$$\begin{aligned}
\frac{\omega_{NO}}{M_{NO}} &= (0-1)\left[K_{f_1}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_{O_2}}{M_{O_2}}\right) - K_{b_1}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)\left(\frac{\rho_{O_2}}{M_{O_2}}\right)\right] \\
&+ (0-1)\left[K_{f_2}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_{N_2}}{M_{N_2}}\right) - K_{b_2}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)\left(\frac{\rho_{N_2}}{M_{N_2}}\right)\right] \\
&+ (0-1)\left[K_{f_3}\left(\frac{\rho_{NO}}{M_{NO}}\right)^2 - K_{b_3}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)\left(\frac{\rho_{NO}}{M_{NO}}\right)\right] \\
&+ (0-1)\left[K_{f_4}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_O}{M_O}\right) - K_{b_4}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)^2\right] \\
&+ (0-1)\left[K_{f_5}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_N}{M_N}\right) - K_{b_5}\left(\frac{\rho_N}{M_N}\right)^2\left(\frac{\rho_{NO}}{M_{NO}}\right)\right] \\
&+ (0-1)\left[K_{f_6}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_{NO+}}{M_{NO+}}\right) - K_{b_6}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)\left(\frac{\rho_{NO+}}{M_{NO+}}\right)\right] \\
&+ (0-1)\left[K_{f_7}\left(\frac{\rho_{NO}}{M_{NO}}\right)\left(\frac{\rho_{e^-}}{M_{e^-}}\right) - K_{b_7}\left(\frac{\rho_N}{M_N}\right)\left(\frac{\rho_O}{M_O}\right)\left(\frac{\rho_{e^-}}{M_{e^-}}\right)\right]
\end{aligned}$$

From this example, it should be readily apparent that a major aspect of a nonequilibrium analysis is book keeping. Note, for a non-equilibrium reaction mechanism, the chemical reactions do not have to be independent. This is in direct contrast to an equilibrium composition calculation where N independent reactions are required. In the non-equilibrium case, the reaction mechanism typically contains a large number of elementary reactions, many of which are not independent.

3.3 Reaction Rates

In this section we provide the different methods for computing the reaction rate constants. Currently in the unstructured code there is one method of calculating the forward reaction rate and three methods of computing the equilibrium rate. The backward rate may then be obtained from the forward and equilibrium rates via

$$K_b = \frac{K_f}{K_e} \tag{3.12}$$

3.3.1 Forward Rate Calculations

For all chemistry models, the forward reaction rate is determined using an Arrhenius curve fit formula.

$$K_{f_j} = C_j T^{\eta_j} e^{-\epsilon_j/T} \tag{3.13}$$

For practically all chemistry models available in literature, the constants C_j, η_j, and ϵ_j are provided with the model or obtainable from other sources.

3.3.2 Backward Reaction Rates

There are three ways to calculate the backward (reverse) reaction rate. In all methods, the equilibrium rate is calculated first from which the backward rate constant may be obtained. The first form of calculating the equilibrium rate is an Arrhenius formula similar to that used for computing the forward rate.

$$K_{e_j} = C_j T^{\eta_j} e^{-\epsilon_j/T} \tag{3.14}$$

However, unlike the forward rate constant, the values of C_j, η_j, and ϵ_j for the backward or equilibrium rate are usually not provided in the literature. This is particularly true for the hydrogen-air chemistry models. Therefore, other options are required for computing the backward rate constant.

A second method, proposed by Park [48], relies on curve fits for the equilibrium rate constant.

$$K_{e_j} = \exp\left[A_1 + A_2 Z + A_3 Z^2 + A_4 Z^3 + A_5 Z^4\right] \tag{3.15}$$

where

$$Z = \frac{T}{10000} \tag{3.16}$$

However, this model has not been extended beyond the Air chemistry models provided by Park.

Typically, for air chemistry models, one of the previous methods will be available for the computation of the equilibrium constant. However, for hydrogen air and more esoteric chemistry models, these methods are usually not applicable. The last method is the most general and may be applied to practically all chemistry models in

the literature. This method relies on the computation of the species Gibbs free energy and the Law of Mass Action. The species Gibbs free energy is calculated using the Lewis curve fits for species thermodynamic properties [43]. The Law of Mass Action is then applied to a given reaction via

$$K_{e_j} = \left(\frac{1}{\tilde{R}T}\right)^{\delta n} \exp\left(\frac{-\delta n G}{\tilde{R}T}\right) \tag{3.17}$$

where

$$\delta n = \sum_{i=1}^{N} \nu_i \tag{3.18}$$

The Gibbs free energy is calculated from the Lewis curve fits via

$$\begin{aligned} \frac{G}{\tilde{R}T} = & - \frac{A_1}{2T} + A_2\left(\ln(T) + 1.0\right) + A_3 T\left(1.0 - \ln(T)\right) + \frac{A_4}{2}T^2 \\ & + \frac{A_5}{6}T^3 + \frac{A_6}{12}T^4 + \frac{A_7}{20}T^5 + A_9 - \frac{A_{10}}{T} \end{aligned} \tag{3.19}$$

These curve fits are available for a wide range of species. However, the curve fits are limited to a range a applicable temperatures which may make them inadequate for many of the air chemistry models.

Chapter 4

Thermodynamics

In the previous chapter, equations were derived for the modeling of the chemical source terms. In the present chapter, we are concerned with the details of the modeling of the thermodynamics for a chemical system with N species. The modeling of the thermodynamics closely follows the work of Grossman and Cinnella [19].

4.1 Equation Of State

To complete the governing equations and form a closed system of equations, the pressure p must be related to the conservative variables Q. The equation describing this relation is the equation of state.

A very important assumption is made in the development of the thermodynamics model. We assume that the particles in the flow field are *weakly interacting.* On the molecular level, aside from the fundamental attributes of mass, a molecule possesses an external force field, typically called the intermolecular force. A sketch of a intermolecular force is shown in figure 4.a. In figure 4.a, the force is sketched as a function of the distance away from the particle. At small distances, the force is strongly repulsive, tending to push molecules apart. However, as we move away from the molecule, the intermolecular force decays rapidly and becomes a weak attractive force tending to pull molecules together. By assuming weakly interacting molecules, we assume that the intermolecular potential is decaying very rapidly as we move away

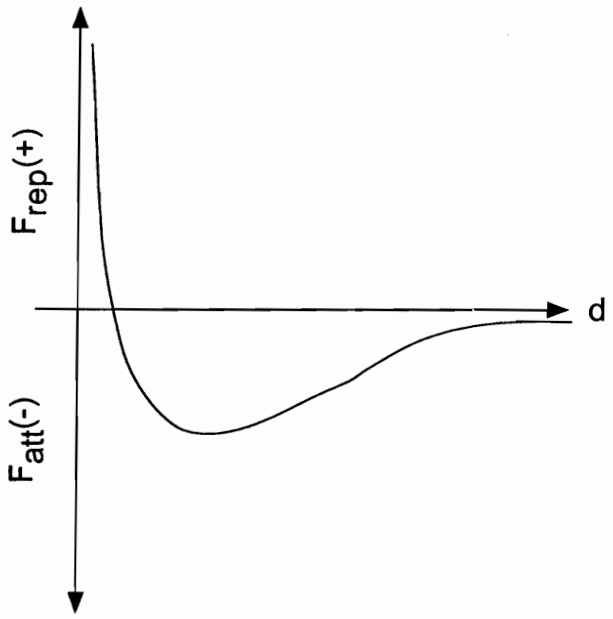


Figure 4.a: Sketch of intermolecular force as a function of distance

from the particle and is negligible when the average distance between particles is on the order of the particles radius.

For a very large percentage of the problems in aerodynamics, the assumption of weakly interacting particles is a reasonable approximation [47]. In particular, this will hold for flows of low density and moderate to high temperatures where the average distance between particles is larger than one radii.

As a consequence of the weakly interacting assumption, the species components of a mixture will behave as *thermally perfect* gases, where the species internal energy is only a function of temperature. As a direct result from the thermally perfect gases assumption, we know from classical thermodynamics that Dalton's Law of Partial pressures holds for the mixture, namely

$$p = \sum_{i=1}^{N} p_i \tag{4.1}$$

where p_i is the partial pressure of species i and the summation is carried out over all of the species present in the mixture. In addition, for a thermally perfect gas, the

state relation is expressed for species i as

$$p_i = \rho_i R_i T \tag{4.2}$$

where R_i is the gas constant for species i and T is the translational temperature. Here, we have made the simplifying assumption that the translational temperature is the same for each species, $T_i = T$. This simplification is reasonable for flows where the mass of the molecules in the mixture are on the same order of magnitude. Thus, this will not hold for flows with free electrons.

The relationship between the mixture pressure and temperature is given by

$$p = \sum_{i=1}^{N} p_i = \sum_{i=1}^{N} \rho_i R_i T = \rho \tilde{R} T \tag{4.3}$$

where the mixture density is given by

$$\rho = \sum_{i=1}^{N} \rho_i \tag{4.4}$$

and the mixture gas constant, $\tilde{R}$ is given by

$$\tilde{R} = \sum_{i=1}^{N} \frac{\rho_i}{\rho} R_i \tag{4.5}$$

Since each species behaves as a thermally perfect gas, the internal energy of the ith species is written in functional form as

$$e_i = \tilde{e}_i(T) + e_{n_i} \tag{4.6}$$

where $\tilde{e}_i$ is the known equilibrium contribution to the internal energy and e_{n_i} is the non-equilibrium contribution. The equation of state, relating the pressure to the internal energy may then be given implicitly through the temperature. For a given chemical composition, the internal energy may be obtained from

$$e = \sum_{i=1}^{N} \frac{\rho_i}{\rho} \tilde{e}_i(T) + \sum_{i=1}^{M} \frac{\rho_i}{\rho} e_{n_i} \tag{4.7}$$

Given the internal energy and non-equilibrium species internal energies, the temperature may be obtained from this equation via iterative procedures. The pressure may

then be computed from the temperature using equation (4.3). However, a simplification of the numerical scheme can be realized if we integrate the primitive variables ρ_i, u, v, w, e_{n_i}, and p in time. With the primitive variables, the iterative procedure required to find the temperature is not required. The temperature may be evaluated directly from the pressure and equation (4.3). The internal energy may then be evaluated using this temperature in equation (4.7). The implementation of time integration strategies for the primitive variables will be discussed in a later chapter. For now it is sufficient to note, that the unstructured code utilizes the primitive variables in this fashion. Therefore, we can solve directly for all of the thermodynamic properties in the system without resorting to iterative procedures.

Recall, that to this point we have not discussed how to calculate the species internal energies $\tilde{e}_i$ and e_{n_i}. The computation of the species internal energy is the subject of the next two sections. The first section details the thermal equilibrium models currently implemented. The last section derives the vibrational rate equation and source terms which form a basis for the thermal non-equilibrium model.

4.2 Equilibrium Thermodynamics

Recall from the previous section that the species internal energy is the sum of an equilibrium contribution and a non-equilibrium contribution. In the thermal equilibrium models discussed in this section, the non-equilibrium contribution is taken to be identically zero,

$$e_{n_i} = 0. \tag{4.8}$$

Therefore, the mixture internal energy is only a function of the species equilibrium internal energies, $\tilde{e}_i$,

$$e = \sum_{i=1}^{N} \frac{\rho_i}{\rho} \tilde{e}_i \tag{4.9}$$

Three methods for obtaining the species equilibrium internal energy have been implemented into the unstructured fluid dynamics code. They range from a very simple calorically perfect gas assumption to the more involved statistical mechanics model. In this section we discuss the various methods readily available.

Before we proceed to discuss the different equilibrium models, it is often convenient to express the equilibrium internal energy in terms of the specific heat at constant volume, $\tilde{c}_{v_i}$, as

$$\tilde{e}_i = \int_{Tref}^{T} \tilde{c}_{v_i}(\tau) d\tau + h_{f_i} \tag{4.10}$$

where

$$\tilde{c}_{v_i} = \frac{d\tilde{e}_i}{dT} \tag{4.11}$$

and h_f is the heat of formation.

4.2.1 Calorically Perfect Gas

For a calorically perfect gas, the specific heats are constant. The species internal energy may then be directly evaluated from equation (4.10).

$$\tilde{e}_i = \tilde{c}_{v_i} T + h_{f_i} \qquad \tilde{c}_{v_i} = \text{constant} \tag{4.12}$$

Obviously, the calorically perfect gas assumption satisfies the assumption of thermally perfect, the former assumption being more restrictive.

4.2.2 Statistical Mechanics

The partition function of statistical mechanics, has the important property of factorization. This property allows the total partition function for a molecule to be expressed as the product of individual partition functions for each type of energy. However, the internal energy of a molecule does not possess this property. In general, one cannot break the internal energy of a molecule into contributions from the different types of energy. For a gas in which the particles are weakly interacting, the internal energy may be broken down into a contribution from translation and a contribution from the internal structure of the molecule, without significant loss of accuracy. For most engineering applications, the internal structure contribution may be further broken down into contributions from rotation, vibration, and electronic excitation. This does result in a loss of accuracy due to the fact that the rotational and vibrational contributions are not entirely independent. However, for our purposes,

the dependency may be neglected. With this assumption, the equilibrium internal energy for a species may be written as the sum of the different types of energy

$$\tilde{e}_i = \tilde{e}_{tr,i} + \tilde{e}_{rot,i} + \tilde{e}_{vib,i} + \tilde{e}_{el,i} \tag{4.13}$$

For a monotomic gas, the only contribution is from the translational internal energy. The translational internal energy is the same regardless of the internal structure of the molecule and is given in statistical mechanics by

$$\tilde{e}_{tr,i} = \frac{3}{2} R_i T \qquad \tilde{c}_{v_{tr,i}} = \frac{3}{2} R_i \tag{4.14}$$

For polyatomic molecules, the contribution due to rotation (in the fully excited state) is given by

$$\tilde{e}_{rot,i} = R_i T \qquad \tilde{c}_{v_{rot,i}} = R_i \tag{4.15}$$

Notice that if we include just the translational and rotational contributions, the model reduces to the species calorically perfect gas model. In fact, the values used for $\tilde{c}_{v_i}$ in that model are given in the previous two equations. Therefore, the two models are essentially the same for monotomic species. For polyatomic molecules, the difference between the models is the vibrational contribution to the internal energy. The vibrational contribution is determined by means of a simple harmonic oscillator formula [46],

$$\tilde{e}_{vib,i} = \frac{R_i \Theta_{v,i}}{exp(\Theta_{v,i}/T) - 1} \tag{4.16}$$

where $\Theta_{v,i}$ are the characteristic temperatures of vibration. Typically more than one vibrational temperature is given for a polyatomic species. If nv vibrational temperatures are given, then nv vibrational contributions to the internal energy are computed.

$$\tilde{e}_{vib,i} = \sum_{j=1}^{nv} \frac{R_i \Theta_{v,i}}{exp(\Theta_{v,i}/T) - 1} \tag{4.17}$$

The total internal energy may then be given for any molecule, noting that $nv = 0$ for monotomic species, via

$$\tilde{e}_i = \eta_i R_i T + \sum_{j=1}^{nv} \frac{R_i \Theta_{v,i}}{exp(\Theta_{v,i}/T) - 1} \tag{4.18}$$

where $\eta = 3/2$ for monotomic species and $\eta = 5/2$ for polyatomic species. Note that electronic excitation has been neglected in this formulation which is valid for the high temperature flow of air.

4.2.3 Lewis Research Center Curve Fits

The third equilibrium thermodynamics model uses the Lewis curve fits briefly discussed in the computation of the equilibrium rate constant. There are ten curve fits for each species and temperature range. In general, there are two temperature ranges, the first range is usually 200K to 1000K. The second temperature range usually goes from 1000K to 6000K. Thus, for each species a total of twenty curve fits are supplied, ten for the lower temperature range and ten for the upper temperature range.

The specific heat at constant volume, $\tilde{c}_{v_i}$ is given by

$$\begin{aligned} \tilde{c}_{v_i} &= a_1/T^2 + a_2/T + a_3 + a_4 T + a_5 T^2 \\ &+ a_6 T^3 + a_7 T^4 + a_8 T^5 \end{aligned} \tag{4.19}$$

where $a_1 - a_8$ are the curve fits. The internal energy if found by simple integration of this formula,

$$\tilde{e}_i = \int_{T_{ref}}^{T} \tilde{c}_{v_i}(\tau)\, d\tau + h_{f_i} \tag{4.20}$$

Where the integration constant, h_{f_i}, is the heat of formation and given as a_9 in the curve fits. The tenth coefficient given in the curve fits is the integration constant for entropy.

4.3 Non-Equilibrium Thermodynamics

Up to this point we have only considered equilibrium thermodynamic models, where the non-equilibrium contribution is by definition zero. In this section, we discuss the non-equilibrium thermodynamic model where this is obviously not true. Therefore, the total internal energy of a species is given by the sum of the equilibrium and non-equilibrium contributions

$$e_i = \tilde{e}_i + e_{n_i} \tag{4.21}$$

The equilibrium contribution may be found using the previous methods described in this chapter. In this section we focus on the non-equilibrium vibrational contribution. In particular, the source terms $\dot{Q}_i$ need to be formulated from which e_{n_i} are determined

via the governing equations. In other words, the non-equilibrium contribution to the internal energy are coupled to the flow equations and are evolved in time along with our conservative variables.

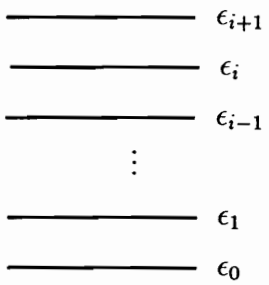
The non-equilibrium thermodynamics model presented here only includes vibrational non-equilibrium. This model is valid for situations such as the high temperature flow of air.

4.3.1 Vibrational Rate Equation

The source terms are derived via the vibrational rate equation of statistical mechanics. In this section we begin by deriving the vibrational rate equation. The derivation found here closely follows the derivation presented in Anderson [47] with some of the detail left out. For a more complete derivation see [47].

These source terms represent the time rate of change of vibrational energy. This can be further broken down into an elastic and inelastic contribution. The elastic contribution is a direct result of the vibrational rate equation. These terms represent the rate of change of vibrational energy due to molecular collision. The inelastic contribution is due to the chemical reactions that take place. The difference between the two and their significance will become clearer as we proceed in this section.

To begin our discussion of the elastic contribution to the source terms, consider a diatomic molecule with the following energy level diagram (vibrational energy only).



We wish to determine how many molecules jump from one level to another for a given time interval. Define $P_{i,i+1}$ as the probability that a molecule in the i^{th} energy level will jump to the $i + 1^{th}$ energy level upon collision with another molecule (transitions/collision). Now, define Z as the frequency of collisions (collisions/second). And finally define N_i as the population of the i^{th} level. Thus, the number of transitions

per second from the i^{th} energy level to the $i+1^{th}$ energy level is given by

$$P_{i,i+1}ZN_i = \frac{\text{transitions}}{\text{collisions}}\frac{\text{collisions}}{\text{seconds}}\text{molecules} = \frac{\text{molecules}}{\text{seconds}} \tag{4.22}$$

Therefore, to obtain the time rate of change of the population of the i^{th} level we need to add the molecules that come into the i^{th} level from the $i-1^{th}$ and $i+1^{th}$ levels and subtract molecules leaving the i^{th} level for the $i-1^{th}$ and $i+1^{th}$ levels. In terms of our variables this is represented as

$$\frac{\partial N_i}{\partial t} = P_{i-1,i}ZN_{i-1} + P_{i+1,i}ZN_{i+1} - P_{i,i-1}ZN_i - P_{i,i+1}ZN_i \tag{4.23}$$

We can define a vibrational rate constant which is a vibrational counterpart to our forward and backward reaction rate constant. If we define the vibrational rate constant as $K_{i+1,i} = P_{i+1,i}\ Z$ then our equation becomes

$$\frac{\partial N_i}{\partial t} = K_{i-1,i}N_{i-1} + K_{i+1,i}N_{i+1} - K_{i,i-1}N_i - K_{i,i+1}N_i \tag{4.24}$$

This equation is known as the "master equation for vibrational energy". Much as the reaction rates govern the concentration of a species, the vibrational rate equation governs the vibrational energy distribution of a species.

From vibrational considerations, it can be shown that

$$K_{i-1,i} = K_{i,i-1}e^{-\theta_v/T} \tag{4.25}$$

where $\theta_v = h\nu/k$. Since energy is in the form of discrete packets we can write

$$K_{i,i-1} = iK_{1,0} \tag{4.26}$$

$$K_{i+1,i} = (i+1)K_{1,0} \tag{4.27}$$

Thus, combining equations (4.25) and (4.26)-(4.27) we obtain

$$K_{i-1,i} = iK_{1,0}e^{-\theta_v/T} \tag{4.28}$$

and

$$K_{i,i+1} = (i+1)K_{1,0}e^{-\theta_v/T} \tag{4.29}$$

Now, we want to get the master equation for vibrational energy in terms of a single rate constant. For convenience the base rate constant is usually chosen ($K_{1,0}$). Substituting equation (4.25)-(4.29) into (4.24) we obtain

$$\frac{\partial N_i}{\partial t} = iK_{1,0}e^{-\theta_v/T}N_{i-1} + (i+1)K_{1,0}N_{i+1} - (i+1)K_{1,0}e^{-\theta_v/T} - iK_{1,0}N_i \quad (4.30)$$

$$= K_{1,0}\left((i+1)N_{i+1} - iN_i + e^{-\theta_v/T}(iN_{i-1} - (i+1)N_i)\right) \quad (4.31)$$

Our original objective was to obtain the time rate of change of vibrational energy for each species present in the flow. Thus we are really interested in $\partial e_n/\partial t$. From statistical mechanics we know that

$$e_n = \sum_{i=1}^{\infty} \epsilon_i N_i = \sum_{i=1}^{\infty} (ih\nu)N_i = h\nu \sum_{i=1}^{\infty} iN_i \quad (4.32)$$

The next step in obtaining our desired goal is to take the partial derivative of equation (4.32) with respect to time. This yields

$$\frac{\partial e_n}{\partial t} = \frac{\partial}{\partial t}\left(h\nu \sum_{i=1}^{\infty} iN_i\right) = h\nu \sum_{i=1}^{\infty} i\frac{\partial N_i}{\partial t} \quad (4.33)$$

Now we can substitute for $\partial N_i/\partial t$ from equation (4.32) into equation (4.33). This substitution yields

$$\frac{\partial e_n}{\partial t} = K_{1,0}h\nu \sum_{i=1}^{\infty}\left[i(i+1)N_{i+1} - i^2N_i + e^{-\theta_v/T}(i^2N_{i-1} - i(i+1)N_i)\right] \quad (4.34)$$

Now, if we make the substitution $s = i + 1$ and note that

$$\sum_{i=1}^{\infty} i(i+1)N_{i+1} - i^2N_i = \sum_{i=1}^{\infty} i^2N_{i+1} + iN_{i+1} - i^2N_i \quad (4.35)$$

$$= \sum_{s=2}^{\infty} s^2N_s - sN_s - \sum_{i=2}^{\infty} i^2N_i - N_1 \quad (4.36)$$

$$= -N_1 - \sum_{s=2}^{\infty} sN_s \quad (4.37)$$

$$= -\sum_{i=0}^{\infty} iN_i \quad (4.38)$$

which leads to

$$\frac{\partial e_n}{\partial t} = h\nu K_{1,0} \sum_{i=0}^{\infty} \left[-iN_i + e^{-\theta_v/T}(i+1)N_i \right] \tag{4.39}$$

$$= h\nu K_{1,0} \left[e^{-\theta_v/T} \sum_{i=0}^{\infty} N_i - (1 - e^{-\theta_v/T}) \sum_{i=0}^{\infty} iN_i \right] \tag{4.40}$$

$$= h\nu K_{1,0} \left[e^{-\theta_v/T} N - (1 - e^{-\theta_v/T}) \frac{e_{vib}}{h\nu} \right] \tag{4.41}$$

$$= K_{1,0}(1 - e^{-\theta_v/T}) \left[\frac{Nh\nu}{(e^{\theta_v/T} - 1)} - e_n \right] \tag{4.42}$$

From statistical mechanics we have

$$\frac{h\nu N}{(e^{\theta_v/T} - 1)} = \frac{h\nu/KT}{(e^{\theta_v/T} - 1)} \frac{N}{KT} \tag{4.43}$$

$$= \frac{\theta_v/T}{(e^{\theta_v/T} - 1)} \frac{R}{T} \tag{4.44}$$

$$= \frac{R\theta_v}{(e^{\theta_v/T} - 1)} \tag{4.45}$$

$$= \tilde{e}_{vib} \tag{4.46}$$

Substituting the last of equations (4.47) into equation (4.43) we obtain for the time rate of change of vibrational energy

$$\frac{\partial e_n}{\partial t} = K_{1,0}(1 - e^{-\theta_v/T})(\tilde{e}_{vib} - e_n) \tag{4.47}$$

This can be further simplified by noticing that the term $K_{1,0}(1 - e^{-\theta_v/T})$ has the units of 1/time. Thus it is usual to see a time constant defined as

$$\tau = \frac{1}{K_{1,0}(1 - e^{-\theta_v/T})} \tag{4.48}$$

The time constant is called the vibrational relaxation time. The introduction of the vibrational relaxation time is solely done for convenience. In terms of the vibrational relaxation time, we finally obtain

$$\frac{\partial e_n}{\partial t} = \frac{1}{\tau}(\tilde{e}_{vib} - e_n) \tag{4.49}$$

where again,

$$\tau = \frac{1}{K_{1,0}(1 - e^{-\theta_v/T})} \tag{4.50}$$

$$\tilde{e}_{vib} = \frac{\theta_v R}{(e^{\theta_v/T} - 1)} \tag{4.51}$$

Equation (4.49) gives the time rate of change of vibrational energy due to collisions between molecules. On a more fundamental level, it represents the difference between the equilibrium and the instantaneous value of vibrational internal energy. Due to molecular collisions, the excited molecules will exchange their excess vibrational energy with the translational and rotational energy of the gas, which will tend to decrease the vibrational energy to its equilibrium value.

The vibrational relaxation time is inversely proportional to the vibrational rate. However, the vibrational rate is proportional to the probability and frequency. From statistical mechanics the collision frequency (Z) is proportional to the pressure and temperature. Thus the vibrational relaxation time can be given as a function of the pressure and temperature.

In summary, we define the elastic contribution to the vibrational source terms as

$$(\dot{Q}_s)_E = \frac{\rho_s}{\tau_s}(\tilde{e}_{vib_s} - e_{n_s}), \qquad s = 1, \ldots, M \tag{4.52}$$

where E denotes the elastic contribution. The relaxation time is given by Millikan and White [49]

$$\tau_s = \frac{1.013\ 10^5}{P} exp[A_{ss}(T^{-1/3} - 0.015\mu_{ss}^{1/4}) - 18.42] \tag{4.53}$$

where

$$A_{ss} = 1.1610 \times 10^3 \mu_{ss}^{1/2} \theta_{v_s}^{4/3} \tag{4.54}$$

$$\mu_{ss} = \frac{M_s}{2} \tag{4.55}$$

The inelastic vibrational source terms are due to chemical reactions; changes in vibrational energy due to species conversion. A species that reacts to become a

different species takes the vibrational energy from the population of the reacting species to the product species. The time rate of change of vibrational energy due to chemical reactions is equal to the rate of production of species i times the vibrational energy of that species.

$$(\dot{Q}_s)_I = \frac{\partial \rho_s}{\partial t} e_{n_s} \tag{4.56}$$

$$= \omega_s e_{n_s} \tag{4.57}$$

Thus the total rate of change of vibrational energy due to collisions and reactions is given by

$$\dot{Q}_s = (\dot{Q}_s)_E + (\dot{Q}_s)_I \tag{4.58}$$

where $(\dot{Q}_s)_E$ is given by equation (4.52) and $(\dot{Q}_s)_I$ is given by equation (4.58).

Chapter 5

Inviscid Fluxes

The spatial discretization of the Euler equations were historically first derived using space-centered schemes. Centered discretization schemes produce a symmetry with respect to the Jacobian eigenvalues which do not distinguish upstream from downstream. Therefore, for hyperbolic equations, in which perturbations typically propagate along characteristics, the physical propagation is not considered in the discretization process. As a result, all second order space-centered schemes suffer from the generation of oscillations at discontinuities which have to be damped by the addition of artificial dissipation. For problems in which strong shocks are present, the spatial discretization of the inviscid fluxes require algorithms which are more physically realistic than the space centered schemes.

Upwind schemes, whose history may be traced back to Courant, Isaacson and Rees, are based on the introduction of the physical properties of the governing equations into the spatial discretization algorithm. Upwind schemes include techniques known as flux vector splitting (FVS) and flux difference splitting (FDS).

Flux vector splitting techniques, such as the Steger-Warming and Van Leer schemes, involve discretizations of the inviscid fluxes based on the sign of the Jacobian eigenvalues, whereby the inviscid flux is directionally discretized based on the associated propagation speeds.

Flux difference splitting techniques involve discretization of the inviscid fluxes based on the exact or approximate solution of the Riemann (shock-tube) problem at

each cell interface. Here, properties obtained from the solution of the Riemann problem are introduced into the discretization procedure. Algorithms based on the exact solution of the Riemann problem are known as Gudonov schemes. Schemes based on an approximate Riemann solver, such as Roe's scheme, are commonly referred to as Gudonov-type methods.

For the unstructured fluid dynamics code, the flux vector splitting methods of Steger-Warming and Van Leer, and the flux difference splitting methods of Roe have been implemented. Originally these schemes were developed for perfect gas flows and in recent years have been extended to include real gas effects. This chapter presents the upwind methods which have been extended to real gas flows. The formulation presented here closely follows the work of Grossman and Cinnella [14], [19], [22].

5.1 Properties of the Inviscid Fluxes

The inviscid fluxes of the governing equations are homogeneous functions of degree one. This implies,

$$F(Q) = \frac{\partial F}{\partial Q} Q = AQ \tag{5.1}$$

where A is the Jacobian.

5.2 Flux Vector Splitting

Two of the more popular flux vector splitting techniques are attributable to Steger & Warming and Van Leer. The basic idea of all flux vector splitting methods is to split the inviscid flux vector in one spatial direction into two parts, one containing the information that propagates downstream and one containing the information that propagates upstream. The inviscid flux is reconstructed consistent with the direction of propagation

$$F = F^+ + F^- \tag{5.2}$$

where the downstream flux, F^+, is a function of the left state, Q^-, and the upstream flux, F^-, is a function of the right state, Q^+. In functional form this is represented

as

$$F^+ = F^+(Q^-), \qquad F^- = F^-(Q^+) \tag{5.3}$$

Like the inviscid fluxes of the governing equations, the split inviscid fluxes are also homogeneous functions of degree one in the conservative variables. Therefore, the splitting may be rewritten in terms of the Jacobians via the relations:

$$A = A^+ + A^- = \frac{\partial F^+}{\partial Q} + \frac{\partial F^-}{\partial Q} \tag{5.4}$$

where the split flux is

$$F = AQ = F^+ + F^- = A^+Q + A^-Q \tag{5.5}$$

5.2.1 Steger & Warming

The Steger-Warming flux vector splitting algorithm for perfect gases is dependent on the homogeneity property of the inviscid fluxes in the Euler equations. The homogeneity property of the inviscid fluxes is carried over into the formulation for flows with thermo-chemical models.

The Steger & Warming flux vector splitting algorithm implemented into the unstructured fluid dynamics code closely follows the work of Grossman and Cinnella [14], [19], [22]. Their work is briefly reviewed here.

Steger & Warming found that the Jacobian matrices, $A = \partial f/\partial Q$, $B = \partial g/\partial Q$, and $C = \partial h/\partial Q$, may be diagonalized by mean of the right eigenvectors

$$A = S\Lambda S^{-1} \tag{5.6}$$

where S^{-1} is the inverse of the matrix of right eigenvectors S, and the diagonal matrix, Λ, contains the eigenvalues, λ_k, of A.

$$\Lambda = \begin{vmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{vmatrix} \tag{5.7}$$

The upwind method of Courant, Issacson, and Rees may then be applied to each of the characteristic speeds separately,

$$\lambda_k^{\pm} = \frac{\lambda_k \pm |\lambda_k|}{2} \tag{5.8}$$

which defines two diagonal matrices $\Lambda^{\pm}$,

$$\Lambda^{\pm} = \begin{vmatrix} \lambda_1^{\pm} & & \\ & \lambda_2^{\pm} & \\ & & \lambda_3^{\pm} \end{vmatrix} \tag{5.9}$$

where Λ^{+} has only non-negative contributions, and Λ^{-} has only non-positive contributions. Now the Jacobians may be split into two parts, one containing the information propagating downstream and one containing the information propagating upstream,

$$\Lambda = \Lambda^{+} + \Lambda^{-}, \qquad A = A^{+} + A^{-}, \qquad A^{\pm} = S\Lambda^{\pm}S^{-1} \tag{5.10}$$

The fluxes associated with the split Jacobians are obtained using the homogeneity property of the inviscid fluxes,

$$F = AQ = F^{+} + F^{-}, \qquad F^{\pm} = A^{\pm}Q. \tag{5.11}$$

The final result of the flux vector splitting may then be written for a general thermo-chemical model as:

$$F^{\pm} = \frac{\tilde{\gamma}-1}{\tilde{\gamma}}\lambda_A^{\pm}\begin{pmatrix} \rho_1/\rho \\ \rho_2/\rho \\ \vdots \\ \rho_N/\rho \\ u \\ v \\ w \\ \rho_1 e_{n_1}/\rho \\ \vdots \\ \rho_M e_{n_M}/\rho \\ h_0 - a^2/(\tilde{\gamma}-1) \end{pmatrix} + \frac{\lambda_B^{\pm}}{2\tilde{\gamma}}\begin{pmatrix} \rho_1/\rho \\ \rho_2/\rho \\ \vdots \\ \rho_N/\rho \\ u+\eta_x a \\ v+\eta_y a \\ w+\eta_z a \\ \rho_1 e_{n_1}/\rho \\ \vdots \\ \rho_M e_{n_M}/\rho \\ h_0 + \tilde{u}a \end{pmatrix} + \frac{\lambda_C^{\pm}}{2\tilde{\gamma}}\begin{pmatrix} \rho_1/\rho \\ \rho_2/\rho \\ \vdots \\ \rho_N/\rho \\ u-\eta_x a \\ v-\eta_y a \\ w-\eta_z a \\ \rho_1 e_{n_1}/\rho \\ \vdots \\ \rho_M e_{n_M}/\rho \\ h_0 - \tilde{u}a \end{pmatrix} \tag{5.12}$$

where the eigenvalues are given by

$$\lambda_A^{\pm} = \frac{\tilde{u} \pm |\tilde{u}|}{2}, \quad \lambda_B^{\pm} = \frac{(\tilde{u}+a) \pm |\tilde{u}+a|}{2}, \quad \lambda_C^{\pm} = \frac{(\tilde{u}-a) \pm |\tilde{u}-a|}{2}. \tag{5.13}$$

It should be noted that the perfect gas form can be obtained by setting the number of species (N) to one and the number of non-equilibrium thermodynamic species (M) to zero and simplifying the thermodynamic model.

5.2.2 Van Leer

The Van Leer flux vector splitting algorithm presented here closely follows the work of Grossman and Cinnella [14], [19], [22].

The split fluxes of the Steger-Warming method as defined in the previous section are not continuously differentiable at sonic and stagnation points. Van Leer developed a flux vector splitting algorithm by imposing additional conditions on the split fluxes, F^+ and F^-. In particular, the split fluxes are to be continuous functions of the Mach number. In addition, the eigenvalues of $\partial F^+/\partial Q$ must be non-negative and the eigenvalues of $\partial F^-/\partial Q$ must be non-positive with one eigenvalue equal to zero in subsonic regions ($|M| < 1$).

We can write the inviscid flux vector F in functional form as

$$F = F(\rho, a, \tilde{M}, \rho_i/\rho, e_{n_i}) \tag{5.14}$$

where $\tilde{M}$ is the normalized velocity parameter $\tilde{u}/a$ which is often referred to as a Mach number, despite that fact that it may be positive or negative. The mass flux component of F, which is $\rho\tilde{u} = \rho a\tilde{M}$ may be split as

$$\rho\tilde{u} = f_m^+ + f_m^- \tag{5.15}$$

where

$$f_m^\pm = \pm\rho a\left(\frac{\tilde{M} \pm 1}{2}\right)^2 \tag{5.16}$$

The remaining components of the inviscid flux vector F may be split in a similar

fashion to yield a Van Leer type flux vector splitting

$$\vec{F}^{\pm} = f_m^{\pm} \begin{pmatrix} \rho_1/\rho \\ \rho_2/\rho \\ \vdots \\ \rho_N/\rho \\ u + \eta_x(-\tilde{u} \pm 2a)/\tilde{\gamma} \\ v + \eta_y(-\tilde{u} \pm 2a)/\tilde{\gamma} \\ w + \eta_z(-\tilde{u} \pm 2a)/\tilde{\gamma} \\ \rho_1 e_{n_1}/\rho \\ \vdots \\ \rho_M e_{n_M}/\rho \\ h_o - (-\tilde{u} \pm a)^2/(\tilde{\gamma}+1) \end{pmatrix} \tag{5.17}$$

The other inviscid fluxes are treated in a similar fashion. Note that for $\tilde{M} \geq 1$ the flux splitting is defined as $F = F^+$, $F^- = 0$ and likewise for $\tilde{M} \leq -1$ the split flux definition is $F = F^-$, $F^+ = 0$. Like the Steger-Warming splitting, the Van Leer splitting introduced here reduces to the perfect gas formulation with simplification of the thermodynamic model and reduction of the chemistry model to a single species.

5.3 Flux Difference Splitting

Flux difference splitting algorithms or Godunov schemes take the interaction between the discretization and the physical properties one step further than the methods discussed in the previous section. In Godunov type methods information from the exact or approximate solution of the Euler equations is introduced directly into the discretization process.

In Godunov methods, the solution is considered piecewise constant over each cell at a fixed time n. The flow at the next time step, $n+1$, results from the interaction of the waves at the interface between two adjacent cells. The cell face separates two states, Q_L and Q_R and the resulting interaction can be resolved at time $n+1$ via solution of the Riemann or shock tube problem.

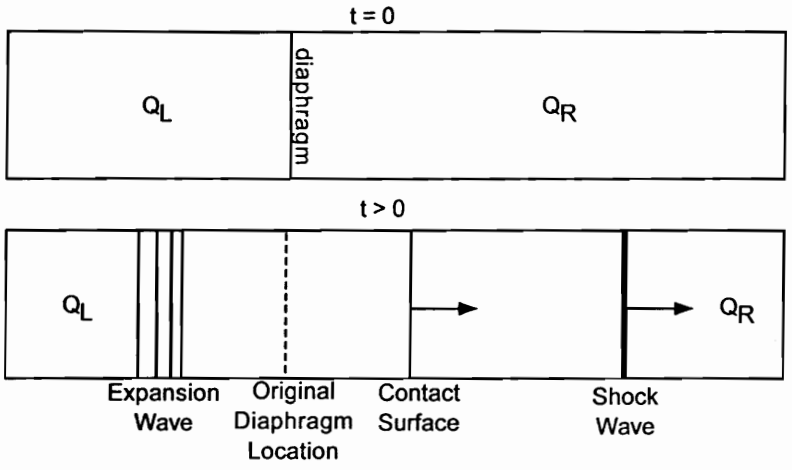


Figure 5.a: Riemann problem at time level n and $n+1$

The Riemann problem is shown in figure 5.a. In this discussion, the diaphragm represents the cell interface at time n. The Riemann problem has an exact solution, generally composed of a shock, contact surface, and expansion fan as shown in the figure. The different waves separate regions of uniform flow. As the waves propagate to the left and right of the cell interface, perturbation in the piecewise constant state are induced. The resulting state is obtained by averaging the fluid states resulting from the propagating waves.

Since the waves carry information in an upwind manner the resulting fluid state will depend only on local physical properties. Unfortunately, the exact solution of the Riemann problem requires the solution of a set of non-linear algebraic equations. Therefore, approximate solutions to the Riemann problem are used to increase the efficiency of these algorithms.

5.3.1 Roe's Approximate Riemann Solver

The essential feature of flux difference splitting techniques is the solution of the Riemann problem discussed in the previous section. The scheme developed by Roe in 1980 for perfect gases falls into this category and has been shown to produce excellent results for inviscid and viscous flows. Like the flux-vector splitting algorithms discussed earlier, Roe's flux difference splitting algorithm has been extended to flows

in chemical equilibrium and non-equilibrium (see historical perspective). The work presented here follows that of Grossman and Cinnella [19]. [14], [22].

Let the *arithmetic* average of a quantity q be defined by

$$< q >= \frac{q_l + q_r}{2}, \tag{5.18}$$

where the subscripts l and r indicate the left and right state, respectively. In addition, define the jump of a quantity as

$$[q] = q_r - q_l. \tag{5.19}$$

Then the solution to the approximate Riemann problem involves the determination of the cell interface fluxes as a summation over all of the wave speeds

$$F_{i+1/2} =< F > -\frac{1}{2}([F]_A + [F]_B + [F]_C) \tag{5.20}$$

where the absolute values of the wave speeds are substituted into the formulas for the jumps in the fluxes,

$$[F] = [F]_A + [F]_B + [F]_C \tag{5.21}$$

The contribution from the $\lambda = \hat{u}$ eigenvalue is given by:

$$[F]_A = \left([\rho] - \frac{[p]}{\hat{a}^2}\right)\hat{\tilde{u}} \begin{pmatrix} \hat{\rho}_1 \\ \hat{\rho}_2 \\ \vdots \\ \hat{\rho}_N \\ \hat{u} \\ \hat{v} \\ \hat{w} \\ \hat{e}_{n_1} \\ \vdots \\ \hat{e}_{n_M} \\ \hat{h}_0 - \hat{a}^2/(\hat{\gamma}-1) \end{pmatrix} + \hat{\rho}\hat{\tilde{u}} \begin{pmatrix} [\rho_1/\rho] \\ [\rho_2/\rho] \\ \vdots \\ [\rho_N/\rho] \\ [u] - \eta_x[\tilde{u}] \\ [v] - \eta_y[\tilde{u}] \\ [w] - \eta_z[\tilde{u}] \\ [\rho_1 e_{n_1}/\rho] \\ \vdots \\ [\rho_M e_{n_M}/\rho] \\ \Theta \end{pmatrix} \tag{5.22}$$

where

$$\Theta = \sum_{j=1}^{M}[\frac{\rho_j e_{n_j}}{\rho}] - \sum_{i=1}^{N}\psi_i[\frac{\rho_i}{\rho}] + (\hat{u}[u] + \hat{v}[v] + \hat{w}[w]) - \hat{\tilde{u}}[\tilde{u}] \tag{5.23}$$

In a similar fashion, the flux contributions from the $\lambda = \hat{u} + \hat{a}$ and $\lambda = \hat{u} - \hat{a}$ eigenvalues are given by $[F]_B$ and $[F]_C$ respectively.

$$[F]_{B,C} = \frac{1}{2\hat{a}^2}([p] \pm \hat{\rho}\hat{a}[\tilde{u}])(\hat{\tilde{u}} \pm \hat{a}) \begin{pmatrix} \hat{\rho}_1 \\ \hat{\rho}_2 \\ \vdots \\ \hat{\rho}_N \\ \hat{u} \pm \eta_x \hat{a} \\ \hat{v} \pm \eta_y \hat{a} \\ \hat{w} \pm \eta_z \hat{a} \\ \hat{e}_{n_1} \\ \vdots \\ \hat{e}_{n_M} \\ \hat{h}_0 \pm \hat{\tilde{u}}\hat{a} \end{pmatrix} \tag{5.24}$$

In the above equations, the definitions

$$\hat{\rho}_i = \left(\frac{\hat{\rho_i}}{\rho}\right), \qquad \hat{e}_{n_i} = \left(\frac{\hat{\rho_i e_{n_i}}}{\rho}\right) \tag{5.25}$$

have been used to simplify the equations. The so called Roe averages $\hat{\rho}, \hat{u}, \hat{\rho}_i, \hat{e}_n, \hat{h}_0, \hat{\psi}, \hat{a}$ used in equations (5.22)-(5.24) for this thermo-chemical model are:

$$\hat{\rho} = \sqrt{\rho_r \rho_l}, \tag{5.26}$$

$$\hat{u} = \frac{< u\sqrt{\rho} >}{< \sqrt{\rho} >}, \quad \hat{v} = \frac{< v\sqrt{\rho} >}{< \sqrt{\rho} >}, \quad \hat{w} = \frac{< w\sqrt{\rho} >}{< \sqrt{\rho} >}, \tag{5.27}$$

$$\hat{\rho}_i = \frac{< (\rho_i/\rho)\sqrt{\rho} >}{< \sqrt{\rho} >}, \tag{5.28}$$

$$\hat{e}_{n_i} = \frac{< (\rho_i e_{n_i})\sqrt{\rho} >}{< \sqrt{\rho} >}, \tag{5.29}$$

$$\hat{h}_0 = \frac{< h_0\sqrt{\rho} >}{< \sqrt{\rho} >}, \tag{5.30}$$

$$\hat{\psi}_i = \frac{R_i \hat{T}}{\hat{\gamma} - 1} - \hat{e}_i + \frac{\hat{u}^2}{2}, \tag{5.31}$$

where

$$\hat{T} = \frac{< T\sqrt{\rho} >}{< \sqrt{\rho} >}, \quad \hat{e}_i = \frac{< \tilde{e}_i\sqrt{\rho} >}{< \sqrt{\rho} >} \tag{5.32}$$

The thermodynamic properties are computed using

$$\tilde{\gamma} = 1 + \frac{\hat{R}}{\hat{c}_v^*} \tag{5.33}$$

where

$$\hat{R} = \sum_{i=1}^{N} \hat{\rho}_i R_i = \frac{< \tilde{R}\sqrt{\rho} >}{< \sqrt{\rho} >}, \quad \hat{c}_v^* = \sum_{i=1}^{N} \hat{\rho}_i c_{v_i}^* \tag{5.34}$$

,

$$c_{v_i}^* = \frac{1}{[T]} \int_{T_i}^{T_r} \tilde{c}_{v_i} dT. \tag{5.35}$$

The averaged contravariant velocity components are given by

$$\begin{pmatrix} \hat{\tilde{u}} \\ \hat{\tilde{v}} \\ \hat{\tilde{w}} \end{pmatrix} = \begin{pmatrix} \eta_x & \eta_y & \eta_z \\ \epsilon_x & \epsilon_y & \epsilon_z \\ \zeta_x & \zeta_y & \zeta_z \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{v} \\ \hat{w} \end{pmatrix} \tag{5.36}$$

and the speed of sound is given by

$$\hat{a}^2 = (\tilde{\gamma} - 1) \left[\hat{h}_o - \frac{\hat{u}^2}{2} + \hat{c}_v^* \hat{T} - \sum_{i=1}^{N} \hat{\rho}_i \hat{e}_i - \sum_{j=1}^{M} \hat{e}_{nj} \right] \tag{5.37}$$

Chapter 6

Reconstruction

The inviscid Euler flux is approximated via a numerical flux function, namely,

$$\iint_S \vec{F} \cdot \hat{n} dS \approx \iint_S F(Q_L, Q_R, \hat{n}) dS \tag{6.1}$$

where Q_L and Q_R are the descriptions of the fluid dynamic state on the left and right of the cell interface. Therefore, in order to compute the inviscid fluxes, first we need to determine the pointwise variation of the solution variables Q from the cell averages $\bar{Q}$ (which are updated in the time integration procedure). As a result, techniques need to be developed in which the spatial variation of Q may be determined in terms of the known cell averages $\bar{Q}$. This process is known as the reconstruction process. A more formal definition of the reconstruction process may be given as:

The reconstruction problem is the process of determining a pointwise distribution function given cell averaged data with the restriction that integrating the function over the domain of the cell recovers the cell average exactly.

It should be noted that in structured fluid dynamic codes, the left and right states are typically determined using the traditional $\phi - \kappa$ formulation. However, this formulation is not directly extendible to an unstructured environment. Therefore, in an unstructured code other methods for obtaining the pointwise variation of the fluid dynamic state are required.

Two distinctly different approaches for solving the reconstruction problem in an

unstructured environment have been developed to date. The first approach is a gradient based linear reconstruction originally proposed by Barth [24] and later extended by Frink [30]. Both of these methods have been implemented and are briefly reviewed here. The second approach is the *k-exact* reconstruction technique originally developed by Barth [34]. The *k-exact* method involves generating k degree polynomials in multiple dimensions for the pointwise variation of the fluid state Q. The original *k-exact* algorithm of Barth has been extended by Walters and Mitchell [33] for use with two dimensional upwind algorithms. The *k-exact* reconstruction techniques used in the unstructured flow solver closely follows the work of Walters and Mitchell with extensions to the three dimensional real gas upwind algorithms discussed in chapter 5.

6.1 K-Exact

The central theme of the *k-exact* method is to accurately approximate the pointwise spatial variation of the solution state, $Q(x, y, z)$, given the cell averaged data, $\bar{Q}$ which is defined subsequently. This is accomplished via multi-dimensional polynomials of degree k, subject to the constraint that integration of the polynomial over the cell volume recovers the cell averages exactly. From the definition it is easy to see why this is called *k-exact* reconstruction.

In the most general sense, a three-dimensional polynomial P of degree k may be written in shorthand notation as

$$P^k(x, y, z) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} C_{i,j,l} x^i y^j z^l \tag{6.2}$$

where $C_{i,j,l}$ are the coefficients of the reconstruction polynomial. These coefficients are not know prior to the solution procedure and must be determined. In three dimensions, the polynomial expression will contain $(k+1)(k+2)(k+3)/6$ unknown constants or degrees of freedom. In two dimensions, the number of unknown constants is reduced to $(k+1)(k+2)/2$.

In order to determine the coefficients we need to look closer at the constraint on the *k-exact* polynomial. In a finite volume discretization, the cell averages of the

solution state are inherently introduced into the governing equations. By definition, the cell average, or more appropriately the volume average is given by

$$\bar{Q} = \frac{1}{\Omega} \iiint_{\Omega} Q(x,y,z) d\Omega \tag{6.3}$$

Substitution of the general polynomial expression into the definition of the volume average gives us our "constraint" equations.

$$\bar{Q} = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-(i+j)} \frac{1}{\Omega} C_{i,j,l} \iiint_{\Omega} x^i y^j z^l d\Omega \tag{6.4}$$

In three dimensions, $(k+1)(k+2)(k+3)/6$ "constraint" equations are required to solve for the unknown coefficients. Likewise, in two dimensions, $(k+1)(k+2)/2$ "constraint" equations are required to solve for the undetermined coefficients. This will become clearer as we proceed with the development for a $k = 1$ reconstruction. The integral on the right side of equation (6.4) may be evaluated using Gauss quadrature rules, which are readily available for a variety of cell types.

Suppose we have N unknown coefficients in our polynomial representation. Then we need N equations to solve for these coefficients. The N equations are obtained by applying the constraint equation to N cells in the domain. These cells make up the reconstruction stencil. There are two basic families of stencils from which the user may choose. The first family is characterized by a fixed stencil which does not vary as the solution is progressed in time. The advantages of this type of stencil are ease of implementation and efficiency in that the reconstruction polynomials are computed only once. As we will see, a fixed stencil allows the reconstruction method to be simplified into one of determining constant reconstruction weights which in many cases may be determined analytically before the solution process. However, for a fixed stencil, the selection of the cells which make up the reconstruction stencil must be made on a geometric basis. That is to say that the solution itself cannot play a role in determining which cells make up the reconstruction stencil. The stencil is determined solely based on the mesh.

The other family of reconstruction stencils are characterized by a non-fixed stencil which varies as the solution is progressed. A non-fixed stencil selection is typically based on the solution, solution gradients and geometric considerations. Non-fixed

stencils have the advantage of more control over the solution process. For non-fixed stencils the need for limiters may be reduced or entirely removed. However, the polynomials cannot be written in terms of reconstruction weights or determined prior to the solution process. Therefore, the polynomial coefficients must be determined numerically every iteration or every few iterations if the stencil is frozen. Because of the desire for efficient algorithms, the fixed stencil algorithm has been implemented into the unstructured code.

For the inviscid fluxes, recall that a left and right state are required for the upwind methods discussed in chapter 5. Therefore, to apply the *k-exact* reconstruction method to an upwind solver, two separate reconstructions are required. One reconstruction for the left state and one reconstruction for the right state. These reconstructions are entirely independent. The method of choosing the proper stencil is the subject matter of the next section. For now, we assume we are given the cells which make up the stencil for both the left and right states.

For implementation into the unstructured code, second order accurate reconstruction methods were considered. Walters & Mitchell [26] showed that the *k-exact* reconstruction algorithm considered here produces a $k+1$ accurate reconstruction. Therefore, the development of the algorithm implemented into the code will be restricted to $k = 1$ polynomials. In three dimensions, our interpolating polynomial is of the form

$$Q(x,y,z) = C_{0,0,0} + C_{1,0,0}x + C_{0,1,0}y + C_{0,0,1}z \tag{6.5}$$

where $C_{0,0,0}$, $C_{1,0,0}$, $C_{0,1,0}$, and $C_{0,0,1}$ are the unknown polynomial coefficients. The four equations required to solve for the undetermined coefficients are derived from application of the constraint equation to the four cells which make up the reconstruction stencil.

$$\frac{1}{V_1}\iiint_{V_1} Q(x,y,z)dV = \frac{1}{V_1}\iiint_{V_1} C_{0,0,0}+C_{1,0,0}x+C_{0,1,0}y+C_{0,0,1}z\ dxdydz = \bar{Q}_1 \tag{6.6}$$

$$\frac{1}{V_2}\iiint_{V_2} Q(x,y,z)dV = \frac{1}{V_2}\iiint_{V_2} C_{0,0,0}+C_{1,0,0}x+C_{0,1,0}y+C_{0,0,1}z\ dxdydz = \bar{Q}_2 \tag{6.7}$$

$$\frac{1}{V_3}\iiint_{V_3} Q(x,y,z)dV = \frac{1}{V_3}\iiint_{V_3} C_{0,0,0}+C_{1,0,0}x+C_{0,1,0}y+C_{0,0,1}z\ dxdydz = \bar{Q}_3 \tag{6.8}$$

$$\frac{1}{V_4}\iiint_{V_4} Q(x,y,z)dV = \frac{1}{V_4}\iiint_{V_4} C_{0,0,0}+C_{1,0,0}x+C_{0,1,0}y+C_{0,0,1}z\ dxdydz = \bar{Q}_4 \quad (6.9)$$

where the numeric subscripts on V represent the a specific cell in the stencil. To evaluate the integral in equations (6.6)-(6.9), the mean value theorem is applied, namely

$$\frac{1}{V}\iiint_{\omega} f(x,y,z)d\omega \approx f(\bar{x},\bar{y},\bar{z}) + O(h^2) \quad (6.10)$$

where $\bar{x}$, $\bar{y}$, and $\bar{z}$ are the centroid of the cell volume V. Application of the mean value theorem to equations (6.6)-(6.9) yields a linear system for the coefficients $C_{i,j,l}$.

$$\begin{bmatrix} 1 & \bar{x}_1 & \bar{y}_1 & \bar{z}_1 \\ 1 & \bar{x}_2 & \bar{y}_2 & \bar{z}_2 \\ 1 & \bar{x}_3 & \bar{y}_3 & \bar{z}_3 \\ 1 & \bar{x}_4 & \bar{y}_4 & \bar{z}_4 \end{bmatrix} \begin{pmatrix} C_{0,0,0} \\ C_{1,0,0} \\ C_{0,1,0} \\ C_{0,0,1} \end{pmatrix} = \begin{pmatrix} \bar{Q}_1 \\ \bar{Q}_2 \\ \bar{Q}_3 \\ \bar{Q}_4 \end{pmatrix} \quad (6.11)$$

Let us denote the 4×4 matrix of the cell centroids as $[A]$ and its inverse as

$$[A]^{-1} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad (6.12)$$

The actual analytical inverse of $[A]$ is not important at this point. Solving for the polynomial coefficients via simple matrix multiplication yields

$$\begin{aligned} C_{0,0,0} &= a_{11}\bar{Q}_1 + a_{12}\bar{Q}_2 + a_{13}\bar{Q}_3 + a_{14}\bar{Q}_4 \\ C_{1,0,0} &= a_{21}\bar{Q}_1 + a_{22}\bar{Q}_2 + a_{23}\bar{Q}_3 + a_{24}\bar{Q}_4 \\ C_{0,1,0} &= a_{31}\bar{Q}_1 + a_{32}\bar{Q}_2 + a_{33}\bar{Q}_3 + a_{34}\bar{Q}_4 \\ C_{0,0,1} &= a_{41}\bar{Q}_1 + a_{42}\bar{Q}_2 + a_{43}\bar{Q}_3 + a_{44}\bar{Q}_4 \end{aligned} \quad (6.13)$$

The last step in finding the left or right state is to evaluate the polynomial expression at the flux quadrature point. For our second order reconstruction, a single flux point at the face centroid is sufficient for second order accuracy. Denoting the face centroid by (x_f, y_f, z_f) and substitution into our polynomial expression, yields

the following formula for the solution variable at the face $Q_f(x_f, y_f, z_f)$

$$\begin{aligned} Q_f(x_f, y_f, z_f) &= C_{0,0,0} + C_{1,0,0}x_f + C_{0,1,0}y_f + C_{0,0,1}z_f \\ &= a_{11}\bar{Q}_1 + a_{21}x_f\bar{Q}_1 + a_{31}y_f\bar{Q}_1 + a_{41}z_f\bar{Q}_1 \\ &+ a_{12}\bar{Q}_2 + a_{22}x_f\bar{Q}_2 + a_{32}y_f\bar{Q}_2 + a_{42}z_f\bar{Q}_2 \\ &+ a_{13}\bar{Q}_3 + a_{23}x_f\bar{Q}_3 + a_{33}y_f\bar{Q}_3 + a_{43}z_f\bar{Q}_3 \\ &+ a_{14}\bar{Q}_4 + a_{24}x_f\bar{Q}_4 + a_{34}y_f\bar{Q}_4 + a_{44}z_f\bar{Q}_4 \end{aligned} \tag{6.14}$$

Note that only $\bar{Q}_1$, $\bar{Q}_2$, $\bar{Q}_3$, and $\bar{Q}_4$ change in the solution process. Therefore instead of writing Q_f in terms of a reconstruction polynomial we may write it in terms of reconstruction weights, where the weights may be predetermined prior to the solution process. Define

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \begin{bmatrix} a_{11} + a_{21}x_f + a_{31}y_f + a_{41}z_f \\ a_{12} + a_{22}x_f + a_{32}y_f + a_{42}z_f \\ a_{13} + a_{23}x_f + a_{33}y_f + a_{43}z_f \\ a_{14} + a_{24}x_f + a_{34}y_f + a_{44}z_f \end{bmatrix} \tag{6.15}$$

Then, we may express Q_f:

$$Q_f(x_f, y_f, z_f) = [\omega_1 \quad \omega_2 \quad \omega_3 \quad \omega_4] \begin{pmatrix} \bar{Q}_1 \\ \bar{Q}_2 \\ \bar{Q}_3 \\ \bar{Q}_4 \end{pmatrix} \tag{6.16}$$

where ω are the constant reconstruction weights. The reconstruction weights may be determined analytically via

$$[\omega] = [1 \quad x_f \quad y_f \quad z_f][A]^{-1} \tag{6.17}$$

It should be pointed out that this does require inversion of a $N \times N$ matrix for each cell. For the $k = 1$ reconstructions considered here, that simply means that we have to invert a 3×3 in two dimensions and a 4×4 in three dimensions. However, extension of the algorithm to a third order accurate $k = 2$ reconstruction will require inversion of a 6×6 and 10×10 in two and three dimensions, respectively.

Once we go through the matrix algebra we arrive at simple formulas for the reconstruction weights. In two dimensions, three reconstruction weights are obtained for the $k = 1$ reconstruction. These weights are given by:

$$\omega_1 = \frac{1}{d}[(\bar{x}_2\bar{y}_3 - \bar{x}_3\bar{y}_2) + (\bar{y}_2 - \bar{y}_3)x_f + (\bar{x}_3 - \bar{x}_2)y_f] \tag{6.18}$$

$$\omega_2 = \frac{1}{d}[(\bar{x}_3\bar{y}_1 - \bar{x}_1\bar{y}_3) + (\bar{y}_3 - \bar{y}_1)x_f + (\bar{x}_1 - \bar{x}_3)y_f] \tag{6.19}$$

$$\omega_3 = \frac{1}{d}[(\bar{x}_1\bar{y}_2 - \bar{x}_2\bar{y}_1) + (\bar{y}_1 - \bar{y}_2)x_f + (\bar{x}_2 - \bar{x}_1)y_f] \tag{6.20}$$

where d is the determinant of $[A]$ and is given by,

$$d = (\bar{x}_3 - \bar{x}_2)\bar{y}_1 + (\bar{x}_1 - \bar{x}_3)\bar{y}_2 + (\bar{x}_2 - \bar{x}_1)\bar{y}_3 \tag{6.21}$$

In three dimensions, four reconstruction weights are obtained. The formulas for the four $k = 1$ reconstruction weights in three dimensions are given by:

$$\begin{aligned}
\omega_1 = \frac{1}{d} \; [\; & \bar{z}_2(\bar{x}_3\bar{y}_4 - \bar{x}_4\bar{y}_3) + \bar{z}_3(\bar{x}_4\bar{y}_2 - \bar{x}_2\bar{y}_4) + \bar{z}_4(\bar{x}_2\bar{y}_3 - \bar{x}_3\bar{y}_2) + \\
& x_f(\bar{z}_2(\bar{y}_3 - \bar{y}_4) + \bar{z}_3(\bar{y}_4 - \bar{y}_2) + \bar{z}_4(\bar{y}_2 - \bar{y}_3)) + \\
& y_f(\bar{z}_2(\bar{x}_4 - \bar{x}_3) + \bar{z}_3(\bar{x}_2 - \bar{x}_4) + \bar{z}_4(\bar{x}_3 - \bar{x}_2)) + \\
& z_f(\bar{y}_2(\bar{x}_3 - \bar{x}_4) + \bar{y}_3(\bar{x}_4 - \bar{x}_2) + \bar{y}_4(\bar{x}_2 - \bar{x}_3))] \\
\omega_2 = \frac{1}{d} \; [\; & \bar{z}_1(\bar{x}_4\bar{y}_3 - \bar{x}_3\bar{y}_4) + \bar{z}_3(\bar{x}_1\bar{y}_4 - \bar{x}_4\bar{y}_1) + \bar{z}_4(\bar{x}_3\bar{y}_1 - \bar{x}_1\bar{y}_3) + \\
& x_f(\bar{z}_1(\bar{y}_4 - \bar{y}_3) + \bar{z}_3(\bar{y}_1 - \bar{y}_4) + \bar{z}_4(\bar{y}_3 - \bar{y}_1)) + \\
& y_f(\bar{z}_1(\bar{x}_3 - \bar{x}_4) + \bar{z}_3(\bar{x}_4 - \bar{x}_1) + \bar{z}_4(\bar{x}_1 - \bar{x}_3)) + \\
& z_f(\bar{y}_1(\bar{x}_4 - \bar{x}_3) + \bar{y}_3(\bar{x}_1 - \bar{x}_4) + \bar{y}_4(\bar{x}_3 - \bar{x}_1))] \\
\omega_3 = \frac{1}{d} \; [\; & \bar{z}_1(\bar{x}_2\bar{y}_4 - \bar{x}_4\bar{y}_2) + \bar{z}_2(\bar{x}_4\bar{y}_1 - \bar{x}_1\bar{y}_4) + \bar{z}_4(\bar{x}_1\bar{y}_2 - \bar{x}_2\bar{y}_1) + \\
& x_f(\bar{z}_1(\bar{y}_2 - \bar{y}_4) + \bar{z}_2(\bar{y}_4 - \bar{y}_1) + \bar{z}_4(\bar{y}_1 - \bar{y}_2)) + \\
& y_f(\bar{z}_1(\bar{x}_4 - \bar{x}_2) + \bar{z}_2(\bar{x}_1 - \bar{x}_4) + \bar{z}_4(\bar{x}_2 - \bar{x}_1)) + \\
& z_f(\bar{y}_1(\bar{x}_2 - \bar{x}_4) + \bar{y}_2(\bar{x}_4 - \bar{x}_1) + \bar{y}_4(\bar{x}_1 - \bar{x}_2))]
\end{aligned}$$

$$\omega_4 = \frac{1}{d} \quad [\quad \bar{z}_1(\bar{x}_3\bar{y}_2 - \bar{x}_2\bar{y}_3) + \bar{z}_2(\bar{x}_1\bar{y}_3 - \bar{x}_3\bar{y}_1) + \bar{z}_3(\bar{x}_2\bar{y}_1 - \bar{x}_1\bar{y}_2) +$$
$$x_f(\bar{z}_1(\bar{y}_3 - \bar{y}_2) + \bar{z}_2(\bar{y}_1 - \bar{y}_3) + \bar{z}_3(\bar{y}_2 - \bar{y}_1)) +$$
$$y_f(\bar{z}_1(\bar{x}_2 - \bar{x}_3) + \bar{z}_2(\bar{x}_3 - \bar{x}_1) + \bar{z}_3(\bar{x}_1 - \bar{x}_2)) +$$
$$z_f(\bar{y}_1(\bar{x}_3 - \bar{x}_2) + \bar{y}_2(\bar{x}_1 - \bar{x}_3) + \bar{y}_3(\bar{x}_2 - \bar{x}_1))] \tag{6.22}$$

where the determinant of the 4×4 matrix $[A]$ is given by

$$\begin{aligned} d \;=\; & ((\bar{x}_3 - \bar{x}_4)\bar{y}_2 + (\bar{x}_4 - \bar{x}_2)\bar{y}_3 + (\bar{x}_2 - \bar{x}_3)\bar{y}_4)\bar{z}_1 + \\ & ((\bar{x}_4 - \bar{x}_3)\bar{y}_1 + (\bar{x}_1 - \bar{x}_4)\bar{y}_3 + (\bar{x}_3 - \bar{x}_1)\bar{y}_4)\bar{z}_2 + \\ & ((\bar{x}_2 - \bar{x}_4)\bar{y}_1 + (\bar{x}_4 - \bar{x}_1)\bar{y}_2 + (\bar{x}_1 - \bar{x}_2)\bar{y}_4)\bar{z}_3 + \\ & ((\bar{x}_3 - \bar{x}_2)\bar{y}_1 + (\bar{x}_1 - \bar{x}_3)\bar{y}_2 + (\bar{x}_2 - \bar{x}_1)\bar{y}_3)\bar{z}_4 \end{aligned} \tag{6.23}$$

As was pointed out, going to third and higher order reconstructions requires the inversion of large matrices. Large enough that analytic inversion is cumbersome. One option to get around this is to numerically evaluate the polynomial coefficients. From a coding standpoint this is not difficult. However, the question of choosing an adequate stencil for a $k = 2$ reconstruction makes it unwieldy from a production level point of view. Since this has been accomplished in research codes, it was determined to be of little benefit in the current research.

Before we proceed to discuss stencil selection, it should be noted that a first order reconstruction ($k = 0$) yields a single reconstruction weight in both two and three dimensions. The reconstruction weight for a first order reconstruction is simply a constant equal to one.

$$\omega_1^{k=0} = 1 \tag{6.24}$$

6.1.1 Stencil Selection

Perhaps the most important and difficult tasks associated with the *k-exact* reconstruction method is the choice of the cells which constitute the stencil. The choice of

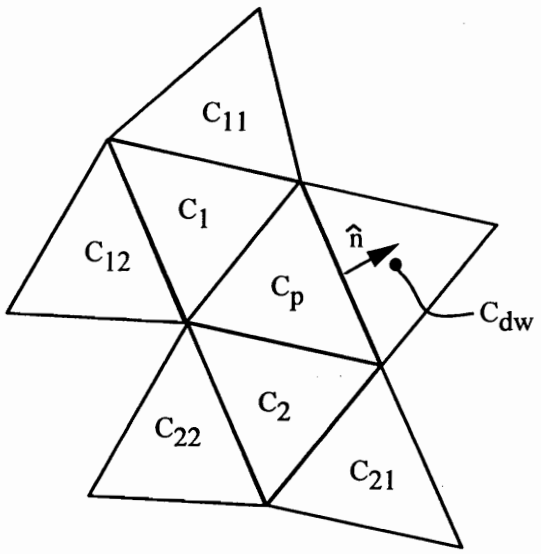


Figure 6.a: Gathering cells for upwind stencil selection

the stencil can have a drastic effect on both the solution quality and convergence to the steady state.

In the past, the *k-exact* method has been used exclusively in a two dimensional setting where things tend to behave much better than when extended to three dimensions. This effect has important consequences on the stencil selection process in three dimensions.

In this section, we describe how to choose the upwind stencil required for the inviscid fluxes. There are two primary steps in determining the best stencil. The first step we call the "gathering step". In this step we gather all of the cells which are possible choices for inclusion into our stencil. The second step is the "testing stage" where we test the possible stencils based on some pre-determined criterion.

Figure 6.a illustrates the steps followed for gathering the cells which will be considered for the stencil. In this figure, the subscript p denotes the parent cell. The parent cell is defined as the upstream cell surrounding the face for which the reconstruction is being sought. The cells with single numeric subscripts are the first dependent cells, denoted C_1 and C_2 in figure 5a. These are the cells surrounding the parent cell (not including the downstream cell C_{dw}). The cells surrounding the first dependent cells are denoted by double numeric subscripts, where the first subscript refers to the first

dependent cell. Therefore, the the cell C_{ij} is the jth cell surrounding the ith first dependent. These are the cells labeled C_{11}, C_{12}, C_{21}, and C_{22} in the figure. Care must be taken in gathering the second level of cells. Some of these cells may lay downstream of the face in question and need to be tossed out. To check to see whether a cell lies upstream of the face, a dot product of the vector formed from the face centroid to the cells centroid with the face normal $\hat{n}$ is computed. If this dot product is negative, the cell is upstream of the face, otherwise the cell is discarded from consideration. The gathering of cells is continued until the appropriate number of cells are obtained to allow a good stencil selection in the testing step. In our unstructured code, we go two levels deep as shown in the figure. The steps for gathering the cells are summarized below

Step 1: Gathering

- From the two cells surrounding the face in question, identify the upstream cell C_p.
- Gather the cells surrounding the parent cell C_p and discard the downstream cell C_{dw}.
- Gather the cells which surrounding the cells found in the second step.
- Make sure the cells are upstream of the face for which the reconstruction is being sought by computing the dot product of the vector formed from the face centroid to cell centroid with the unit normal to the face.
- Continue the process until the desired number of cells have been gathered.

After we have gathered the S cells in the first step, we must choose the N cells required to complete the stencil. This is accomplished in the testing stage. Walters and Mitchell [33] recommend using a dot product test to determine the upwind stencil. In their method, the dot products are those used to ensure the cell is upstream of the face. From the dot products, the minimum N dot products are used to determine the cells in the stencil. This test works well in two dimensions for most meshes, where it was applied by Walters and Mitchell. However, in three dimensions the test is less

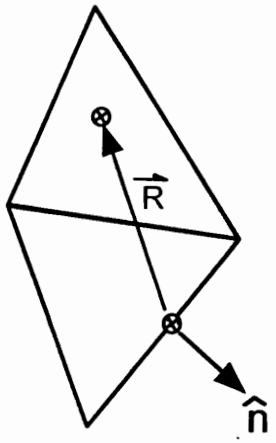


Figure 6.b: Picking cells via the dot product test

than acceptable. In fact, in three dimensions, no single test was found to produce acceptable results for a wide variety of problems. Therefore, in our unstructured fluid dynamics code, several test criterion are available from which the user may choose. Two of the more successful tests are discussed here. The first is a version of the dot product method of Walters and Mitchell. Figure 5b illustrates how this method is implemented.

In figure 6.b, the vector $\vec{R}$ is formed from the centroid of the face to the centroid of the cell for which the dot product is being computed. The vector $\hat{n}$ is the unit normal from the face for which the reconstruction is being determined. The algorithm for the Walters and Mitchell dot product test is presented below.

Stage 2: Testing - Dot product

- For each of the S cells obtained in the gathering stage, the dot product of $\vec{R}$ with $\hat{n}$ is computed. This yields S dot products.
- The absolute value of the S dot products are taken and sorted in ascending order.
- The N cells with the minimum dot product are used for the stencil.

It should be noted that this method does not insure a continuous stencil. In other words, a stencil may be formed where one or more of the cells do not share a face with any of the other cells in the stencil. As an example, the dot product

test would allow the cells C_p, C_1, and C_{21} to form a valid stencil despite the fact that cell C_{21} does not share a face with either cell C_p or C_1. All of the other tests implemented in the unstructured fluid dynamics code ensures continuous stencils. These methods are slightly more involved than the dot product test discussed above. In these methods, we generate every possible continuous stencil from the S cells obtained in the gathering step. Once that has been accomplished, we can apply any number of tests of the actual stencils instead of individual cells. Obviously, this has some benefits over methods which try to pick the stencil based on tests applied to individual cells. Since we are applying a test directly to the stencil, we have more control over the selection process.

Stage 2: Testing - Continuous stencils

- Determine all of the possible continuous stencils from the list of S cells generated in the gathering step. In figure 6.a, the valid continuous stencils are:
  - C_p, C_1, and C_2
  - C_p, C_1, and C_{11}
  - C_p, C_1, and C_{12}
  - C_p, C_2, and C_{21}
  - C_p, C_2, and C_{22}
- From the list of valid stencil, determine the best possible stencil based on a predetermined criterion.

One of the most promising test criterions is to form the matrix $[A]$ defined in equations (6.11) and (6.12). The eigenvalues of $[A]$ are then numerically computed and the condition number is used to determine the best stencil. Other tests which have been tried using the continuous stencil method include generating the reconstruction weights themselves for each possible stencil, whereby the test may be applied directly to the weights.

These methods of determining the reconstruction stencil allows the user more control over the stencil selection process. However, this added flexibility does not

come free. These methods are inherently less efficient than the simple dot product test. Fortunately, since we are using a fixed stencil which is computed only once, even a complex stencil selection process will contribute little to the computational time associated with the overall solution process.

6.2 Gradient Based Linear Reconstruction

The development of a linear reconstruction algorithm for modeling the spatial variation of the solution variables begins by expressing the pointwise variation in terms of a Taylor series. The solution vector Q may be expanded about the cell centroid $(\bar{x}, \bar{y})$ in two dimensions as

$$Q(x,y) = Q(\bar{x},\bar{y}) + \frac{\partial Q}{\partial x}(x-\bar{x}) + \frac{\partial Q}{\partial y}(y-\bar{y}) \tag{6.25}$$

which may be rewritten in terms of gradients as

$$Q(x,y) = Q(\bar{x},\bar{y}) + \triangle Q \bigtriangledown r \tag{6.26}$$

where $\triangle Q$ represents the gradient of the solution variables for the cell. Substitution of the pointwise variation into the definition of the volume average yields,

$$\bar{Q} = \frac{1}{\Omega} \iiint_{\Omega} \left[Q(\bar{x},\bar{y}) + \frac{\partial Q}{\partial x}(x-\bar{x}) + \frac{\partial Q}{\partial y}(y-\bar{y}) \right] d\Omega \tag{6.27}$$

The solution gradient may be computed via application of the divergence theorem

$$\iiint_{\Omega} \triangle Q \; d\Omega = \iint_{S} Q\hat{n} \; dS \tag{6.28}$$

For a linear reconstruction (or constant gradient), the expression reduces to

$$\triangle Q = \frac{1}{\Omega} \iint_{S} Q\hat{n} \; dS \tag{6.29}$$

The gradient is determined by computing the surface integral on the right hand side of equation (6.29) over a closed path defined by the surface area of the volume Ω. If Q varies linearly, then as long as the surface integral is computed without error, the calculation of the gradient will be exact.

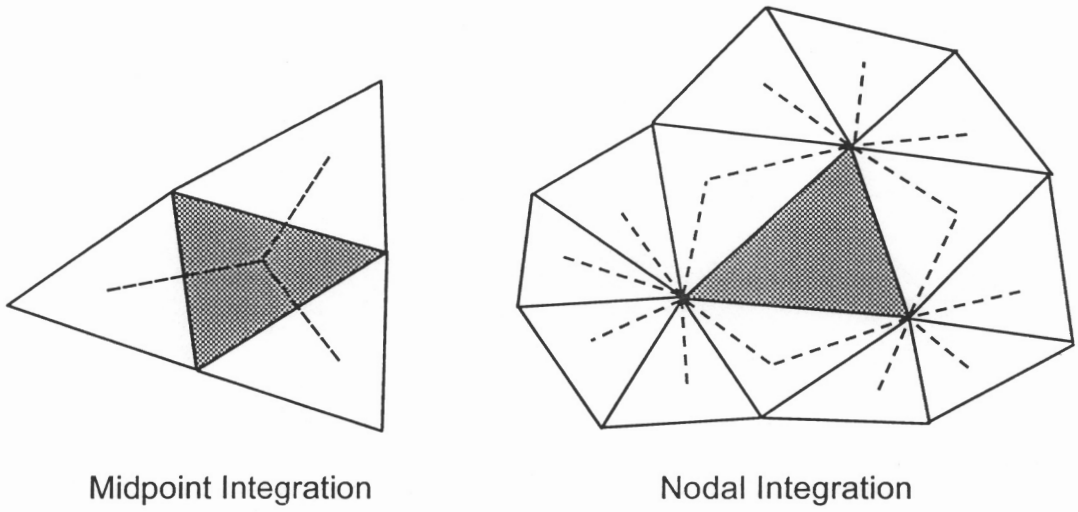


Figure 6.c: Integration methods for linear reconstruction algorithm

The simpliest surface integration path is the path defined by the actual cell control volume. This is shown in figure 6.c along with two alternatives for estimating the integral in equation (6.29). The two methods are briefly discussed below

(I) The value of Q at the midpoint of the face is determined using an arithmetic average of the cells surrounding the face. The midpoint values are determined for each face surrounding the cell. Application of a mid-point trapezoidal rule provides the gradient.

(II) The nodal values of Q are estimated by averaging the surrounding cell data to the node. The cells which contribute to the nodal values are shown in figure 6.c as dashed lines. The gradient is then computed using a trapezoidal rule. The averaging of the cell centered data to the nodes is done through weights. Frink proposed the following weights for calculating the nodal values of Q

$$Q_{node} = \frac{\sum_{i=1}^{N} Q_i/r_i}{\sum_{i=1}^{N} 1/r_i} \tag{6.30}$$

where

$$r_i = [(x_n - x_i)^2 + (y_n - y_i)^2]^{1/2} \tag{6.31}$$

Here, N is the number of cells surrounding the node, (x_n,y_n) is the coordinates of the node, and (x_i,y_i) are the coordinates of the ith cell center.

The extension of these methods to three dimensions is straight forward. In three dimensions, the truncated Taylor series becomes

$$Q(x,y,z) = Q(\bar{x},\bar{y},\bar{z}) + \frac{\partial Q}{\partial x}(x-\bar{x}) + \frac{\partial Q}{\partial y}(y-\bar{y}) + \frac{\partial Q}{\partial z}(z-\bar{z}) \tag{6.32}$$

The computation of the gradient is achieved in the same manner, where for the Frink method, the radius r_i is given by

$$r_i = [(x_n - x_i)^2 + (y_n - y_i)^2 + (z_n - z_i)^2]^{1/2} \tag{6.33}$$

As we saw with the *k-exact* reconstruction method, the linear gradient based methods can be simplified to reduce the memory and CPU overhead associated with the algorithm. Recall, from the *k-exact* discussion that only the values of the solution vector Q at the face centroid are required to evaluate the inviscid fluxes. Consider the reconstruction of Q to the centroid of a face f_i using the midpoint method outline in this section. Assuming we have obtained the average values of Q at the midpoints of all of the faces surrounding the cell, f_j, $j = 1,\ldots,N$ where N is the number of faces surrounding the cell. the algebraic equation for the value of Q at the centroid of face f_i is given by

$$Q_{\bar{f}_i} = Q_c + \frac{1}{N}\left[NQ_{fi} - \sum_{j=1}^{N} Q_{fj}\right] \tag{6.34}$$

where Q_c is the value at the cell centroid. This equation uses the midpoint formulation and may be used for any cell type. The node based algorithm can be given for the special cases of a triangular mesh and tetrahedral mesh. For a triangular mesh,

$$Q_{\bar{f}_i} = Q_c + \frac{1}{3}\left[\frac{1}{2}(Q_{n1} + Q_{n2}) - Q_{n3}\right] \tag{6.35}$$

where Q_{n3} is the node opposite to the face f_i. For tetrahedral meshes, the equation reduces to

$$Q_{\bar{f}_i} = Q_c + \frac{1}{4}\left[\frac{1}{3}(Q_{n1} + Q_{n2} + Q_{n3}) - Q_{n4}\right] \tag{6.36}$$

where Q_{n4} is the node opposite to face f_i.

The gradient based methods are attractive due to their ease of implementation. However, they suffer from the fact that an exact calculation of the solution gradient when Q varies linearly is typically not possible. In particular, the Taylor series and

trapezoidal rules used to derive the method are second order accurate, however, the averaging process will only be second order accurate in limited special cases. Therefore, the accuracy of these methods on the global level is questionable. Although these methods are based on a linear reconstruction, like the $k = 1$ reconstruction discussed earlier, in general they will not produce second order accurate reconstructions. Recall, a $k = 1$ reconstruction will produce a globally second order accurate scheme. However, the results obtained from these methods are adequate for most engineering problems and are a valuable alternative to the more involved *k-exact* method.

6.3 Limiters

The linear reconstruction methods developed in this chapter can result in new extrema in the solution. The upwind algorithms described in chapter 3 are not sufficient by themselves to avoid the generation of oscillations near discontinuities when these linear reconstruction methods are employed.

The upwind algorithms with a first order accurate reconstruction do not produce the numerical oscillations experienced with the higher order reconstruction methods. Therefore, the generation of the non-physical over and undershoots around discontinuities can be attributed to the treatment of the higher order spatial discretization.

In order to prevent the numerical oscillations that can occur in the presence of strong gradients, the reconstructed left and right states, Q_L and Q_R respectively, must be bounded such that values at the cell interface do not introduce new extrema into the solution. For our purposes, the following is applied to the reconstructed states

$$Q_{L,R} \in [\phi_L, \phi_R] \tag{6.37}$$

where ϕ_L and ϕ_R are typically taken to represent the minimum and maximum values of the cell averages used in the reconstruction process of the the left and right state. However, for the gradient based linear reconstruction methods, a stencil is not explicitly developed. Therefore, we define ϕ_L and ϕ_R as the minimum and maximum cell averages of the cells surrounding the face, as shown in figure 6.d. The number of cells from which we determine the ϕ_L and ϕ_R is a matter of user preference. To restrict

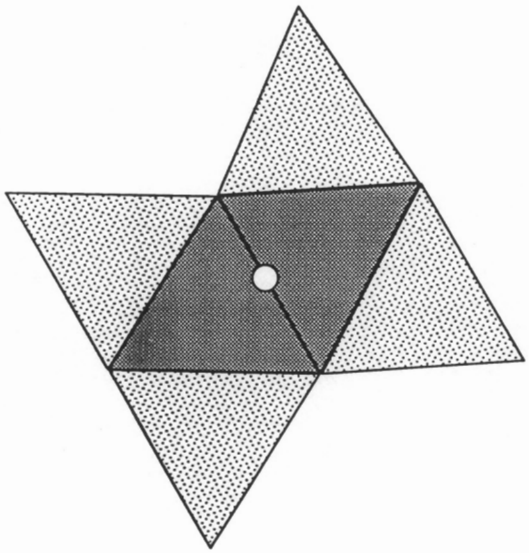


Figure 6.d: Cell selection for the inviscid limiters

the reconstruction to a first order accurate scheme in the presence of a discontinuity, the minimum and maximum of (C_1 and C_2) may be chosen as the limits. For a less restrictive scheme, the minimum and maximum search may be extended to include the cells surrounding C_1 and C_2 (shown as shaded cells in figure 6.d).

Chapter 7

Viscous Fluxes

7.1 Introduction

The viscous fluxes,

$$f_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_x - q_x \end{pmatrix}, g_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_y - q_y \end{pmatrix}, h_v = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ 0 \\ 0 \\ \vdots \\ 0 \\ \bar{\tau}_z - q_z \end{pmatrix}. \tag{7.1}$$

involve the computation of the stresses,

$$\tau_{xx} = \mu\left(2u_x - \frac{2}{3}(u_x + v_y + w_z)\right) \tag{7.2}$$

$$\tau_{yy} = \mu\left(2v_y - \frac{2}{3}(u_x + v_y + w_z)\right) \tag{7.3}$$

$$\tau_{zz} = \mu\left(2w_z - \frac{2}{3}\left(u_x + v_y + w_z\right)\right) \tag{7.4}$$

$$\tau_{xy} = \mu\left(u_y + v_x\right) \tag{7.5}$$

$$\tau_{xz} = \mu\left(u_z + w_x\right) \tag{7.6}$$

$$\tau_{yz} = \mu\left(v_z + w_y\right) \tag{7.7}$$

$$\bar{\tau}_x = u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \tag{7.8}$$

$$\bar{\tau}_y = u\tau_{xy} + v\tau_{yy} + w\tau_{yz} \tag{7.9}$$

$$\bar{\tau}_z = u\tau_{xz} + v\tau_{yz} + w\tau_{zz} \tag{7.10}$$

and heat fluxes,

$$q_x = -kT_x \tag{7.11}$$

$$q_y = -kT_y \tag{7.12}$$

$$q_z = -kT_z \tag{7.13}$$

Recall that u_x represents the gradient of the u velocity component in the x spatial coordinate. Since these represent numerical fluxes and must be computed using values at the cell interfaces, the solution variables, which are cell centered must be reconstructed to the faces. By picking suitable reconstruction algorithms, both the face values and gradients may be obtained accurately and efficiently from the same algorithm.

In the previous chapter the *k-exact* reconstruction algorithm was discussed and applied to computing the left and right state for the upwind methods discussed in chapter 2. The same reconstruction method will be used here, with some notable differences. In addition, two "finite-difference" type methods will be introduced for doing a thin-layer approximation of the viscous fluxes. The last few sections detail the computation of the laminar transport properties.

7.2 Spatial Discretization

One of the concerns in implementing any solution algorithm is the ease at which both the fluxes and Jacobians may be computed. Since two different methods for computing the viscous fluxes are introduced, it is desirable to have one flux and Jacobian routine which is independent of the reconstruction method being used. This is accomplished by introducing general interpolation weights into the formulation.

We begin the formulation with the introduction of four weights, which is the number required to complete a $k = 1$ reconstruction in three dimensions. As we will see, some of the weights may be set to 0. In terms of the four interpolation weights, any cell centered variable may be written at the face as

$$Q_f = \omega_1 \bar{Q}_1 + \omega_2 \bar{Q}_2 + \omega_3 \bar{Q}_3 + \omega_4 \bar{Q}_4 \tag{7.14}$$

where $\omega = \omega(x, y, z)$ is the reconstruction weight for a given cell in the stencil and Q is the cell centered variable for that cell. Since, in general the weights are a function of the spatial coordinates, the gradients in any of the spatial coordinates may be found by simple differentiation of the weights.

$$\begin{aligned} Q_x &= \omega_{1x} \bar{Q}_1 + \omega_{2x} \bar{Q}_2 + \omega_{3x} \bar{Q}_3 + \omega_{4x} \bar{Q}_4 \\ Q_y &= \omega_{1y} \bar{Q}_1 + \omega_{2y} \bar{Q}_2 + \omega_{3y} \bar{Q}_3 + \omega_{4y} \bar{Q}_4 \\ Q_z &= \omega_{1z} \bar{Q}_1 + \omega_{2z} \bar{Q}_2 + \omega_{3z} \bar{Q}_3 + \omega_{4z} \bar{Q}_4 \end{aligned} \tag{7.15}$$

where ω_x, ω_y, and ω_z are the derivative of the reconstruction weights with respect to the x, y, and z spatial coordinates, respectively. Three methods are available for computing the reconstruction weights, ω, and its derivatives, ω_x,ω_y,ω_z. These methods are the topic of the next two sections.

7.2.1 K-Exact Reconstruction

The *k-exact* reconstruction method for the viscous gradients is similar to the method used for determining the left and right state for the upwind methods discussed earlier

with a few important differences. First and foremost, only one stencil is required for each face in contrast to the two required for the inviscid fluxes. Second, the stencil used for the viscous fluxes is a centered stencil, which simplifies the stencil selection process. The equations derived in chapter 6 were independent of the stencil selection process and are therefore applicable here. For a $k = 1$ reconstruction in two dimensions, the weights are determined via

$$\omega_1 = \frac{1}{d}[(\bar{x}_2\bar{y}_3 - \bar{x}_3\bar{y}_2) + (\bar{y}_2 - \bar{y}_3)x_f + (\bar{x}_3 - \bar{x}_2)y_f] \tag{7.16}$$

$$\omega_2 = \frac{1}{d}[(\bar{x}_3\bar{y}_1 - \bar{x}_1\bar{y}_3) + (\bar{y}_3 - \bar{y}_1)x_f + (\bar{x}_1 - \bar{x}_3)y_f] \tag{7.17}$$

$$\omega_3 = \frac{1}{d}[(\bar{x}_1\bar{y}_2 - \bar{x}_2\bar{y}_1) + (\bar{y}_1 - \bar{y}_2)x_f + (\bar{x}_2 - \bar{x}_1)y_f] \tag{7.18}$$

where d is given by,

$$d = (\bar{x}_3 - \bar{x}_2)\bar{y}_1 + (\bar{x}_1 - \bar{x}_3)\bar{y}_2 + (\bar{x}_2 - \bar{x}_1)\bar{y}_3 \tag{7.19}$$

and in three dimensions the weights are given by,

$$\begin{aligned}
\omega_1 = \frac{1}{d} \; [\; & \bar{z}_2(\bar{x}_3\bar{y}_4 - \bar{x}_4\bar{y}_3) + \bar{z}_3(\bar{x}_4\bar{y}_2 - \bar{x}_2\bar{y}_4) + \bar{z}_4(\bar{x}_2\bar{y}_3 - \bar{x}_3\bar{y}_2) + \\
& x_f(\bar{z}_2(\bar{y}_3 - \bar{y}_4) + \bar{z}_3(\bar{y}_4 - \bar{y}_2) + \bar{z}_4(\bar{y}_2 - \bar{y}_3)) + \\
& y_f(\bar{z}_2(\bar{x}_4 - \bar{x}_3) + \bar{z}_3(\bar{x}_2 - \bar{x}_4) + \bar{z}_4(\bar{x}_3 - \bar{x}_2)) + \\
& z_f(\bar{y}_2(\bar{x}_3 - \bar{x}_4) + \bar{y}_3(\bar{x}_4 - \bar{x}_2) + \bar{y}_4(\bar{x}_2 - \bar{x}_3))] \\
\omega_2 = \frac{1}{d} \; [\; & \bar{z}_1(\bar{x}_4\bar{y}_3 - \bar{x}_3\bar{y}_4) + \bar{z}_3(\bar{x}_1\bar{y}_4 - \bar{x}_4\bar{y}_1) + \bar{z}_4(\bar{x}_3\bar{y}_1 - \bar{x}_1\bar{y}_3) + \\
& x_f(\bar{z}_1(\bar{y}_4 - \bar{y}_3) + \bar{z}_3(\bar{y}_1 - \bar{y}_4) + \bar{z}_4(\bar{y}_3 - \bar{y}_1)) + \\
& y_f(\bar{z}_1(\bar{x}_3 - \bar{x}_4) + \bar{z}_3(\bar{x}_4 - \bar{x}_1) + \bar{z}_4(\bar{x}_1 - \bar{x}_3)) + \\
& z_f(\bar{y}_1(\bar{x}_4 - \bar{x}_3) + \bar{y}_3(\bar{x}_1 - \bar{x}_4) + \bar{y}_4(\bar{x}_3 - \bar{x}_1))] \\
\omega_3 = \frac{1}{d} \; [\; & \bar{z}_1(\bar{x}_2\bar{y}_4 - \bar{x}_4\bar{y}_2) + \bar{z}_2(\bar{x}_4\bar{y}_1 - \bar{x}_1\bar{y}_4) + \bar{z}_4(\bar{x}_1\bar{y}_2 - \bar{x}_2\bar{y}_1) + \\
& x_f(\bar{z}_1(\bar{y}_2 - \bar{y}_4) + \bar{z}_2(\bar{y}_4 - \bar{y}_1) + \bar{z}_4(\bar{y}_1 - \bar{y}_2)) + \\
& y_f(\bar{z}_1(\bar{x}_4 - \bar{x}_2) + \bar{z}_2(\bar{x}_1 - \bar{x}_4) + \bar{z}_4(\bar{x}_2 - \bar{x}_1)) + \\
& z_f(\bar{y}_1(\bar{x}_2 - \bar{x}_4) + \bar{y}_2(\bar{x}_4 - \bar{x}_1) + \bar{y}_4(\bar{x}_1 - \bar{x}_2))]
\end{aligned}$$

$$\begin{aligned}\omega_4 = \frac{1}{d} \; [\; & \bar{z}_1(\bar{x}_3\bar{y}_2 - \bar{x}_2\bar{y}_3) + \bar{z}_2(\bar{x}_1\bar{y}_3 - \bar{x}_3\bar{y}_1) + \bar{z}_3(\bar{x}_2\bar{y}_1 - \bar{x}_1\bar{y}_2) + \\ & x_f(\bar{z}_1(\bar{y}_3 - \bar{y}_2) + \bar{z}_2(\bar{y}_1 - \bar{y}_3) + \bar{z}_3(\bar{y}_2 - \bar{y}_1)) + \\ & y_f(\bar{z}_1(\bar{x}_2 - \bar{x}_3) + \bar{z}_2(\bar{x}_3 - \bar{x}_1) + \bar{z}_3(\bar{x}_1 - \bar{x}_2)) + \\ & z_f(\bar{y}_1(\bar{x}_3 - \bar{x}_2) + \bar{y}_2(\bar{x}_1 - \bar{x}_3) + \bar{y}_3(\bar{x}_2 - \bar{x}_1))]\end{aligned} \tag{7.20}$$

where

$$\begin{aligned} d \; = \; & ((\bar{x}_3 - \bar{x}_4)\bar{y}_2 + (\bar{x}_4 - \bar{x}_2)\bar{y}_3 + (\bar{x}_2 - \bar{x}_3)\bar{y}_4)\bar{z}_1 + \\ & ((\bar{x}_4 - \bar{x}_3)\bar{y}_1 + (\bar{x}_1 - \bar{x}_4)\bar{y}_3 + (\bar{x}_3 - \bar{x}_1)\bar{y}_4)\bar{z}_2 + \\ & ((\bar{x}_2 - \bar{x}_4)\bar{y}_1 + (\bar{x}_4 - \bar{x}_1)\bar{y}_2 + (\bar{x}_1 - \bar{x}_2)\bar{y}_4)\bar{z}_3 + \\ & ((\bar{x}_3 - \bar{x}_2)\bar{y}_1 + (\bar{x}_1 - \bar{x}_3)\bar{y}_2 + (\bar{x}_2 - \bar{x}_1)\bar{y}_3)\bar{z}_4 \end{aligned} \tag{7.21}$$

The gradients required for the computation of the viscous fluxes are computed via the derivatives of the reconstruction weights with respect to the three spatial coordinate. In two dimensions, the derivatives with respect to x are given by

$$\omega_{1x} = \frac{1}{d}(\bar{y}_2 - \bar{y}_3) \tag{7.22}$$

$$\omega_{2x} = \frac{1}{d}(\bar{y}_3 - \bar{y}_1) \tag{7.23}$$

$$\omega_{3x} = \frac{1}{d}(\bar{y}_1 - \bar{y}_2) \tag{7.24}$$

and with respect to y,

$$\omega_{1y} = \frac{1}{d}(\bar{x}_3 - \bar{x}_2) \tag{7.25}$$

$$\omega_{2y} = \frac{1}{d}(\bar{x}_1 - \bar{x}_3) \tag{7.26}$$

$$\omega_{3y} = \frac{1}{d}(\bar{x}_2 - \bar{x}_1) \tag{7.27}$$

Likewise, the derivatives of the three dimensional $k = 1$ reconstruction weights with respect to x are given by

$$\omega_{1x} = \frac{1}{d}(\bar{z}_2(\bar{y}_3 - \bar{y}_4) + \bar{z}_3(\bar{y}_4 - \bar{y}_2) + \bar{z}_4(\bar{y}_2 - \bar{y}_3)) \tag{7.28}$$

$$\omega_{2x} = \frac{1}{d}(\bar{z}_1(\bar{y}_4 - \bar{y}_3) + \bar{z}_3(\bar{y}_1 - \bar{y}_4) + \bar{z}_4(\bar{y}_3 - \bar{y}_1)) \tag{7.29}$$

$$\omega_{3x} = \frac{1}{d}(\bar{z}_1(\bar{y}_2 - \bar{y}_4) + \bar{z}_2(\bar{y}_4 - \bar{y}_1) + \bar{z}_4(\bar{y}_1 - \bar{y}_2)) \tag{7.30}$$

$$\omega_{4x} = \frac{1}{d}(\bar{z}_1(\bar{y}_3 - \bar{y}_2) + \bar{z}_2(\bar{y}_1 - \bar{y}_3) + \bar{z}_3(\bar{y}_2 - \bar{y}_1)) \tag{7.31}$$

and the y derivatives,

$$\omega_{1y} = \frac{1}{d}(\bar{z}_2(\bar{x}_4 - \bar{x}_3) + \bar{z}_3(\bar{x}_2 - \bar{x}_4) + \bar{z}_4(\bar{x}_3 - \bar{x}_2)) \tag{7.32}$$

$$\omega_{2y} = \frac{1}{d}(\bar{z}_1(\bar{x}_3 - \bar{x}_4) + \bar{z}_3(\bar{x}_4 - \bar{x}_1) + \bar{z}_4(\bar{x}_1 - \bar{x}_3)) \tag{7.33}$$

$$\omega_{3y} = \frac{1}{d}(\bar{z}_1(\bar{x}_4 - \bar{x}_2) + \bar{z}_2(\bar{x}_1 - \bar{x}_4) + \bar{z}_4(\bar{x}_2 - \bar{x}_1)) \tag{7.34}$$

$$\omega_{4y} = \frac{1}{d}(\bar{z}_1(\bar{x}_2 - \bar{x}_3) + \bar{z}_2(\bar{x}_3 - \bar{x}_1) + \bar{z}_3(\bar{x}_1 - \bar{x}_2)) \tag{7.35}$$

and finally, the z derivatives,

$$\omega_{1z} = \frac{1}{d}(\bar{y}_2(\bar{x}_3 - \bar{x}_4) + \bar{y}_3(\bar{x}_4 - \bar{x}_2) + \bar{y}_4(\bar{x}_2 - \bar{x}_3)) \tag{7.36}$$

$$\omega_{2z} = \frac{1}{d}(\bar{y}_1(\bar{x}_4 - \bar{x}_3) + \bar{y}_3(\bar{x}_1 - \bar{x}_4) + \bar{y}_4(\bar{x}_3 - \bar{x}_1)) \tag{7.37}$$

$$\omega_{3z} = \frac{1}{d}(\bar{y}_1(\bar{x}_2 - \bar{x}_4) + \bar{y}_2(\bar{x}_4 - \bar{x}_1) + \bar{y}_4(\bar{x}_1 - \bar{x}_2)) \tag{7.38}$$

$$\omega_{4z} = \frac{1}{d}(\bar{y}_1(\bar{x}_3 - \bar{x}_2) + \bar{y}_2(\bar{x}_1 - \bar{x}_3) + \bar{y}_3(\bar{x}_2 - \bar{x}_1)) \tag{7.39}$$

Note, the reader should be careful to use the proper definition of d for the formulas presented above.

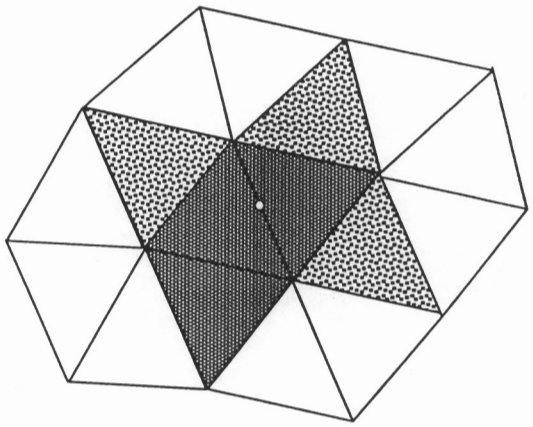


Figure 7.a: Viscous stencil selection

Stencil Selection

The primary difference between the inviscid reconstruction and the viscous reconstruction is the steps for determining the cells in the reconstruction stencil. In the inviscid stencil selection we saw that one method for picking a stencil is not sufficient for all meshes. This is particularly true in three dimensions where the stencil selection will have a dramatic effect on the solution quality and convergence to the steady state. The same effect has been seen in the viscous stencil selection to a lesser degree. Therefore, like the inviscid stencil selection, different methods for determining a stencil are available to the user.

For the viscous stencil, the two cells on either side of the face are always chosen for inclusion into the stencil. In our nomenclature, these cells would be called the parent cells. Recall, in two dimensions three cells are required to complete a $k = 1$ stencil. Therefore, in two dimensions one additional cell must be chosen to complete the stencil. In three dimensions, four cells are required to complete a $k = 1$ stencil. Therefore, two additional cells must be provided in three dimensions. Figure 7.a shows a representation of the stencil selection process for the viscous reconstruction in two dimensions. In this figure, the light shaded triangles are the cells from which we may choose. The darker cells are the cells which were chosen for this particular stencil.

As with the inviscid reconstruction, the viscous stencil selection may be broken

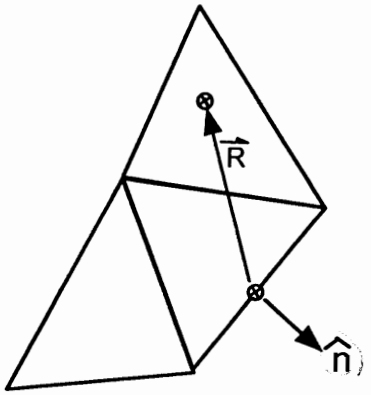


Figure 7.b: Dot product for viscous stencil selection

down into two stages. The first stage is the gather stage, where all of the cells which are to be considered for the stencil are marked. Once the cells have been gathered, the stencil is generated based on a given criterion. One method for picking a stencil which has proved successful is the dot product method described for the inviscid reconstruction. In this method, the dot product of the normal to the face with the vector formed from the centroid of the face to the centroid of the cell is computed. The cells with the smallest dot product are then used in the stencil with the two parent cells. This is shown in figure 7.b.

A slight variation on the dot product method in three dimensions is to ensure that the two additional cells are taken from opposite sides of the face. In other words, one additional cell must come from the downstream side of the face, and the other from the upstream side of the face. Obviously, exceptions would have to be made on the boundary unless ghost cells are implemented.

7.2.2 Thin-Layer Approximation

A thin-layer approximation has been implemented into the flow solver by using a simple finite-difference approximation for the gradients along with an arithmetic average for the solution variables at the cell faces.

The cell interface is represented in figure 7.c. In this figure, Q_1 and Q_2 are the cell centered solution variables and the solid line represents the cell face. Using a simple

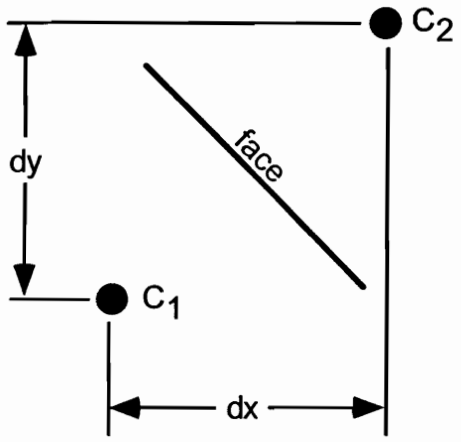


Figure 7.c: Cell interface for thin layer approximation type 1

central difference, one can represent the cell gradients via

$$\begin{aligned} Q_x &= \frac{\bar{Q}_2 - \bar{Q}_1}{\bar{x}_2 - \bar{x}_1} & \bar{x}_2 \neq \bar{x}_1 \\ &= 0 & \bar{x}_2 = \bar{x}_1 \\ Q_y &= \frac{\bar{Q}_2 - \bar{Q}_1}{\bar{y}_2 - \bar{y}_1} & \bar{y}_2 \neq \bar{y}_1 \\ &= 0 & \bar{y}_2 = \bar{y}_1 \\ Q_z &= \frac{\bar{Q}_2 - \bar{Q}_1}{\bar{z}_2 - \bar{z}_1} & \bar{z}_2 \neq \bar{z}_1 \\ &= 0 & \bar{z}_2 = \bar{z}_1 \end{aligned} \tag{7.40}$$

and the face values are obtained from a simple average

$$Q_f = \frac{Q_1 + Q_2}{2} \tag{7.41}$$

This may be rewritten in terms of the interpolation weights as

$$\omega_1 = 1/2 \qquad \omega_2 = 1/2 \tag{7.42}$$

$$\omega_{1x} = -1/dx \qquad \omega_{2x} = 1/dx \tag{7.43}$$

$$\omega_{1y} = -1/dy \qquad \omega_{2y} = 1/dy \tag{7.44}$$

$$\omega_{1z} = -1/dz \qquad \omega_{2z} = 1/dz \tag{7.45}$$

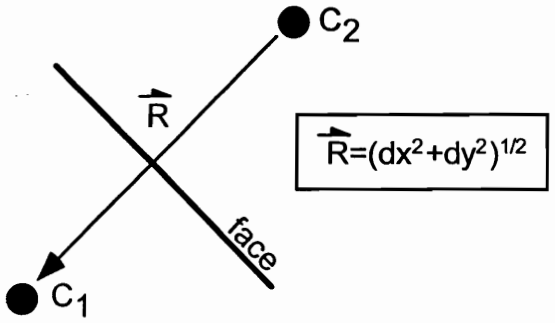


Figure 7.d: Cell interface for thin layer approximation type 2

Note, the weights in equation (7.46) are for the cases when singularities do not exist in equation (7.40).

An alternative method may be derived in which the division by zero is not a concern. This is accomplished by the introduction of a radius from one cell center to the other, as shown in figure 7.d.

The radius is given by

$$\vec{r} = dx\ \hat{i} + dy\ \hat{j} + dz\ \hat{k} \qquad r = \sqrt{dx^2 + dy^2 + dz^2} \tag{7.46}$$

where the gradient in the $\vec{r}$ direction is given by

$$Q_r = \frac{Q_2 - Q_1}{r} \tag{7.47}$$

and the components in the x, y, and z principle directions may be obtained via a chain rule

$$\begin{aligned} Q_x &= \frac{\partial Q}{\partial r}\frac{\partial r}{\partial x} = (Q_2 - Q_1)\frac{dx}{r^2} \\ Q_y &= \frac{\partial Q}{\partial r}\frac{\partial r}{\partial y} = (Q_2 - Q_1)\frac{dy}{r^2} \\ Q_z &= \frac{\partial Q}{\partial r}\frac{\partial r}{\partial z} = (Q_2 - Q_1)\frac{dz}{r^2} \end{aligned} \tag{7.48}$$

Note that when $dx = 0$, $Q_x = 0$. The weights for this method are computed as

$$\omega_1 \quad = 1/2 \qquad \omega_2 \quad = 1/2 \tag{7.49}$$

$$\omega_{1x} = -dx/r^2 \qquad \omega_{2x} = dx/r^2 \tag{7.50}$$

$$\omega_{1y} = -dy/r^2 \qquad \omega_{2y} = dy/r^2 \tag{7.51}$$

$$\omega_{1z} = -dz/r^2 \qquad \omega_{2z} = dz/r^2 \tag{7.52}$$

7.2.3 Transport Properties

Viscosity Coefficient

The laminar viscosity coefficient (μ) is computed for each species in the chemistry model using curve fits which are based on experimental data. The advantage of using curve fits is their simplicity and availability of semi-empirical rules for recovering the mixture values. Two experimental curve fits are available in the unstructured code. The first is one of the most widely adopted curve fits due to Blottner [50],

$$\mu_s = exp[(A_s \ln T + B_s) \ln T + C_s] \qquad s = 1, \ldots, N \tag{7.53}$$

where A_s, B_s, and C_s are coefficients determined from fitting the experimental data. Another popular experimental curve fit is due to Sutherland,

$$\mu_s = T^{1.5} \frac{E_s}{T + F_s}, \qquad s = 1, \ldots, N \tag{7.54}$$

where E_s and F_s are empirically derived.

The mixture viscosity coefficient is determined via Wilke's semi-empirical rule [51], which is based on kinetic theory,

$$\mu = \sum_{s=1}^{N} \frac{X_s \mu_s}{\sum_j^N X_j \phi_{sj}}, \qquad \psi_{sj} = \frac{1}{\sqrt{8}} (1 + \frac{M_s}{M_j})^{-1/2} \left[1 + \sqrt{\frac{\mu_s}{\mu_j}} \left(\frac{M_j}{M_s} \right)^{1/4} \right]^2 \tag{7.55}$$

where X_s is the mole fraction of species s.

Thermal Conductivity Coefficient

Like the species viscosity, the species thermal conductivity is computed from experimental curve fits. One such curve fit is Eucken's relation,

$$k_s = \mu_s \left(\frac{3}{2} \tilde{c}_{v_{tr,s}} + \tilde{c}_{v_s} \right), \qquad s = 1, ;N \tag{7.56}$$

where $\tilde{c}_{v_{tr,s}}$ is the translational contribution to the specific heat at constant volume. For a perfect gas, with $\tilde{\gamma} = 7/5$, Eucken's formula returns a Prandtl number $Pr = 0.737$, which is in good agreement with experimental values.

A second curve fit, due to Sutherland, has been implemented for determining the species thermal conductivity. The form of this curve fit is given by

$$k_s = T^{1.5}\frac{E_s}{T + F_s}, \qquad s = 1, \ldots, N \tag{7.57}$$

where the constants E_s and F_s are determined from experimental data.

The mixture thermal conductivity is computed using Wilke's mixture rule [51],

$$k = \sum_{s=1}^{N} \frac{X_s k_s}{\sum_j^N X_j \phi_{sj}}, \qquad \psi_{sj} = \frac{1}{\sqrt{8}}(1 + \frac{M_s}{M_j})^{-1/2}\left[1 + \sqrt{\frac{\mu_s}{\mu_j}}\left(\frac{M_j}{M_s}\right)^{1/4}\right]^2 \tag{7.58}$$

where the $\phi_{s,j}$ are the same as in the formulation for the mixture viscosity coefficient.

A simplified method for determining the mixture thermal conductivity has been implemented by assuming a constant Prandtl number and using the definition of the Prandtl number,

$$k = \frac{\mu \tilde{c}_p}{Pr}, \qquad \text{where } Pr = \text{constant} \tag{7.59}$$

In this formulation, the species thermal conductivities are not required.

Chapter 8

Turbulence Modeling

8.1 Introduction

With the growing acceptance and use of computational fluid dynamics, many new applications are conceived which require new and more sophisticated algorithms to model the various physical processes associated with current vehicle design. Since most of the flows of interest in practice are turbulent, one component required to successfully model these more complex flows is the modeling of turbulence.

The objective of all turbulence models is to specify closure conditions which relate the unknown Reynolds stresses of turbulence to the known mean flow variables via either algebraic or differential relations. In general, turbulence models are developed by first postulating a mathematical formulation containing undetermined constants and then attempting to chose those constants to match experimental data.

Turbulence models may be broken down into two distinct classes: the eddy viscosity models in which the Reynolds stresses are related to the mean rate of strain, and the Reynolds-stress models in which the complete set of Reynolds stresses are directly modeled. Currently, only the eddy viscosity models have gained wide spread acceptance and are readily available for implementation. For this reason, we will focus our attention on these models. The eddy-viscosity models may be divided into three sub-categories which are identified by the number of additional field (differential) equations required to specify the eddy-viscosity function.

Zero equation (algebraic) models are generally the most simple of all the turbulence models because they require no additional differential (field) equations and generally contain the fewest modeling coefficients. Zero equation models relate the eddy viscosity to a velocity scale determined from the mean velocity and/or gradients of the mean velocity and an algebraically determined length scale. Zero equation models suffer from the fact that they do not incorporate flow-history or stress-relaxation effects which are known to be important in modeling complex turbulent flows. More importantly from an unstructured point of view, algebraic models require the specification of an algebraic length scale which is difficult to specify for complex flows and leads to a loss of simplicity and generality for these models. In all cases, the computation of this length scale is either impossible or cumbersome in an unstructured fluid dynamics code. Therefore, these models are generally inadequate for implementation into unstructured flow solvers.

One equation models use one additional field equation to specify a velocity scale. Historically the velocity scale chosen for one-equation models is the turbulent kinetic energy (TKE). However, one equation models based on the turbulent kinetic energy still require an algebraic relation for the length scale which leads to results not very different from the algebraic models. Like the algebraic models, these models are cumbersome to implement in an unstructured format. A new group of one equation models has been developed in which the field equation represents a transport equation for the turbulence viscosity. These models have been developed with the unstructured fluid dynamics codes in mind. These new one equation models are "local" in that the equation at one point does not depend on the solution at other points and therefore are compatible with both structured and unstructured grid topologies.

Two equation models require two additional differential equations to specify the eddy-viscosity function. In general one equation is used to specify velocity scale and the second equation specifies the length scale. Since the length scale is determined from a field equation these models are attractive from a theoretical viewpoint. Many of the two equation models are "local" although some have non-local near-wall terms. From a numerical point of view, these models generally suffer from requiring a lower time step and the occurrence of numerical instabilities in the start-up phase regardless

of the time step. Other difficulties arise at sharp leading and trailing edges. Two equation models have not yet shown a decisive advantage for prediction of shock-wave boundary layer interaction or separation from smooth surfaces. In addition, they are more complex, and require more storage and CPU resources than the other eddy viscosity turbulence models.

The more recent one equation models developed for both structured and unstructured fluid dynamics codes are considered here. Two such models have become popular in the unstructured fluid dynamics community. The first model to become relatively accepted was proposed by Baldwin and Barth [40]. Their model is derived from the two-equation $k-\epsilon$ model by using some additional assumptions to arrive at a single equation for the turbulent viscosity. Here k represents the turbulent kenetic energy and ϵ is the dissipation rate.

The second and more recent one-equation model was developed by Spalart and Allmaras [41]. Unlike the Baldwin and Barth model, this model was generated from empirical data and dimensional analysis and therefore has no relation to the two-equation models. This model was chosen over the other sub-class of models for its simplicity. It was chosen over the Baldwin & Barth model primarily because it is not a simplified form of a two equation model and it is more recent and therefore has benefited from the problems discovered with the development of the Baldwin and Barth model.

It should be noted that turbulence models for complex, compressible flows have not been fully investigated. Most turbulence models have been developed for incompressible, attached boundary layers and shear flows. When these models are applied to compressible flows they yield results that vary widely in terms of their agreement with experimental data. The one-equation Spalart and Allmaras model is no different and therefore, its extension to reacting compressible flows is questionable at this time.

The remainder of this chapter discusses the turbulence model as proposed by Spalart and Allmaras and the implementation of the model into the unstructured flow solver. The full derivation of the model is beyond the scope of this discussion. The reader should see the original papers by Spalart and Allmaras for the development of the individual terms of the turbulence model [41] [42].

8.2 Transport Equation

This section details the Spalart and Allmaras one equation turbulence model implemented into the unstructured fluid dynamics code. We begin with a summary of the model as presented in the original Spalart and Allmaras paper [41]. The differential transport equation along with all relevant closure coefficients and auxiliary relations are provided. Immediately following the presentation of the differential form of the governing equation is the derivation by the present author of the integral form of the transport equation suitable for a finite volume fluid dynamics code.

8.2.1 Differential Formulation

Our discussion closely follows the appendix of the Spalart and Allmaras paper with two exceptions. The implementation followed here does not include the optional trip function required for transition. Due to the removal of this term, the boundary and initial conditions must also be slightly modified from those discussed in the appendix of the original paper.

The one-equation model provides a single differential transport equation for the turbulent quantity, $\tilde{\nu}$. The dependent variable in the transport equation is related to the eddy viscosity ν_t by:

$$\nu_t = \tilde{\nu} f_{v1}, \qquad f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}, \qquad \chi = \frac{\tilde{\nu}}{\nu} \tag{8.1}$$

where $\tilde{\nu}$ is the dependent (working) variable and ν is the molecular viscosity (μ/ρ). The differential form of the transport equation for the dependent variable is given by:

$$\frac{D\tilde{\nu}}{Dt} = C_{b1}\tilde{S}\tilde{\nu} - C_{w1} f_w \left[\frac{\tilde{\nu}}{d}\right]^2 + \frac{1}{\sigma}\left[\bigtriangledown \cdot ((\nu + \tilde{\nu}) \bigtriangledown \tilde{\nu}) + C_{b2} (\bigtriangledown \tilde{\nu})^2\right], \tag{8.2}$$

The difference between equation (8.2) and equation (A2) in [41] is the absence of the tripping functions required for modeling transition. This is equivalent to setting the closure coefficients C_{t1} and C_{t3} to zero in the equations provided in the appendix of [41].

The transport equation (8.2) represents on the left side a local time rate of change and convective acceleration via the material derivative, and on the right hand

side from left to right a production term, destruction term, and two diffusion terms, respectively.

The auxiliary relations required to complete the production term are given by:

$$\tilde{S} = S + \frac{\tilde{n}u}{\kappa^2 d^2} f_{v2} \qquad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{8.3}$$

where S is the magnitude of the vorticity and given in terms of the velocity gradients using tensor notation

$$S = \sqrt{2\Omega ij \Omega ij}, \qquad \Omega ij = \frac{1}{2}\left(\frac{\partial U_i}{\partial X_j} - \frac{\partial U_j}{\partial X_i}\right) \tag{8.4}$$

and d is the distance to the nearest solid boundary.

The auxiliary relations required to complete the destruction term are provided by:

$$f_w = g\left[\frac{1 + c_w 3^6}{g^6 + c_w 3^6}\right]^{1/6}, \quad g = r + c_w 2\left(r^6 - r\right), \quad r = \frac{\tilde{n}u}{\tilde{S}\kappa^2 d^2} \tag{8.5}$$

For large values of r, the function f_w reaches a constant. Therefore, values of r greater than 10 may be truncated.

The model is completed by the closure coefficients. The coefficients are $c_b1 = 0.1355$, $c_b2 = 0.622$, $\kappa = 0.41$, $\sigma = 2/3$, $c_w1 = c_b1/\kappa^2 + (1 + c_b2)/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2$, and $c_{v1} = 7.1$.

Turbulent heat transfer is computed using a constant turbulent Prandtl number input by the user (typically equal to 0.9). The turbulent thermal conductivity is then computed using the definition of the Prandtl number

$$k_t = \frac{\mu_t C_p}{\sigma_t} \tag{8.6}$$

where k_t is the turbulent thermal conductivity, $\mu_t = \rho\nu_t$ is the turbulent viscosity, C_p is the mixture specific heat, and σ_t is the turbulent Prandtl number.

The boundary conditions for the turbulence viscosity are simple and easily implemented. For wall bounded flows, the wall boundary condition is to set the turbulence viscosity to zero ($\tilde{\nu} = 0$). All other boundary conditions are fixed at the freestream. The initial condition is set to the freestream turbulence viscosity. The freestream turbulence viscosity is input by the user, however, a value roughly equal to 10 times the

molecular viscosity has been found to produce adequate results. It should be noted that this is different from the values proposed in the original model. The values provided in the original model assume the triping term is implemented.

8.2.2 Integral Formulation

In the previous section, the one-equation turbulence model was presented with the transport equation given in differential form. In this section the differential form is converted to integral form compatible with the finite volume flow solver. The individual steps required to arrive at the final transport equation are provided including any assumptions or simplifications.

The material derivative on the left side of equation (8.2) represents a local time rate of change and a convective acceleration.

$$\frac{D\tilde{\nu}}{Dt} = \frac{\partial\tilde{\nu}}{\partial t} + \vec{V}\cdot\nabla\tilde{\nu} \tag{8.7}$$

which may be rewritten using the chain rule and rearranging to give

$$\frac{D\tilde{\nu}}{Dt} = \frac{\partial\tilde{\nu}}{\partial t} + \nabla\cdot(\tilde{\nu}\vec{V}) - \tilde{\nu}(\nabla\cdot\vec{V}). \tag{8.8}$$

Taking the volume integral yields

$$\iiint_V \frac{D\tilde{\nu}}{Dt}\,dV = \iiint_V \frac{\partial\tilde{\nu}}{\partial t}\,dV + \iiint_V \nabla\cdot(\tilde{\nu}\vec{V})\,dV - \iiint_V \tilde{\nu}\left(\nabla\cdot\vec{V}\right)\,dV. \tag{8.9}$$

The second term on the right side is converted to a surface integral using the divergence theorem:

$$\iiint_V \frac{D\tilde{\nu}}{Dt}\,dV = \iiint_V \frac{\partial\tilde{\nu}}{\partial t}\,dV + \iint_S \tilde{\nu}\vec{V}\cdot\hat{n}\,dS - \iiint_V \tilde{\nu}\left(\nabla\cdot\vec{V}\right)\,dV. \tag{8.10}$$

The left side of the transport equation yields three terms in the integral formulation, a local time rate of change or acceleration, a convective flux which may be upwinded, and a source term. In the Spalart and Allmaras paper, they effectively neglect the contribution of the source term. For completeness the source term has been retained in the formulation, however, the source term is neglected in the unstructured fluid dynamics code.

The differential form of the diffusive flux given in equation (8.2) may be rearranged to give:

$$\frac{1}{\sigma} \nabla ((\nu + \tilde{\nu})) \nabla \tilde{\nu} + \frac{c_{b2}}{\sigma} (\nabla \tilde{\nu} \cdot \nabla \tilde{\nu}) \tag{8.11}$$

where we have replaced the last term by

$$(\nabla \tilde{\nu})^2 = \nabla \tilde{\nu} \cdot \nabla \tilde{\nu} \tag{8.12}$$

Taking the volume integral and applying the divergence theorem to the first term yields our diffusive flux:

$$\iiint_V \frac{1}{\sigma} \nabla \cdot [(\nu + \tilde{\nu}) \nabla \tilde{\nu}] \; dV = \iint_S \frac{1}{\sigma} (\nu + \tilde{\nu}) \nabla \tilde{\nu} \cdot \hat{n} \; dS \tag{8.13}$$

The second term is left as a volume integral and represents a "diffusion" source term:

$$\iiint_V \frac{c_{b2}}{\sigma} (\nabla \tilde{\nu} \cdot \nabla \tilde{\nu}) \; dV. \tag{8.14}$$

The source terms are simply the volume integral of the terms in equation (8.2). Therefore, the production source term in the integral formulation becomes

$$\iiint_V C_{b1} \tilde{S} \tilde{\nu} \; dV \tag{8.15}$$

and the integral formulation of the destruction source term becomes

$$- \iiint_V C_{w1} f_w \left[\frac{\tilde{\nu}}{d}\right]^2 \; dV \tag{8.16}$$

The integral conservation form of the turbulence model transport equation may be written as:

$$\begin{aligned} \frac{\partial}{\partial t} \quad & \iiint_V \tilde{\nu} \, dV + \iint_S \tilde{\nu} \left(\vec{V} \cdot \hat{n}\right) \, dS - \iiint_V \tilde{\nu} \nabla \cdot \vec{V} \, dV = \\ & \iiint_V \left(C_{b1} \tilde{S} \tilde{\nu} - C_{w1} f_w \left[\frac{\tilde{\nu}}{d}\right]^2 + \frac{C_{b2}}{\sigma} (\nabla \tilde{\nu} \cdot \nabla \tilde{\nu}) \right) \, dV + \\ & \iint_S \left(\frac{1}{\sigma} (\nu + \tilde{\nu}) \nabla \tilde{\nu} \cdot \hat{n} \right) \, dS \end{aligned} \tag{8.17}$$

The auxiliary functions and closure coefficients remain the same for the integral transport equation.

8.3 Implementation

The one equation Spalart & Allmaras turbulence model is implemented into the unstructured code using the integral transport equation. In this section we detail the methods used in the unstructured code to compute the different terms. As will be seen, some of the terms are computed in the same manner as the gradients in the viscous fluxes while other terms are computed in a similar fashion to the inviscid fluxes.

8.3.1 Convective Flux

The convective flux,

$$\iint_S \tilde{\nu} \left(\vec{V} \cdot \hat{n} \right) dS \tag{8.18}$$

has been implemented into the unstructured flow solver with two different methods. In both methods, regions where the Mach number is greater than unity, a full upwind biased flux is used. In regions where the Mach number is less than unity, either a full flux is used or a flux vector splitting is used depending on user preference. At this point it is too early to determine whether one method has a distinct advantage over the other.

The left and right state vectors (solution and thermodynamic property arrays) are determined using a first order reconstruction. If we denote the x,y, and z metrics by η_x, η_y, and η_z respectively, we compute the left and right Mach numbers via:

$$\bar{U}_l = \eta_x u_l + \eta_y v_l + \eta_z w_l \qquad M_l = \bar{U}_l / a_l \tag{8.19}$$

$$\bar{U}_r = \eta_x u_r + \eta_y v_r + \eta_z w_r \qquad M_r = \bar{U}_r / a_r \tag{8.20}$$

where u, v, and w are the x, y, and z components of velocity, M is the Mach number, and a the speed of sound. Based on the magnitude and sign of the Mach numbers, the method of computing the convective flux is determined.

If the left state Mach number is greater than unity ($M_l > 1$) then the flux is computed using a full upwind biased stencil,

$$F = \bar{U}_l \tilde{\nu}_l \tag{8.21}$$

and in a similar fashion, if the right state Mach number is less than -1 then the flux is computed using ,

$$F = \bar{U}_r \tilde{\nu}_r \tag{8.22}$$

Obviously, in both cases the flow is supersonic and an upwind stencil is desirable.

Regions where the flow is subsonic, the convective flux is computed using either a full flux,

$$F = \frac{1}{2}\left(\bar{U}_l \tilde{\nu}_l + \bar{U}_r \tilde{\nu}_r\right) \tag{8.23}$$

or a split flux which follows the Van Leer flux vector splitting scheme discussed in the earlier sections,

$$F = \left(\frac{1}{4}\left(M_l + 1\right)^2 a_l\right) \tilde{\nu}_l - \left(\frac{1}{4}\left(M_r - 1\right)^2 a_r\right) \tilde{\nu}_r \tag{8.24}$$

8.3.2 Diffusive Flux

The diffusive flux,

$$\iint_S \left(\frac{1}{\sigma}\left(\nu + \tilde{\nu}\right) \bigtriangledown \tilde{\nu} \cdot \hat{n}\right) \, dS \tag{8.25}$$

is computed in a similar fashion to the viscous fluxes. The same reconstruction weights that were used to compute the viscous fluxes are used to compute the turbulence model diffusive flux. Recall, that in general there are four reconstruction weights for $k = 1$ reconstruction in three dimensions. The variables are reconstructed to the face via the reconstruction weights,

$$\tilde{\nu}_{face} = \omega_1 \tilde{\nu}_1 + \omega_2 \tilde{\nu}_2 + \omega_3 \tilde{\nu}_3 + \omega_4 \tilde{\nu}_4 \tag{8.26}$$

where ω_i is the reconstruction weight for the ith cell in the reconstruction stencil and $\tilde{\nu}_i$ is the cell centered value of the reconstructed variable. The molecular viscosity is reconstructed in the same manner.

The gradients are computed using the partial derivatives of the reconstruction weights. Recall from previous discussions that for $k = 1$ and greater reconstructions, the weights are a function of spatial coordinates x, y, and z. Therefore, the gradients of the turbulence viscosity may be computed using

$$\frac{\partial \tilde{\nu}}{\partial x} = \omega_{1x} \tilde{\nu}_1 + \omega_{2x} \tilde{\nu}_2 + \omega_{3x} \tilde{\nu}_3 + \omega_{4x} \tilde{\nu}_4 \tag{8.27}$$

where ω_{ix} is the partial derivative of the ith reconstruction weight with respect to the x spatial coordinate. The gradients in the other spatial coordinate directions are obtained in the same manner. The formulation of the reconstruction weights and their derivatives for $k = 1$ were given in chapter 7.

8.3.3 Diffusion Source Term

The diffusion source term,

$$\iiint_V \frac{C_{b2}}{\sigma} (\nabla\tilde{\nu} \cdot \nabla\tilde{\nu}) \; dV \tag{8.28}$$

represents a volume integral and therefore must be computed from cell centered data. In general, the computation of source terms are fairly straightforward compared to the fluxes. However, the diffusion source term and production source term are the exceptions to the rule. The difficulty arises from the fact that reconstruction does not lend itself to computing gradients at cell centers. This can be seen from the fact that at the cell center the reconstruction weight for that cell center will reduce to one, while the other reconstruction weights will reduce to zero. That this must be true can be seen from the problem statement for reconstruction. In addition, reconstruction weights are derived for each face and backing out which reconstruction weight to use for a given cell is difficult.

To circumvent this problem, the desired gradients are computed on each face. The cell centered gradients may then be computed using an area average of the face gradients. From a coding point of view, this is easily accomplished by looping through all of the faces in the domain, adding the gradient contributions to the cells surrounding the face. This is similar to integrating the face gradients using a midpoint quadrature formulation and then dividing by the total surface area to determine the cell centered gradients

8.3.4 Production and Destruction Source Terms

The production source term,

$$\iiint_V C_{b1}\tilde{S}\tilde{\nu} \; dV \tag{8.29}$$

and the destruction source term,

$$\iiint_V C_{w1} f_w \left[\frac{\tilde{\nu}}{d}\right]^2 dV \tag{8.30}$$

are fairly straight forward in their implementation. Recall, that the vorticity is computed from the gradients of the mean velocity components at the cell center. Similar to the diffusive source term, this represents problems from a reconstruction point of view. The velocity gradients are computed in the same fashion as the gradients of the turbulent viscosity. Once the cell centered velocity gradients are determined, the vorticity is computed at the cell center via,

$$S = \sqrt{\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial z} - \frac{\partial w}{\partial y}\right)^2}. \tag{8.31}$$

The other elements of the production and destruction source terms are straightforward with the exception of the distance function, d.

8.3.5 Distance Function

From an unstructured coding point of view, the computation of the distance function represents the greatest challenge in implementing the one equation turbulence model. The distance function may be computed very accurately at the expense of CPU resources, or it may be approximated to a lower degree of accuracy with little CPU resources expended.

At this point, a simple, efficient approach for finding the distance from the cell center to the closest wall was chosen. The algorithm begins by gathering all of the faces on the boundary patches which compose our solid surface. From this list of boundary faces, a simple search is done finding the minimum distance from each face centroid to the given cell centroid. Once the closest face centroid is determined, the distance to the nodes of this face are computed and the minimum of the distances is taken to be our distance to the wall. Graphically, this is shown in figure 8.a.

In the figure, the face centroids are shown as solid circles, the nodes of the closest face as empty circle, the further distances are represented as dashed lines, and the closest distance is represented by the solid line. In this case, the face centroid labeled

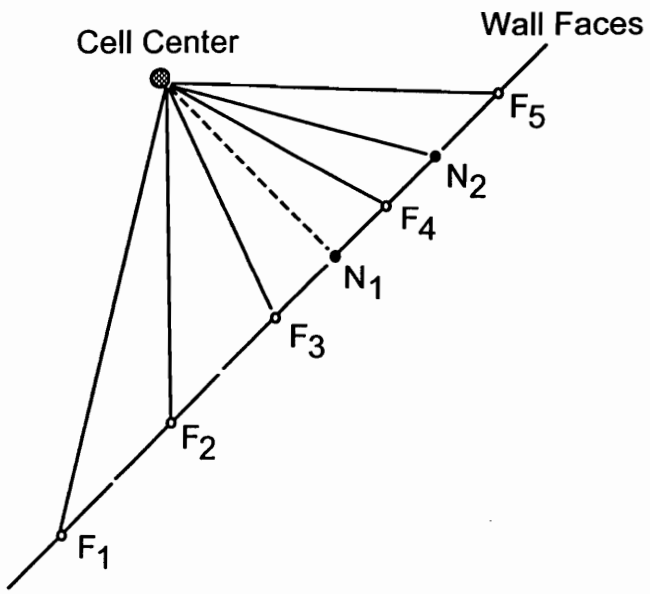


Figure 8.a: Distance function pseudo-algorithm

as F_4 is the closest face, and the node labeled N_1 is the minimum distance from the cell centroid. Therefore, in this simple algorithm, this distance would be used as the distance from the cell centroid to the wall (d).

Chapter 9

Time Integration

Two classes of problems typically arise when dealing with time integration schemes. The first class contains the unsteady problem, where accurate time integration is essential. This class of problem can be handled by the second order accurate m-stage explicit Runge Kutta time integration scheme. The second class of problem is the steady state calculation where the time path of obtaining the steady state is not important. In this case, the solution is advanced in pseudo-time until convergence to the steady state is reached. Time accuracy is not important in this case, however, efficiency is very important. Implicit techniques are used for this class of problems.

In this research, both explicit and implicit time integration schemes are employed. The explicit time integration scheme implemented in the unstructured code is an m-stage Jameson style Runge-Kutta algorithm. The implicit time integration schemes are based on the Euler implicit time integration. The resulting system of linear equations is solved iteratively via either a Jacobi relaxation or the general minimum residual (GMRES) method [45]. In this chapter, these three time integration strategies are discussed.

We begin the development of the time integration schemes by writing the governing equations in their semi-discrete form,

$$\frac{\partial Q}{\partial t}\Omega + R(Q) = 0 \tag{9.1}$$

where the residual, $R(Q)$, is defined by

$$R(Q) = \sum_{i=1}^{nf} [(F(Q_L, Q_R, \hat{n}, \zeta) - F_v(Q_f, \hat{n}, \zeta))d\Omega - W\Omega] \tag{9.2}$$

In this equation nf is the number of faces surrounding a given cell and ζ is the face metrics.

The distinction between explicit and implicit time integration occurs in the evaluation of the residual. For an explicit scheme, the residual is evaluated at the current time level, n, namely

$$\Omega \frac{\Delta Q}{\Delta t} = -R(Q^n) \tag{9.3}$$

where

$$\Delta Q = Q^{n+1} - Q^n \tag{9.4}$$

An implicit formulation begins by evaluating the residual at the next time step, $n+1$,

$$\Omega \frac{\partial Q}{\partial t} = R(Q^{n+1}) \tag{9.5}$$

which may be expanded in a Taylor series to yield a system of linear equations

$$\left[\frac{\Omega}{\Delta t} I + \frac{\partial}{\partial Q} R(Q^n)\right] \Delta Q = -R(Q^n) \tag{9.6}$$

In discussing the thermodynamics it was mentioned that time evolution of the primitive variables simplifies the computation of the temperature and therefore application of the equation of state is simplified. Another benefit of working with primitive variables is that the linearization of the residual in equation (9.6) is simplified, thus reducing the CPU resources required for the computations. The integration of the primitive variables is accomplished by the transformation of the conservative variables to the primitive variables via the chain rule, namely,

$$\frac{\partial Q}{\partial t} = \frac{\partial Q}{\partial q} \frac{\partial q}{\partial t} = M \frac{\partial q}{\partial t} \tag{9.7}$$

where

$$q = (\rho_i, u, v, w, e_{n_i}, p)^T \tag{9.8}$$

The explicit time integration scheme may then be written in terms of the primitive variables

$$\Omega \frac{\Delta q}{\Delta t} = -M^{-1} R(q^n) \tag{9.9}$$

and, in a similar fashion, the implicit time integration scheme becomes

$$\left[\frac{\Omega}{\Delta t} M + \frac{\partial}{\partial q} R(q^n) \right] \Delta q = -R(q^n) \tag{9.10}$$

This equation represents a linear system of equations which may be solved using direct methods. However, the computational and memory requirements for such a calculation are excessive. Therefore, iterative methods are used to approximately solve for Δq. These methods are typically called inner-iterative procedures since an iterative procedure is introduced inside the time integration scheme. Two methods for solving the system of equations defined by equation (9.10) are given in this chapter. The first is a simple block Jacobi technique. The second method relies on the GMRES method.

9.1 Runge Kutta

The explicit time integration scheme implemented into the unstructured code is the Jameson style, m-stage Runge Kutta algorithm. For convenience, define

$$-M^{-1} R(q^n) = -\hat{R}(q^n) \tag{9.11}$$

where M^{-1} has been included in the residual term. The solution is advanced in time according to the following:

$$\begin{aligned}
q^0 &= q^n, \\
q^1 &= q^0 - \alpha_1 \frac{\Delta t}{\Omega} \hat{R}(q^0) \\
q^2 &= q^0 - \alpha_2 \frac{\Delta t}{\Omega} \hat{R}(q^1) \\
&\vdots \\
q^m &= q^0 - \alpha_m \frac{\Delta t}{\Omega} \hat{R}(q^{m-1}) \\
q^{n+1} &= q^m.
\end{aligned} \tag{9.12}$$

where the weights, α_i, are computed via

$$\alpha_i = \frac{1}{m - i + 1} \tag{9.13}$$

9.2 Euler Implicit W/Jacobi Inner Iteration

The implicit scheme discussed in the introduction to this chapter resulted in a linear system of equations defined by

$$\left[\frac{\Omega}{\Delta t} M + \frac{\partial}{\partial q} R(q^n)\right] \Delta q = -R(q^n) \tag{9.14}$$

which is equivalent to the linear system

$$Ax = b \tag{9.15}$$

where A is the term in brackets, $x = \Delta q$, and $b = -R(q^n)$. The matrix A may be split into a diagonal and off-diagonal parts. Let D be the block diagonal matrix whose diagonal is the same as A, L be the lower triangular part of A, and U the upper triangular part of A. With this notation, the matrix A in equation (9.15) becomes

$$A = D + L + U \qquad (D + L + U)x = b \tag{9.16}$$

The solution, x, may then be found using a Jacobi type iterative technique given by

$$x^{i+1} = D^{-1}(b - (L + U)x^i) \tag{9.17}$$

The procedure is continued until the L_2 norm of the solution vector x reaches a user defined limit or until a maximum number of iterations has been completed.

9.3 Euler Implicit W/Generalized Minimum Residual Method

In the previous section, Jacobi relaxation was applied to solve the linear system of equations that arises from the linearization of the discrete governing equations.

Relaxation algorithms are very efficient in damping high frequencing waves, however, as the residual of the linear system is reduced we are left with the low frequency waves. Unfortunately, convergence suffers dramatically for relaxation methods at this point. For stiff problems where accurate solution of the linear problem is vital, relaxation methods are inadequate.

As an alternative to Jacobi relaxation, the Generalized Minimum Residual (GMRES) method may be applied to the linear system. The GMRES method, originally proposed by Saad and Schultz [45], is typically called a conjugate gradient like algorithm due to its roots in the conjugate gradient methods. A detailed derivation of the method can be found in [45]. Here we are interested in its application to the linear system defined by equation (9.10).

For clarity, let us represent this equation as

$$Ax = b \tag{9.18}$$

The residual vector of the linear system, r_i, is defined by

$$r_i = b - Ax_i \tag{9.19}$$

where x_i is the current approximate solution. This should not be confused with the residual of the governing equations. Obviously, from equation (9.19), as the residual goes to zero the closer the approximation represents the exact solution of (9.18). If $r_i = 0$, then x_i is a solution to the linear system.

The GMRES method computes the approximate solution which minimizes the l_2-norm of the residual over the Krylov subspace defined by

$$K_k = span(\nu_1, A\nu_1, \ldots, A^{k-1}\nu_1). \tag{9.20}$$

This is accomplished by generating k search directions formed by the l_2-orthonormal basis of the Krylov subspace K_k. The k search directions are obtained from a Gram-Schmidt method for computing orthonormal bases. The algorithm for determining the k directions is as follows: the first search direction is given by the initial residual, namely

$$\nu_1 = r_0 = b - Ax_0 \tag{9.21}$$

The remainder of the search directions are determined from the Gram-Schmidt method via

$$\nu_{j+1} = A\nu_j - \sum_{i=1}^{j} h_{i,j}\nu_i \qquad j = 1, 2, \ldots, k-1 \tag{9.22}$$

where,

$$h_{i,j} = \frac{(A\nu_j, \nu_i)}{(\nu_i, \nu_i)} \tag{9.23}$$

Once, the k search directions have been found, the approximate solution is advanced by determining which x_k element of the space formed by $x_0 + < \nu_1, \nu_2, \ldots, \nu_k >$ minimizes the l_2-norm of the residual vector,

$$r_k = b - Ax_k \tag{9.24}$$

The algorithm reduces to one of solving the least squares problem defined by

$$||r_k|| = \min_z ||\nu_1 + Az|| \quad \text{where} \quad z \in < \nu_1, \nu_2, \ldots, \nu_k > . \tag{9.25}$$

The final step is to update the approximate solution. If z_k satisfies equation (9.25), then the approximate solution is updated via,

$$x_k = x_o + z_k \tag{9.26}$$

This completes one iteration of the GMRES method. The method is continued until a user specified tolerance on the residual is met or a specified number of iterations have been performed.

Chapter 10

Mesh Sequencing

In the previous chapters, the focus has been on algorithms for modeling various aspects of the governing equations. Up to this point, little has been stated about the efficiency of the unstructured fluid dynamics code. In general, the efficiency of unstructured fluid dynamics codes fall short of that obtained in most structured codes available today. The primary cause of the reduction in efficiency is a direct result of the fact that the mesh has no orderly structure. Information about the connectivity of the cells, faces, and nodes must be provided which results in the requirement of the use of an indirect addressing system for the arrays. As a result, efficient algorithms such as the alternating direction implicit method are not available for unstructured fluid dynamics codes. These schemes must be abandoned in favor of more general algorithms, which are typically less efficient in CPU and memory requirements.

In order for unstructured codes to be competitive with their structured counterparts, methods for accelerating convergence to the steady state are required. For single grid methods, where the solution is obtained on a single mesh, the log of the spectral radius varies linearly with the spacing of the mesh. This makes computations on very fine meshes impractical in many situations. Most of the acceleration techniques applied in computational fluid dynamics operate on a series of meshes. Several methods for accelerating convergence using multiple meshes have been applied to unstructured fluid dynamics codes. Two methods which are closely related are the full approximate storage multigrid algorithm and the nested iteration or mesh sequencing

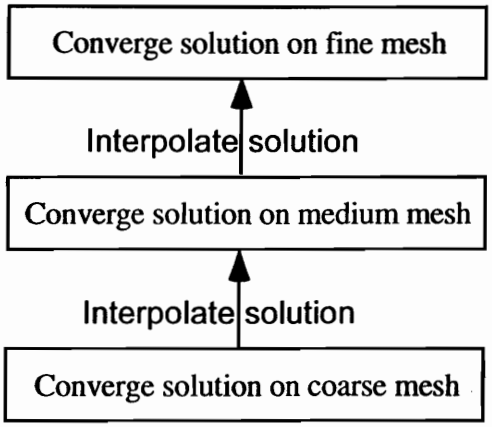


Figure 10.a: Mesh sequencing Pseudo algorithm

algorithm. Both methods operate on a hierarchy of meshes where the coarse meshes typically have 1/4 to 1/2 the nodes of the next finest mesh. They rely on the fact that convergence of low frequency waves is improved on coarse meshes where they appear as high frequency waves which are damped out significantly faster, especially when relaxation methods are employed.

The basic idea of mesh sequencing is to obtain the solution on coarse grids and use that solution as an initial guess on the finer meshes. The steps in the mesh sequencing method are illustrated in figure 10.a.

In this figure we have three meshes; coarse, medium, and fine. The first step in any mesh sequencing algorithm is to converge the solution on the coarse mesh and interpolate the converged solution to the medium mesh. The interpolated solution then acts as an initial guess to the solution on the medium mesh. The solution is converged on the medium mesh and interpolated to the fine mesh. This is continued until the finest mesh in the sequence has been reached. The last step is to converge the solution on the finest mesh in the hierarchy. Generally, the solution would only need to be converged 2 or 3 orders on each mesh. An important property of mesh sequencing is that it always steps from coarse to fine grids.

In contrast, the multigrid method is bi-directional. In the multigrid method, the solution is restricted from fine meshes to coarser meshes and prolongated from coarser meshes to finer meshes. This is illustrated in figure 10.b. The FAS multigrid method

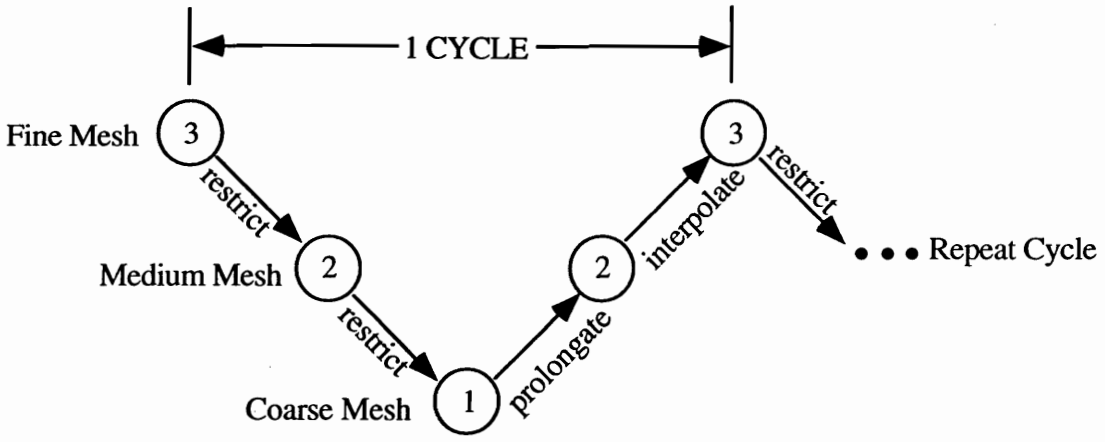


Figure 10.b: V-Cycle multigrid algorithm

for non-linear equations is complicated by the fact that the solution to the governing equations is not actually solved on the coarser meshes. Only on the finest mesh is the true solution obtained. On coarser meshes, the discrete equations are modified so that Q represents a correction to the finer grid solution. A detailed discussion of the multigrid method is outside the scope of this dissertation. The multigrid method has been implemented into unstructured codes by several researchers [52] [53] [54]. An excellent introduction to multigrid is provided in the paper by Brandt [55].

Mesh sequencing is a precursor to the multigrid method. Most multigrid algorithms include a form of mesh sequencing in the algorithm to obtain the initial finest mesh solution. Although multigrid has shown great promise in unstructured fluid dynamics codes, mesh sequencing was chosen for implementation into our unstructured code. The primary reason for chosing mesh sequencing over multigrid is that mesh sequencing does not require any modifications to the actual fluid dynamics code. Mesh sequencing can be accomplished external to the fluid dynamics code. Therefore, memory requirements are not increased above and beyond the memory required for the finest mesh. The coarser solutions require only the memory that would be needed to run these solutions from scratch. The implementation of mesh sequencing into the unstructured fluid dynamics code is the topic of this chapter.

Before we proceed to discuss the implementation of mesh sequencing, it should be noted that several methods for accelerating convergence on a single mesh have been

developed. These operate on the principle that we can treat the fluid dynamics code as a Newton type iteration process. In other words, we pass into the fluid dynamics code an initial solution vector Q_i and get back an updated approximation to the non-linear system of discrete equations, Q_{i+1}. Mathematically, this may be written as:

$$Q_{i+1} = F(Q_i) \tag{10.1}$$

where F represents a call to the fluid dynamics code. Attempts in this research were made to apply a Steffenson type accelerator [44] to a quasi-1D Euler code. Recently, several authors have applied a modified form of the GMRES algorithm applicable to non-linear systems of equations in order to accelerate convergence with good success [56] [57].

10.1 Implementation

The implementation of mesh sequencing in the unstructured package is handled externally from the fluid dynamics code. Therefore, this method can be used for practically all fluid dynamics codes, structured and unstructured.

In the mesh sequencing algorithm, the first step is to converge the solution on the coarsest mesh in the hierarchy. After the coarse grid solution has been obtained, the coarse grid solution is interpolated to the medium mesh where it will be used as an initial guess for the solution on the medium mesh. The interpolation of the solution from one mesh to another is accomplished by means of an efficient search algorithm based on quad-tree storage.

The quad tree is a storage structure which aids in sorting points in n-dimensional space [58]. After points have been placed in the quad tree structure, efficient routines may be used to locate points within n-dimensional volumes and/or near n-dimensional points. The quad tree can be best described by how the structure is filled. Suppose we want to fill a two dimensional quad tree structure. Points are placed one at a time into the quad tree structure which contains four storage locations. Figure 10.c shows the structure with four points occupying the four storage locations. At initialization all of the four storage locations are empty. One by one, points are added until the

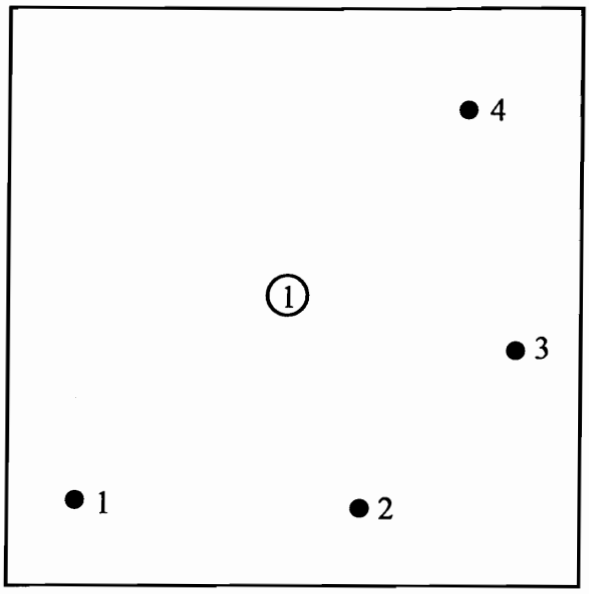


Figure 10.c: Initial quad tree structure before subdivision

four storage locations are filled. With the four storage locations filled, each dimension of the structure is divided in half. Therefore, the x-dimension is divided in half and the y-dimension is divided in half, giving rise to four subdivisions as shown in figure 10.d. In three dimensions, the z-dimension would also be divided in half, resulting in 8 subdivisions. Each subdivision becomes a new quad tree structure with four storage locations. The four points which made up the original structure are saved and used to locate the four new structures. These points are stored in the subquadrant which bounds that point, as shown in figure 10.d.

Now we are ready to add more points. With the initial quadrant subdivided, the new points are added to the appropriate bounding subquadrant. When one of the subquadrants becomes filled with four new points, the subquadrant is divided into four new quadrants. This is shown in figure 10.e. In this figure, the subquadrant labeled (2) has been filled and subdivided. This is continued until all of the points have been placed into the quad-tree structure.

In our algorithm, the locations of the cell centroids for the coarser mesh are placed into a quad-tree structure in this fashion. The location of the cell centroids on the finer mesh are then used in a search algorithm to find the coarse mesh cell centroid

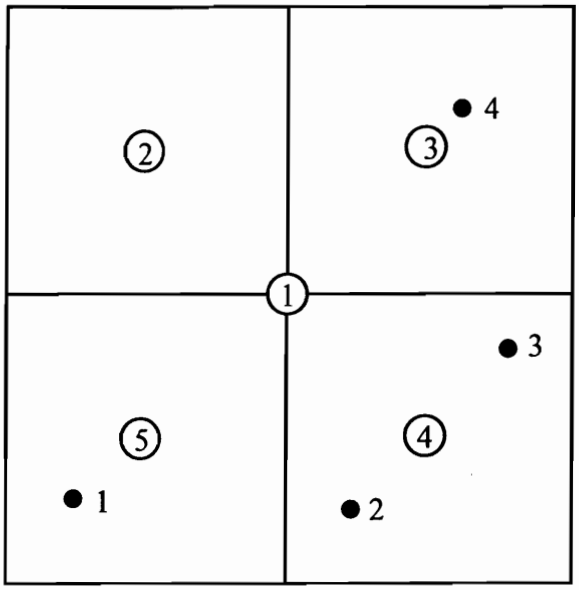


Figure 10.d: Initial quad tree structure after subdivision

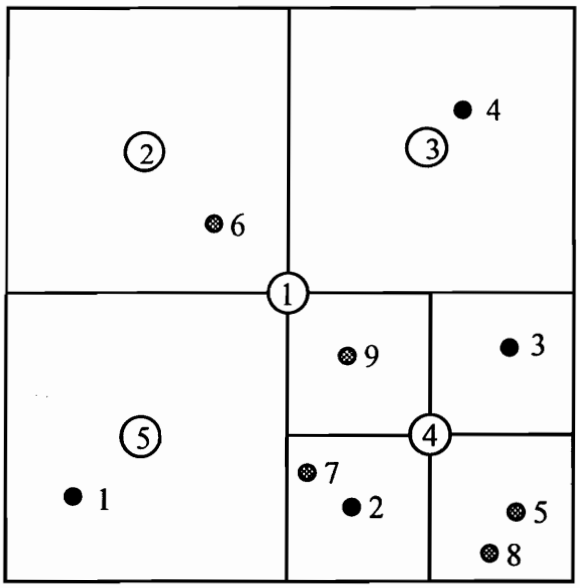


Figure 10.e: Quad tree structure after additional subdivisions

which lies near this point. In this algorithm, one simply steps down into the quad tree staying inside the subquadrants which bound the specified point. Eventually, you will reach a subquadrant which is not subdivided. The points stored in this quadrant will be near our given point. We feed each of our finest mesh cell centroids into this search algorithm. The solution at the resulting coarse mesh cell centroid is taken as our initial solution for the fine mesh cell. If higher accuracy is required, a reconstruction may be applied. For example, one could find the three closest points and create a $k = 1$ reconstruction using these points (in two dimensions). The reconstruction would then be evaluated at the fine grid cell centroid. For our purposes, the resulting solution obtained from the search algorithm was sufficient.

The last step is to write a solution file which may be used as the initial conditions for the finer mesh. The solution is converged on this mesh and the process is started again. This is continued until the finest mesh has been converged to the desired level of accuracy.

This algorithm has several advantages. Convergence on the coarser meshes is obtained much quicker than on the finer meshes. The physics may be setup quickly on these meshes, thereby reducing the computational requirements of the finer meshes. Furthermore, in many instances running a solution on a fine mesh with initialization to the freestream is either impossible or impractical due to the fact that the solution algorithms are less stable and require lower CFL's on the finer meshes. The aeroassist flight experiment vehicle is a prime example of this. In this case, running the solution on the finest mesh was not feasible without mesh sequencing.

Chapter 11

Solutions

Throughout the research, test cases were sought which would adequately test the individual components of the unstructured fluid dynamics code. In this chapter we present several of these test cases.

The first two solutions presented are designed to test the thermodynamics and chemistry modeling. The first of these solutions is the Ram II-c axisymmetric probe. The Ram II-c was considered by many to be the standard for testing thermo-chemical modeling. However, in recent years the focus has shifted to the Aeroassist Flight Experiment (AFE) vehicle. The AFE is a large blunt body operating at extremely high velocities. Solutions for the AFE problem are presented for a low speed case corresponding to a Mach number of 10 and a high speed case with a Mach number of 34.

The third solution presented is an analytic forebody. The analytic forebody is a three dimensional, inviscid, supersonic flow over the cockpit of a high performance aircraft. The analytic forebody has been studied both numerically and experimentally [59]. This problem was chosen to test the three dimensional *k-exact* reconstruction algorithm.

The last set of solutions presented are designed to test the viscous fluxes and the turbulence model. It should be noted, that the Ram II-c was run with the viscous fluxes. However, the primary focus in that discussion is on the thermo-chemical modeling. The first solution in this set is the subsonic, laminar flow over a two

dimensional flat plate. Solutions were obtained using both methods for computing the viscous fluxes discussed in chapter 7. The numerical solutions are compared against the analytical solution of Blasius. The second solution is the supersonic, turbulent flow over the same flat plate. For this case solutions are compared against the experimental data curve fit of Clauser [36].

11.1 Axi-symmetric Ram II-c

During the late sixties a series of probes were flown in a velocity regime in which ionization would play an important role. The second series of tests, known as the Ram II-c experiments, have become a popular test case for modern CFD codes. During the later series of flights, water and electrophilic liquid [60] were injected into the plasma layer to reduce the free electron density level. However, only data for the flights without injection are available.

The geometry of the probes consisted of a sphere–cone configuration with a 15.24 cm nose radius and a 9 degree cone half angle. The probes were instrumented to measure electron number density during the flight. Eight wire probes were located on the leading edge of the rake with the last probe extending approximately 7 cm. into the plasma layer. Collectors were placed with their longitudinal axis at 46 degrees to the flow direction and biased with a constant negative voltage to collect ions.

The case considered in this paper corresponds to a flight altitude of 61 Km with free stream conditions: $T_\infty = 254K$, $U_\infty = 7650$ m/s, and a Reynolds number based on the nose radius $R_{e_n} = 19,500$. These conditions correspond to a flight Mach number of 23.9. The chemistry model used to obtain the solutions is the air chemistry model described by Park [48]. The Park model contains 7 species and 18 chemical reactions. The model is summarized below:

Species	Reactions
N2	N2 + N2 --> N + N + N2
O2	N2 + O2 --> N + N + O2
NO	N2 + NO --> N + N + NO
NO+	N2 + N --> N + N + N

```
N                    N2 + O  --> N + N + O
O                    O2 + N2 --> O + O + N2
e-                   O2 + O2 --> O + O + O2
                     O2 + NO --> O + O + NO
                     O2 + N  --> O + O + N
                     O2 + O  --> O + O + O
                     NO + N2 --> N + O + N2
                     NO + O2 --> N + O + O2
                     NO + NO --> N + O + NO
                     NO + N  --> N + O + N
                     NO + O  --> N + O + O
                     O + N2  --> N + NO
                     NO + O  --> N + O2
                     O + N   --> NO(+) + e(-)
```

The free stream was assumed to consist of standard air percentages for the mass fractions of N_2 and O_2. All other species were assumed to be absent from the free stream. The equilibrium rate constants are computed via equation 3.15.

Figure 11.a shows the mesh used to obtain the solutions presented here. This mesh contains approximately 3,500 interior cells and 10,000 total cells.

The solution is presented in figure 11.b. This figure shows the temperature contours over the Ram II-c. The solutions were obtained using the *k-exact* reconstruction for the viscous fluxes and Van Leer flux vector splitting for the inviscid fluxes. Notice the high temperatures behind the shock, particularly in the nose region. It is in this region where the chemical effects will become important. This will be seen more dramatically in the Aeroassist Flight Experiment solutions presented in the next section. For this problem the high temperature effects are clearly seen in figure 11.c. In figure 11.c the computed and experimental electron number densities on the surface of the probe are presented.

The unstructured code produces good results over the nose of the Ram II-c, however, it slightly overpredicts the electron number density on the aft section of the body. Recall, in the thermodynamics chapter two important assumptions were made.

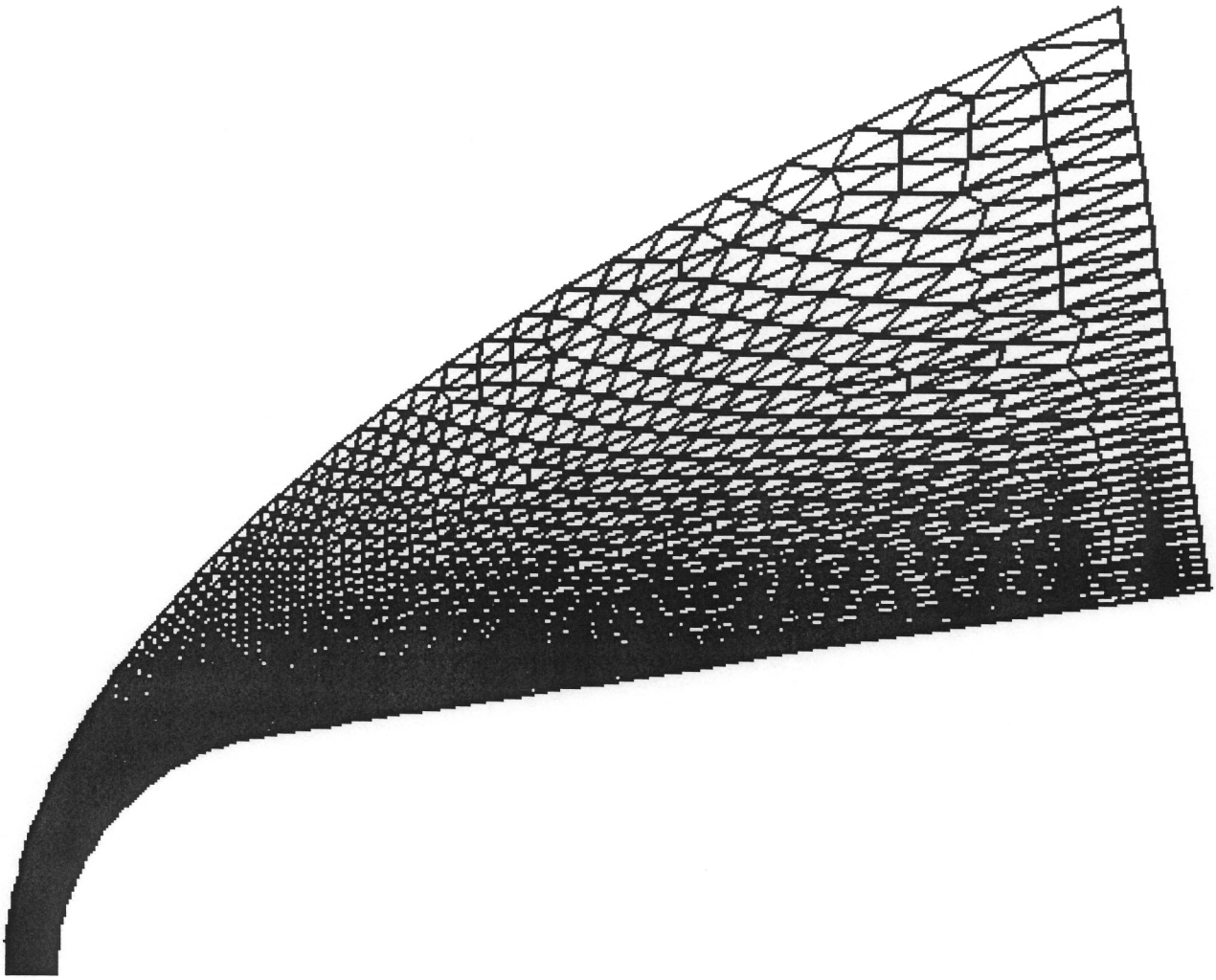


Figure 11.a: Ram II-c prism mesh

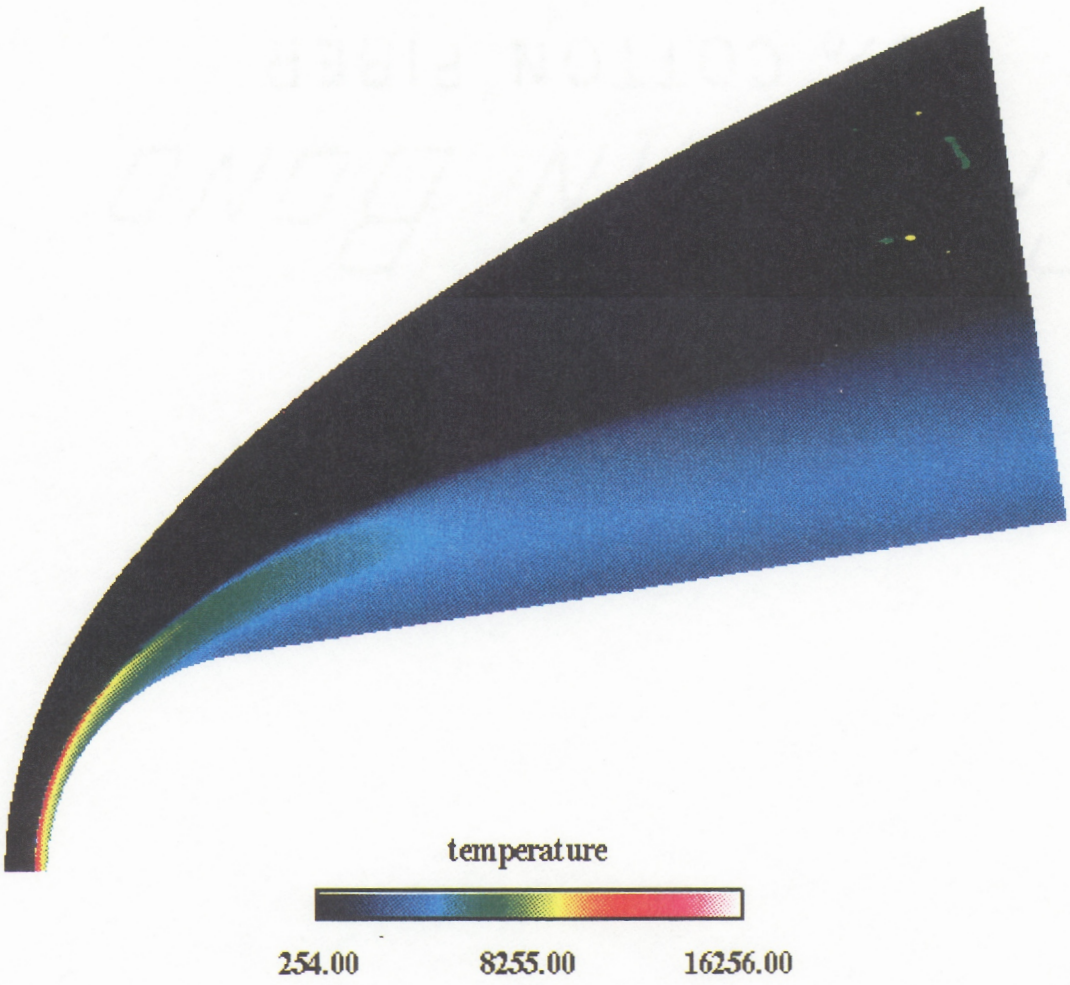


Figure 11.b: Ram–II–c Temperature Contours

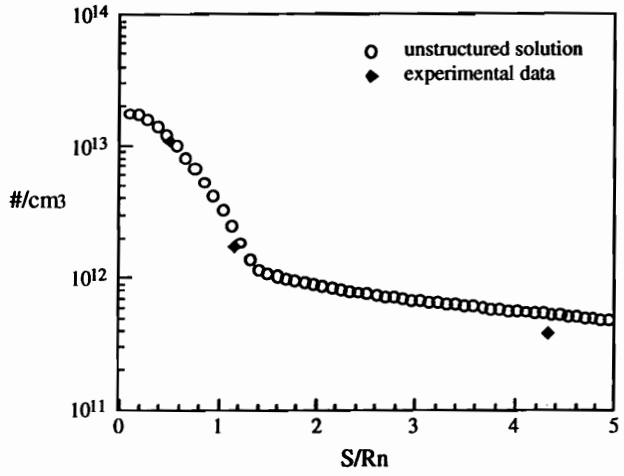


Figure 11.c: Electron number density on surface of Ram II-C probe

The first was that each species is subjected to the same temperature. This assumption implies that the mass of each species is on the same order. Obviously for this test case, that is not true. The density of the electrons is several orders lower than that of the other species. In addition, electronic excitation was neglected. Neither of these assumptions are completely accurate for this flow. However, despite these assumptions, the results obtained for this flow are comparable to the solutions obtained by structured codes with equivalent levels of thermo-chemical modeling.

11.2 3-D AFE

A new family of vehicles, known as aeroassist vehicles, are currently under design. These vehicles typically operate in the upper reaches of the atmosphere and at velocities higher than those encountered by typical reentry vehicles.

A class of these vehicles, known as aeroassist orbital transfer vehicles (AOTV), are being designed to transfer payloads from a low earth orbit to a high earth orbit and back. In returning, the vehicles velocity must be greatly reduced to the earth orbital velocity and achieve capture in low earth orbit. This reduction in velocity may be achieved by one of two methods, the use of retro rockets or by guiding the

vehicle through the earths atmosphere allowing aerodynamic drag forces to act on the vehicle. Studies indicate that fuel loads are lowered and therefore the payload may be increased using the second method. Because of the high altitude and high velocity, ground based facilities are not capable of reproducing these conditions. In order to better understand the physics of these problems, the AFE has been proposed by the National Aeronautics and Space Administration. The vehicle will be transported by the space shuttle and placed in a low earth orbit. The AFE will then be propelled into the atmosphere to simulate the velocity and trajectory of a return mission from geosynchronous orbit. The data obtained will then be used to validate current computational fluid dynamics codes which will be used in future AOTV designs.

The AFE will travel in the Earths upper atmosphere at velocities ranging from 7 to 10 km/s. At these conditions, chemical and thermal non-equilibrium effects are significant. The AFE project was designed to obtain critical flight data that will be used to validate hypersonic computational fluid dynamic codes. Two sets of solutions for the AFE were obtained. The first set of solutions corresponds to a free stream Mach number of 10. The Mach 10 case is significant because good experimental data exists for this problem. The second set of solutions is for a free stream Mach number of 31.7. In this case only code on code validations are available at this time.

Solutions to the AFE proved to be very stiff and difficult to obtain. The solutions presented here were obtained with the GM-Res time integration scheme. In fact, the other two time integration schemes were incapable of running this problem effectively. To further increase the efficiency, mesh sequencing was used on a series of meshes. A total of 5 meshes were generated in the process. Several views of the finest AFE mesh are shown in figure 11.d. The body of the AFE has been mirrored about the plane of symmetry to provide a full view of the proposed configuration. The mesh displayed in figure 11.d contains approximately 106,000 cells and 202,000 faces.

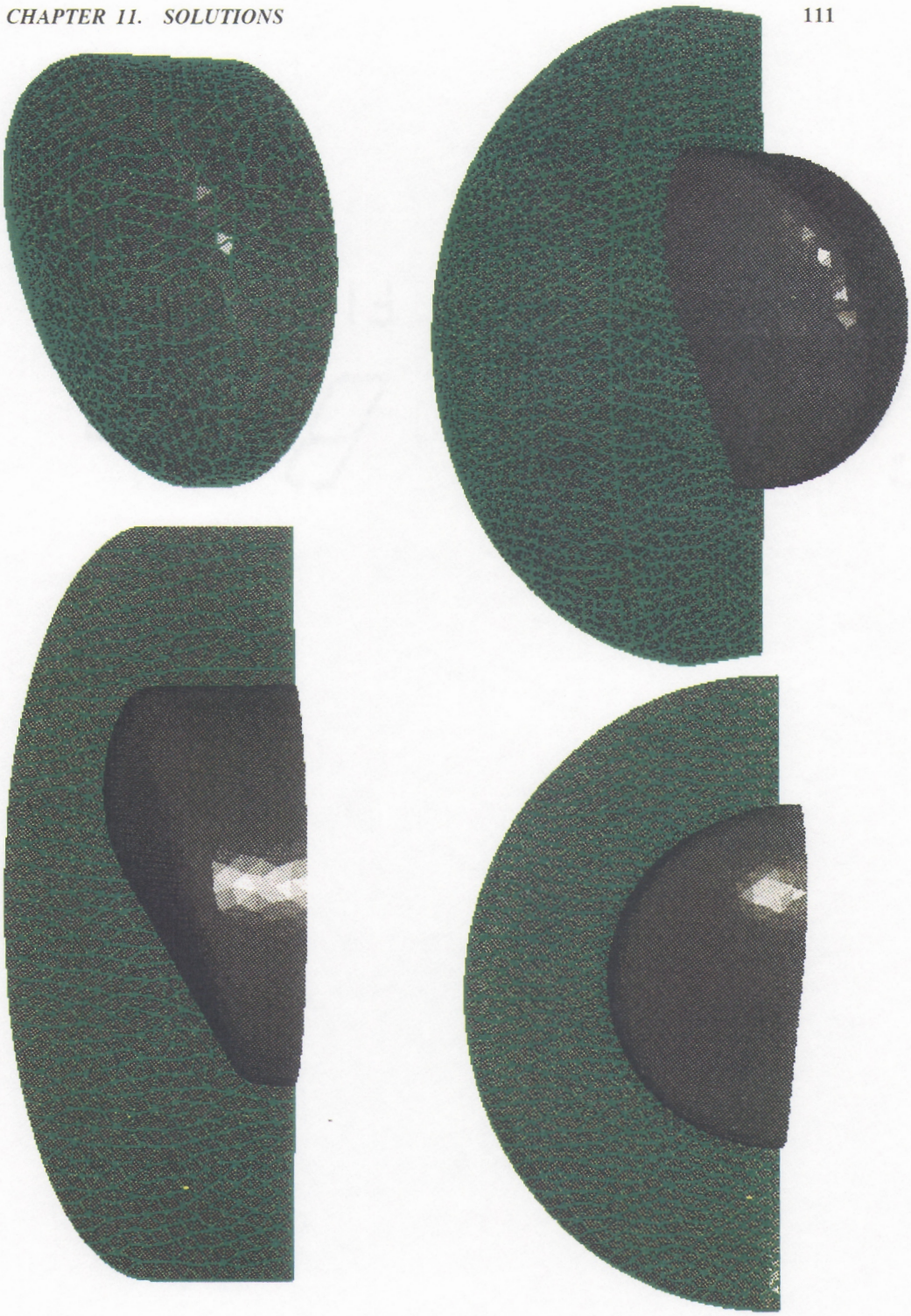


Figure 11.d: Several views of the Aeroassist Flight Experiment mesh. Clockwise from top: surface, outer boundary, symmetry plane, and exit plane.

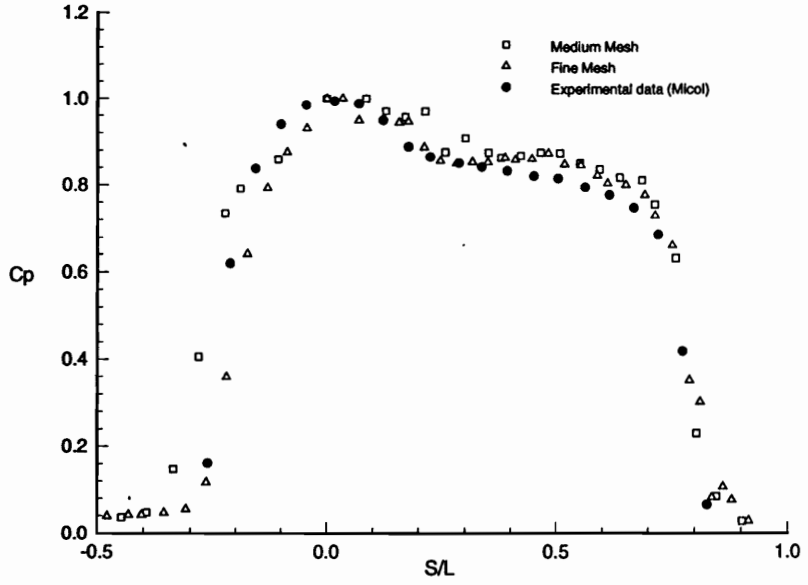


Figure 11.e: Pressure distribution on surface of AFE for Mach 10 test case

Solutions for the Mach 10 test case are presented in figures 11.e and 11.f. Figure 11.e shows a comparison of the computed and experimental data for the pressure coefficient over the surface of the AFE in the plane of symmetry. Two sets of computed results are presented in this figure; the first set corresponds to a medium mesh of approximately 50,000 cells, and the second set corresponds to the fine mesh shown in figure 11.d. As can be seen from the figure, both sets of computed results compare favorably to the experimental data of J.R. Micol [28]. The figure also shows that the finer mesh produces better results, particularly in regions of large gradients.

In figure 11.f, clockwise from the top left, the mixture density, diatomic nitrogen density, internal energy, and pressure are shown. The free stream pressure and temperature for this test case are 0.03239 psi and 89.99 R respectively. The flow was assumed to be in chemical non-equilibrium and thermal equilibrium. The 7 species - 18 reaction Park air chemistry model was used for both the Mach 10 and Mach 31.7 test cases. The freestream species densities were computed in the same fashion as the Ram II-c discussed in the previous section.

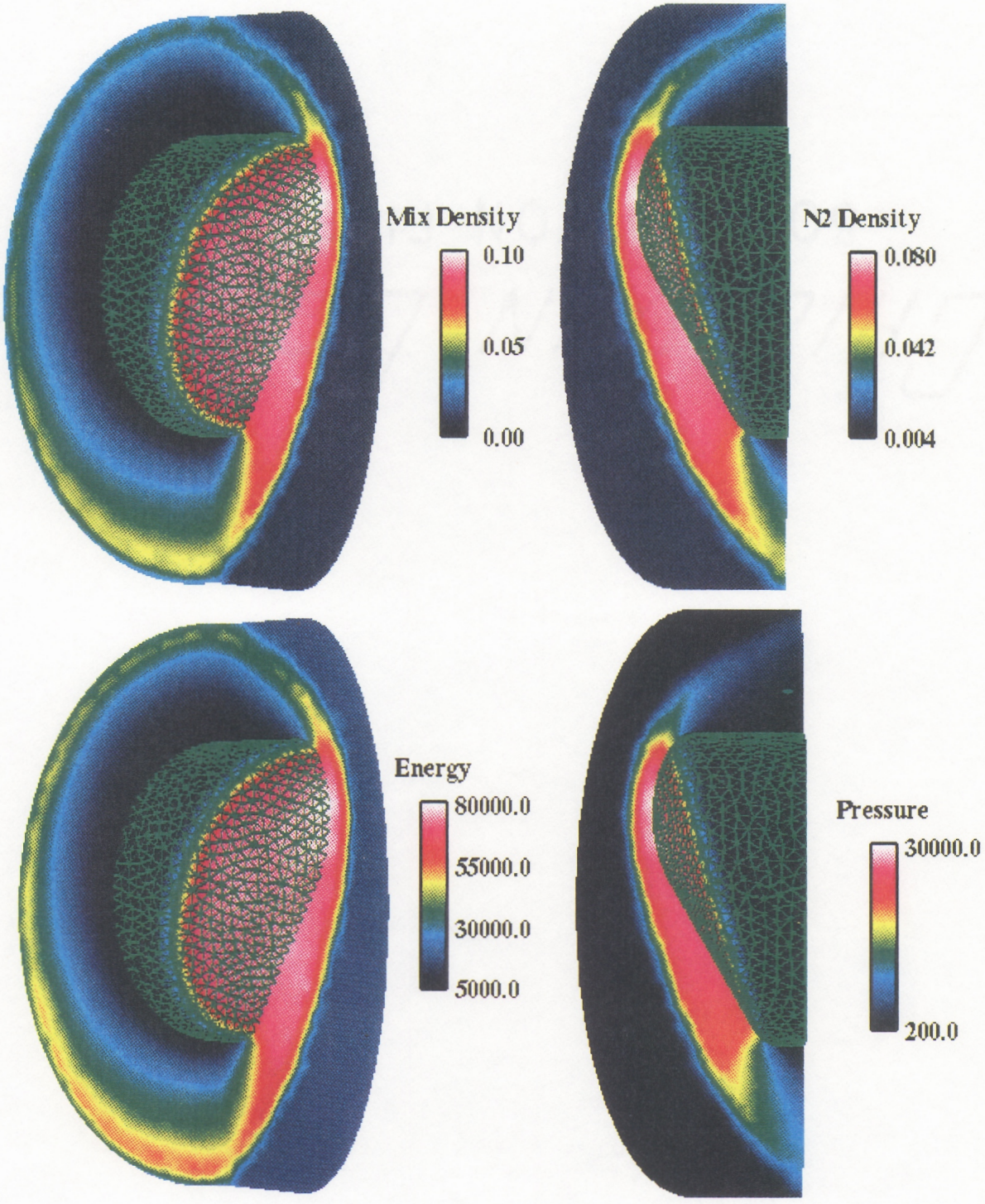


Figure 11.f: AFE solutions for the Mach 10 case. Clockwise from top: mixture density, nitrogen density, pressure, and energy

Results for the Mach 31.7 test case are presented in figures 11.g and 11.h. The freestream conditions for this test case correspond to point 1 of the planned trajectory. This corresponds to a free stream velocity of 9.818 Km/s, a mixture density of 1.542×10^{-5} kg/m^3, and a temperature of 195.3 K. The flow was assumed to be in chemical and thermal non-equilibrium.

In figure 11.g, clockwise from top left, the mixture density, electron number density, and temperature are shown. The electron number density plot compares favorably to the solutions obtained by Greendyke, Gnoffo, and Lawrence [29]. In figure 11.h, clockwise from top left, the density of N_2, N, O_2, and O are shown. Obviously, the shock shown in these figures is much stronger than the shock for the Mach 10 test case. As a result, the temperatures behind the shock are much higher as can be seen in figure 11.g. Recall from the chemistry chapter that ionization of air (at 1 atm) begins to occur above temperatures of 9000K. Below that, dissociation reactions are dominant. At these temperatures, diatomic oxygen is almost totally dissociated and diatomic nitrogen has undergone considerable dissociation. For this problem, the temperatures far exceed 9000K behind the shock. Therefore, we would expect to see strong ionization effects. This is clearly shown in the electron number density plot. The Park model has one reaction for the ionization, that being the ionization of NO. From the diatomic oxygen and monotomic oxygen contours, it is evident that the oxygen in the freestream has undergone complete dissociation behind the shock. In the region of the body, the density of the diatomic oxygen is nonexistent. Likewise, the density of monotomic oxygen reaches a peak behind the shock. The nitrogen in the freestream undergoes considerable dissociation behind the shock, however it does not undergo complete dissociation as did the oxygen. In all of these figures, it is clear that the chemical reactions are confined to the region between the shock and the body of the AFE. The temperatures in the freestream are not high enough to cause dissociation or ionization.

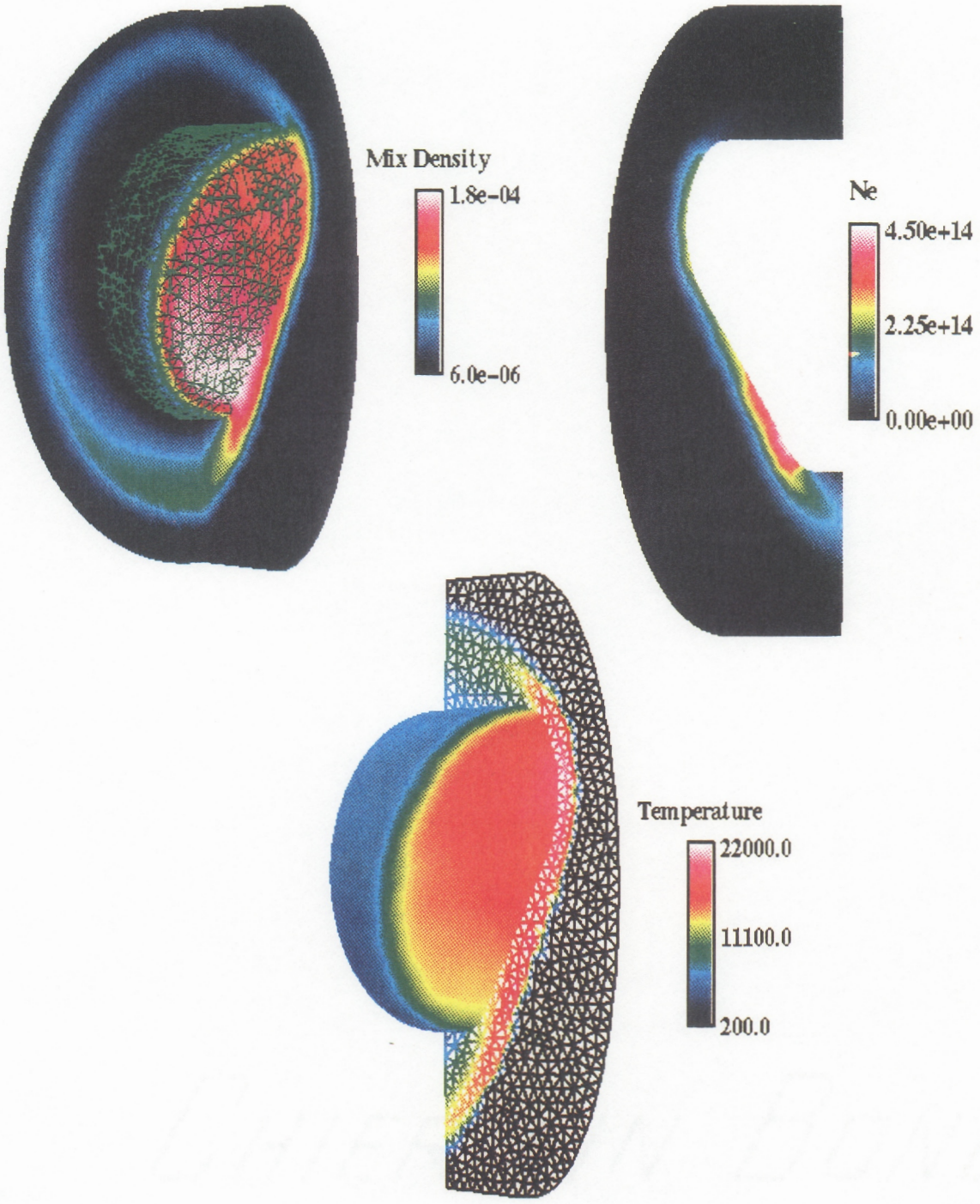


Figure 11.g: AFE solutions for the Mach 31.7 case. Clockwise from top: mixture density, electron number density, and temperature

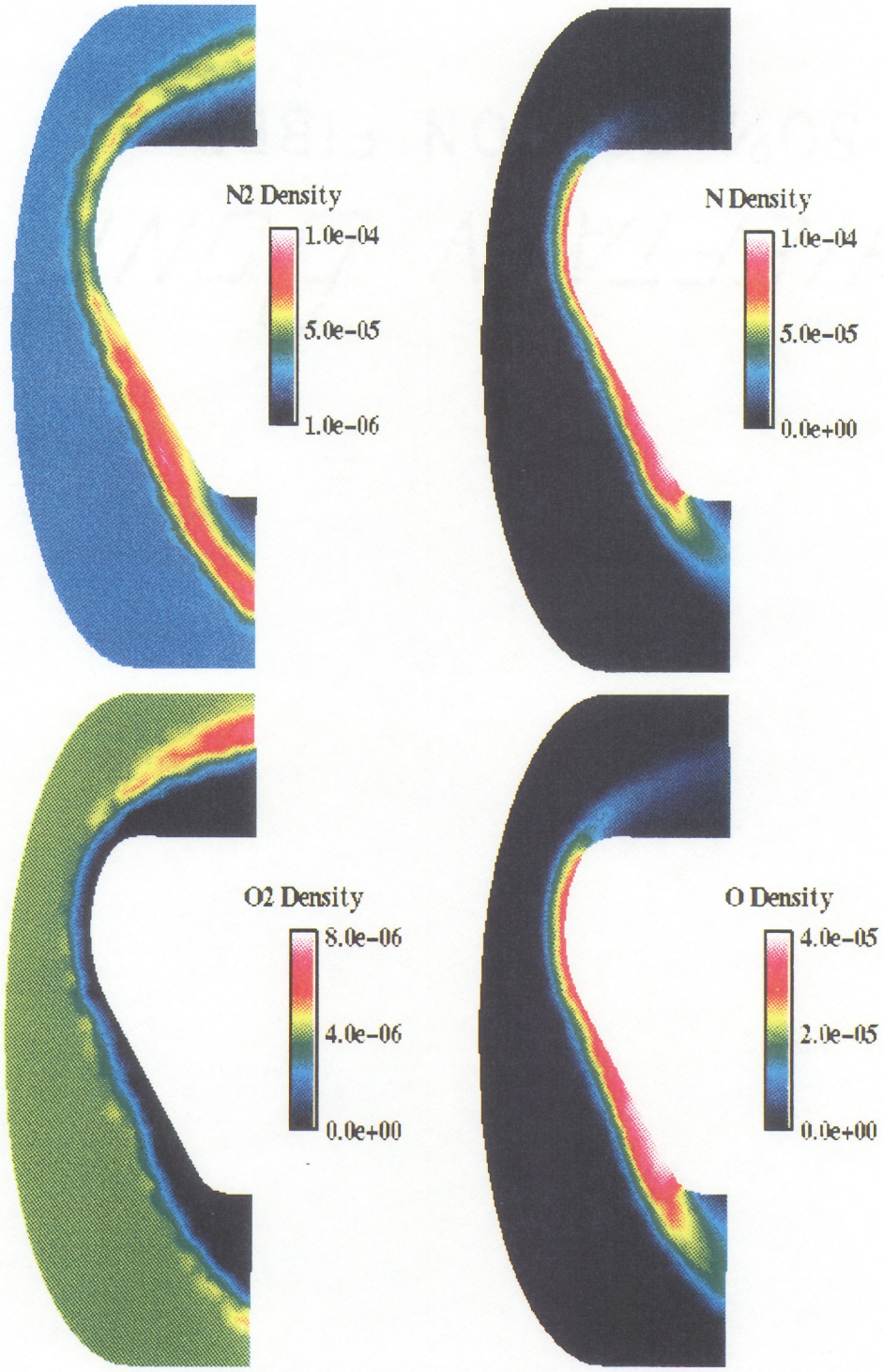


Figure 11.h: AFE solutions for the Mach 31.7 case. Clockwise from top: N2 density, N density, O2 density, O density

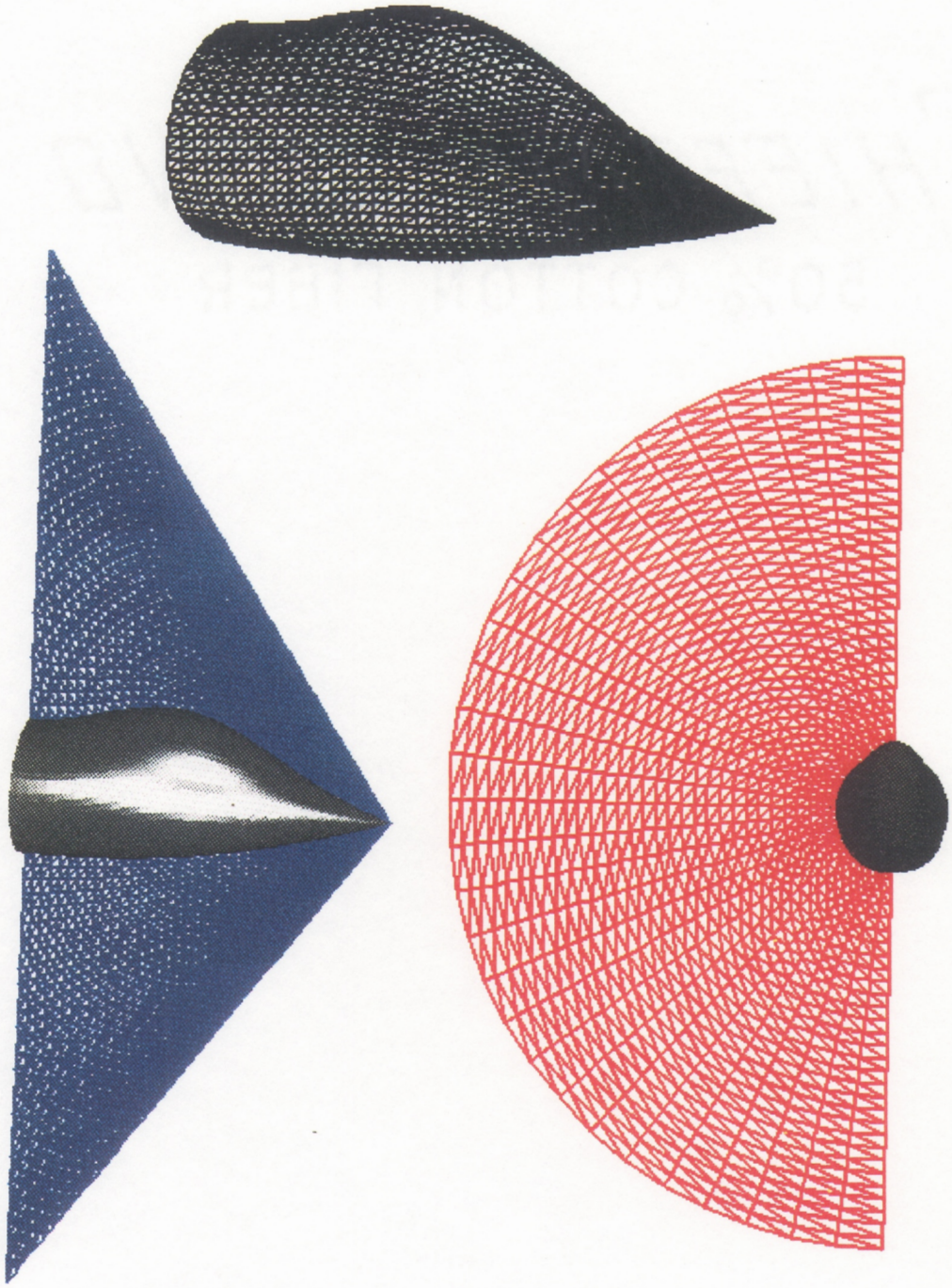


Figure 11.i: Several views of the analytic forebody mesh. Clockwise from top: surface, exit plane, and symmetry plane

11.3 3-D Analytic Forebody

The analytic forebody problem has been studied experimentally as well as numerically [59]. The problem represents the inviscid supersonic flow over the cockpit region of a high performance aircraft. The geometry consist of a cross section which develops smoothly from circular at the nose through a lobed analytic curve and back to circular at the aft section of the body. This case was chosen to test the *k-exact* reconstruction method in three dimensions. The solution was obtained using mesh sequencing on two meshes. The finer of the meshes is shown in figure 11.i. This mesh corresponds to a structured 31x31x41 mesh which has been subdivided into tetrahedral cells. Three views of the forebody are shown, clockwise from the top, they represent the surface of the forebody, exit plane, and symmetry plane. The mesh in figure 11.i contains approximately 230,000 cells, 440,000 faces, and 40,000 nodes.

The freestream conditions for this problem are given by: $M_\infty = 1.7$, $\rho_\infty = 1.1774$ kg/m^3, $P_\infty = 101325$ N/m^2, and the angle of attack $\alpha = 0$. The solution was obtained using the *k-exact* reconstruction method for the inviscid fluxes. The inviscid fluxes were computed using the Roe flux difference splitting algorithm discussed earlier. The Runge-Kutta explicit time integration scheme was used to advance the solution in time. The solution on the coarse mesh required 11306 iterations to reduce the residual 6 orders. On a Cray YMP this required 17 minutes of CPU time. The finer solution was converged an additional 2 orders in 8306 iterations which required 1 hour and 46 minutes of Cray YMP time.

The solution to the forebody is shown in figures 11.j and 11.k. In figure 11.j, the pressure contours are shown on the top and bottom planes of symmetry. In figure 11.k, the surface pressure distribution in the plane of symmetry is compared to experimental data [59]. As can be clearly seen, the results compare favorably to the experimental data. Note that in the aft section of the forebody the predicted values are slightly lower than the experimental data. Papay [61] showed that for an inviscid solution to the analytic forebody this is typical. He also showed that solutions obtained with with an appropriate turbulence model provide better agreeement in this region.

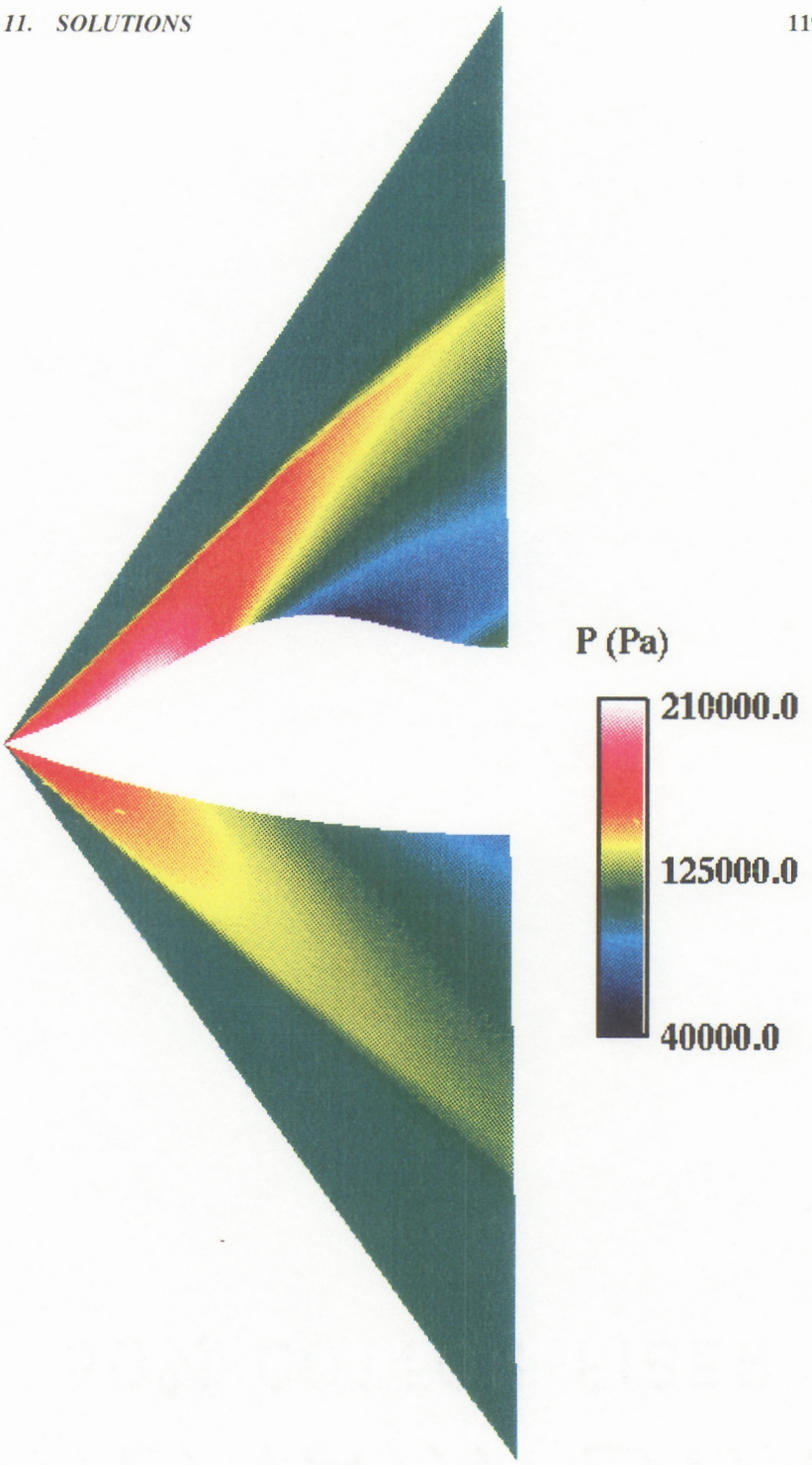


Figure 11.j: Analytic forebody pressure contours in symmetry plane

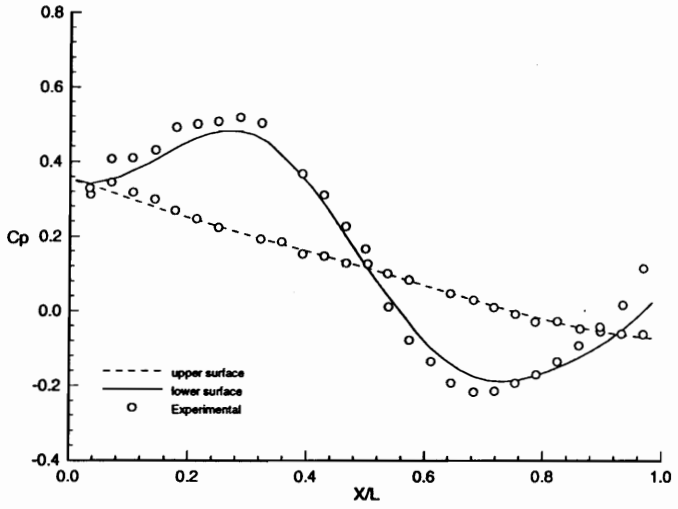


Figure 11.k: Surface pressure distribution in symmetry plane

11.4 2-D Laminar Flat Plate

The two dimensional flow of a perfect gas over a one meter long flat plate at a freestream Mach number of 0.3 is presented in figure 11.l. The freestream temperature and density are 500 K and 0.03973 kg/m^3 respectively. The Reynolds number based on the length of the plate is 200,000 which is well within the valid range for laminar flow. The problem is aligned so that the plate runs along the x-axis and the y-axis is normal to the plate. To obtain the solution, a quadrilateral 21x41 mesh was converted to a triangular mesh by subdivided each quadrilateral into two triangular cells.

The boundary condition at the inflow fixes the total temperature and total pressure. The pressure is extrapolated from the interior, and the density and velocity are computed based on the extrapolated pressure and the specified total temperature and total pressure. The boundary condition at the outflow sets the pressure to the specified back pressure. All other flow quantities are extrapolated. The surface of the flat plate is modeled using a no-slip adiabatic boundary condition. This boundary condition extrapolates the species densities, pressure, and temperature from the first interior cell. The velocity components u, v, w are set to zero. For the top boundary,

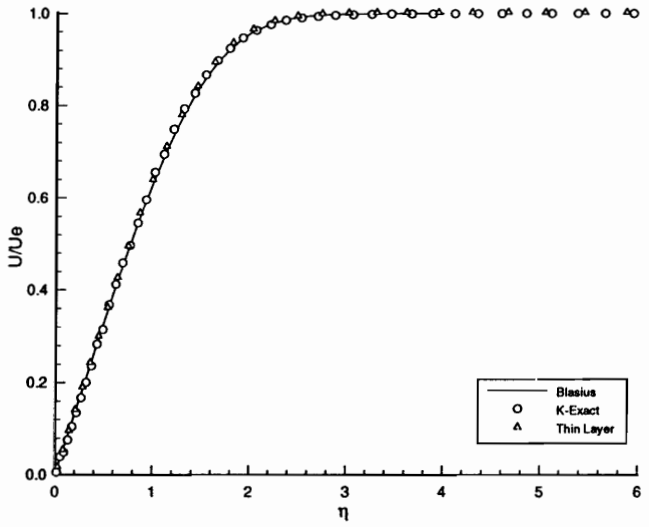


Figure 11.1: Similarity profiles for subsonic laminar flat plate

all values are extrapolated from the interior.

Figure 11.1 shows a comparison of the similarity profiles computed at the exit plane to those of Blasius. Two solutions are presented in this figure. One of the solutions was computed using the *k-exact* reconstruction method for the viscous flux. The other solution was obtained using the thin layer approximation. The solutions were obtained using the GMRES implicit time integration, along with a second order k-exact reconstruction for the inviscid fluxes.

Both solutions provide excellent results. The numerical boundary layer thickness is approximately 0.012 meters which compares favorably to the Blasius value of 0.0116.

11.5 2-D Turbulent Flat Plate

The solution presented in this section represents the two dimensional turbulent, supersonic flow over a one meter long flat plate. The conditions in the freestream correspond to $M_\infty = 2.244$, $T_\infty = 170K$, and $\rho_\infty = 0.4538kg/m^3$. This corresponds

to a Reynolds number of $2x10^7$. The solutions were obtained using the one equation Spalart and Allmaras model discussed earlier. The turbulence model working variable was initialized to ten times the freestream laminar viscosity. The *k-exact* reconstruction algorithm was used to determine the viscous and inviscid fluxes. The wall boundary condition extrapolates the density, pressure, and temperature and sets the velocity components to zero. This corresponds to an adiabatic wall boundary. To obtain the solution, a 41x81 quadrilateral mesh was converted to a triangular mesh. This was accomplished by subdividing each quadrilateral cell into two triangular cells. The solution was advanced in time using the Runge-Kutta explicit time integration scheme. The solution required 59853 iterations to converge the solution 6 orders. On a Cray YMP this required approximately 23 minutes.

The solution is shown in Figure 11.m. Figure 11.m compares the unstructured solution at the exit plane to the analytic law of the wall plot. The analytical solution was obtained using the coefficients developed by Clauser [36]. Using these coefficients, the equation for the log region is given by

$$u^+ = 5.6 \log(y^+) + 4.9 \tag{11.1}$$

In this figure, the frictional velocity u^* was taken to be constant throughout the boundary layer. The value of the frictional velocity was obtained from

$$u^* = \sqrt{\frac{\tau_w}{\rho_w}} \tag{11.2}$$

The wall shear stress was evaluated using the vorticity at the first cell off the wall.

Figure 11.n shows the profile of the working variable, $\tilde{\nu}$, in the exit plane. The working variable is nondimensionalized by the reference viscosity and density.

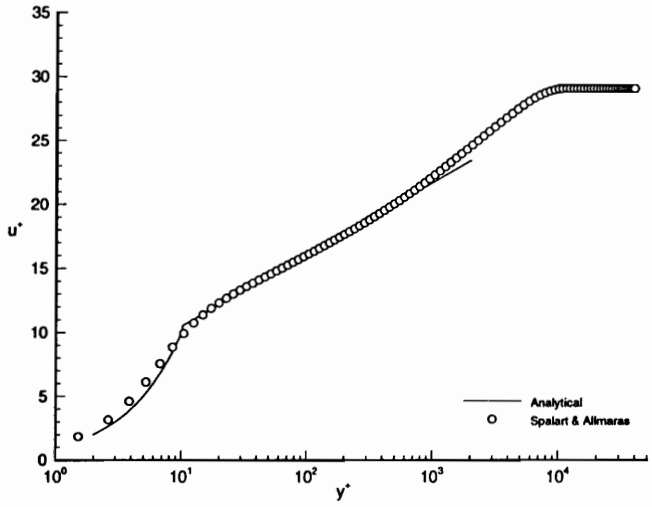


Figure 11.m: Wall law plot for the supersonic flat plate

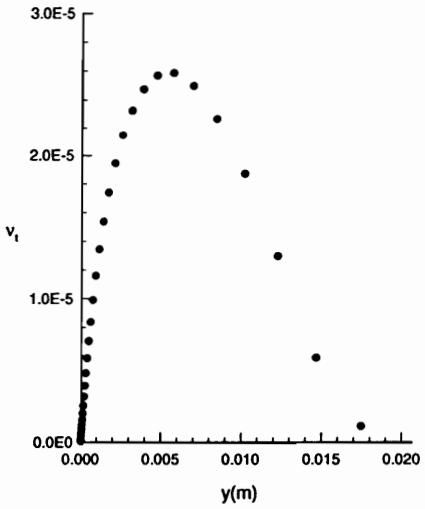


Figure 11.n: Turbulent viscosity profile in the exit plane

Chapter 12

Conclusions

Algorithms for solving the full Navier-Stokes equations with a generalized thermo-chemical model and one-equation turbulence model in an unstructured grid environment were presented. In the governing equations, we saw that the addition of a generalized thermo-chemical model was accomplished via the addition of N species continuity equations and M equations accounting for the species which have non-equilibrium vibrational contributions to their internal energy. The equilibrium and non-equilibrium chemistry was handled through source terms for the N species continuity equations. The source terms were derived from empirical relations. Several of the more common methods for computing the forward and backward rate constants were discussed.

Three upwind solvers for the inviscid fluxes for flows in thermal equilibrium and non-equilibrium were presented. The upwind solvers presented follow the work of Grossman and Cinnella and included the flux vector splitting techniques of Steger and Warming and Van Leer, and the flux difference splitting technique developed by Roe. Several equilibrium thermodynamics models were discussed in relation to the upwind solvers and the equation of state. The equilibrium thermodynamic models include a model based on each species behaving as a calorically perfect gas, a statistical mechanics model where we assume a harmonic oscillator formulation for the vibrational contribution, and a model based on the NASA Lewis Research Center curve fits. We also saw that by integrating the primitive variables in time, we can simplify

the implementation of the equation of state. Namely, we can directly compute the temperature instead of using an iterative technique. In addition to the equilibrium contribution to the internal energy, additional governing equations were presented for modeling non-equilibrium vibrational contributions to the internal energy. The source terms were derived from basic statistical mechanics considerations. The source terms were shown to have two contributions, an elastic contribution due to molecular collisions and an inelastic contribution due to chemical reactions. Solutions to the Aeroassist Flight Experiment vehicle and Ram-IIc where shown to produce good results for flows in which dissociation and ionization play an important role. We also saw from these solutions that while the assumptions made in deriving the thermo-chemical model are not completely valid for these problems, the thermo-chemical model is still capable of simulating these types of flows.

Several methods for interpolating the cell averaged data to the faces for the unstructured upwind solvers were presented. These methods include the *k-exact* reconstruction method and a linear gradient-based reconstruction method. The *k-exact* reconstruction presented closely follows the work of Mitchell and Walters. In the algorithm presented for the inviscid fluxes, a separate reconstruction is obtained for the left and right state. This required two separate stencils to be developed. As was discussed, no single method proved to be adequate when the *k-exact* reconstruction was extended to three dimensions. Therefore, several algorithms for choosing the inviscid stencils were given. Solutions to the three dimensional analytic forebody using the *k-exact* formulation were presented and compared to both experimental and numerical data.

One of the major issues in solving the Navier Stokes equations on unstructured meshes is the modeling of the gradients. In the research presented here, the gradients were computed using the derivatives of the reconstruction weights obtained from the *k-exact* reconstruction. For the viscous fluxes a single centered stencil was used. Like the inviscid stencil selection, several methods were presented for picking the viscous stencil. The modeling of the transport properties was discussed for multicomponent mixtures. Appropriate mixture rules for the species transport properties were given.

Solutions to the axisymmetric Ram-IIc and a subsonic flat plate were shown to produce good agreement with experimental and analytical data.

The physical modeling was completed with the Spalart and Allmaras one equation turbulence model. In their original work, Spalart and Allmaras developed a differential transport equation for the turbulent viscosity. In this research, the differential transport equation was transformed into an equivalent integral transport equation. A Van Leer type upwind scheme was then applied to the convective flux. The gradients in the diffusive fluxes were obtained in a similar fashion to the gradients in the viscous fluxes. Namely, by differentiating the *k-exact* reconstruction weights. A simple method for computing the wall distance function was presented. Results for a supersonic, two dimensional flat plate were presented and compared to experimental data and numerical solutions from a structured code using the Baldwin-Lomax algebraic model and a compressible form of the two equation K-Epsilon model.

Several methods for advancing the solution in time were presented, including the m-stage Runge Kutta time integration and Euler implicit algorithm with inner iterations. We saw that linearization of the Euler implicit scheme produces a system of linear equations. Two iterative methods for solving the resulting linear system were discussed, a block Jacobi relaxation and the generalized minimum residual method (GMRES). A brief discussion on integrating the primitive variables was given. To increase the efficiency of the unstructured flow solver a simple algorithm for mesh sequencing was discussed.

Even though good progress is being made in the unstructured arena, this technology still has a way to go in certain areas before it can be competitive with current structured technology. Several of the areas which are in need of improvement include mesh generation for unstructured viscous meshes, efficiency of the numerical procedures in terms of memory and computational resources, robustness of the reconstruction methods, and thorough testing of the unstructured turbulence model in three dimensions. In this research, the testing of the turbulence model was confined to two dimensional problems. This was largely due to inadequate techniques for obtaining the required stretching in the boundary layer with tetrahedral grid generators.

In this writers view, *k-exact* is beneficial to the researcher since it can be applied to a wide variety of problems as demonstrated here. In this research *k-exact* reconstruction is applied to the inviscid fluxes, the gradients of the viscous fluxes, and the gradients in the transport equation for the turbulence model. Other areas where *k-exact* reconstruction would be useful include interpolation of data to a line for output and enhancing the mesh sequencing procedure. However, from a production level point of view, the method is not robust enough to be of tremendous value. The use of the *k-exact* reconstruction technique requires that the user of the code has a good knowledge of the stencil selection procedures and in many cases must be able to modify the code to produce the desired results. Research into non-fixed stencils may provide some insight into the stability of the algorithm.

Mesh sequencing provided a great benefit particularly for complex flows like the Aeroassist Flight Experiment. Several authors have shown results using multigrid techniques for unstructured flow solvers, which look promising. [52] [53] [54]. Research into space marching on unstructured grids and better time integration strategies are still needed at this point. Many of these advanced algorithms require a direct link between the grid generation and solution process. In order for the unstructured technology presented here to be competitive in todays market, research is needed in improving the efficiency of unstructured fluid dynamics codes. Inherently this will require simultaneous research into both unstructured mesh generation and fluid dynamics technologies.

Bibliography

[1] Courant, R., Issacson, E., and Rees, M., "On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences," *Comm. Pure and Applied Mathematics*, vol. 5, pp. 243–255, 1952.

[2] Harten, A., Lax, P. D., and Van Leer, B., "On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws," *SIAM Review*, vol. 25, no. 1, pp. 35–61, 1983.

[3] Roe, P. L., "Characteristic-Based Schemes for the Euler Equations," *Annual Review of Fluid Mechanics*, vol. 18, pp. 337–365, 1986.

[4] Steger, J. L. and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Applications to Finite-Difference Methods," *Journal of Computational Physics*, vol. 40, 1981.

[5] Van Leer, B., "Flux-vector Splitting for the Euler Equations," in *Lecture Notes in Physics*, vol. 170, Springer-Verlag, 1982.

[6] Godunov, S. K., "A Difference scheme for numerical computation of discontinous solution of hydrodynamic equations," *Math. Sbornik*, vol. 47, 1959. (translated from Russian by US Joint Publ. Res. Service, JPRS 7226).

[7] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, vol. 43, pp. 357–372, 1981.

[8] Colella, P. and Glaz, H. M., "Efficient Solution Algorithms for the Riemann Problem for Real Gases," *Journal of Computational Physics*, vol. 59, p. 264, 1985.

[9] Vinokur, M. and Liu, Y., "Equilibrium Gas Flow Computations II: An Analysis of Numerical Formulations of Conservation Laws," AIAA 88-0127, 1988.

[10] Vinokur, M., "Flux Jacobian Matrices and Generalized Roe Averge for an Equilibrium Real Gas," NASA CR-117512, 1988.

[11] Vinokur, M. and Montagné, J. L., "Generalized Flux-Vector Splitting for an Equilibrium Real Gas," NASA CR-117512, 1988.

[12] Glaister, P., "An Approximate Linearised Riemann Solver for the Euler Equations for Real Gases," *Journal of Computational Physics*, vol. 74, p. 382, 1988.

[13] Glaister, P., "An Approximate Linearised Riemann Solver for the Three-Dimensional Euler Equations for Real Gases Using Operator Splitting," *Journal of Computational Physics*, vol. 77, 1988.

[14] Grossman, B. and Walters, R. W., "Flux Split Algorithms for the Multi-Dimensional Euler Equations with Real Gases," *Computers and Fluids*, vol. 17, no. 1, 1989.

[15] Grossman, B. and Walters, R. W., "Analysis of Flux-Split Algorithms for Euler's Equations with Real Gases," *AIAA Journal*, vol. 27, no. 5, 1989.

[16] Liu, Y. and Vinokur, M., "Nonequilibrium Flow Computations I. An Analysis of Numerical Formulations of Conservation Laws," NASA Contractor Report 177489, 1998.

[17] Walters, R. W., Cinnella, P., and Slack, D. C., "Characteristic-Based Algorithms for Flows in Thermo-Chemical Nonequilibrium," AIAA 90-0393, Jan. 1990.

[18] Candler, G. V. and MacCormack, R. W., "The Computation of Hypersonic Ionized Flows in Chemical and Thermal Nonequilibrium," AIAA 88-0511, Jan. 1988.

[19] Grossman, B. and Cinnella, P., "The Development of Flux-Split Algorithms for Flows with Non-Equilibrium Thermodynamics and Chemical Reactions," AIAA 88-3595-CP, 1988.

[20] Grossman, B. and Cinnella, P., "The Computation of Non-Equilibrium Chemically-Reacting Flows," *Computers and Structures* , vol. 30, no. 1/2, 1988.

[21] Grossman, B. and Cinnella, P., "Upwind Methods for Flows with Non-Equilibrium Chemistry and Thermodynamics," in *Third International Conference on Numerical Combustion*, (Antibes), 1989.

[22] Grossman, B., Cinnella, P., and Garrett, J., "A Survey of Upwind Methods for Flows with Equilibrium and Non-Equilibrium Chemistry and Thermodynamics," AIAA 89-1653, 1989.

[23] Batina, J. T., "Three-Dimensional Flux-Split Euler Schemes Involving Unstructured Meshes," AIAA 90-1649, Jan. 1990.

[24] Barth, T. J. and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA 89-0366, Jan. 1989.

[25] Barth, T. J., "A 3-D Upwind Euler Solver for Unstructured Meshes," AIAA 91-1548-CP, 1991.

[26] Mitchell, C. R., *High Resolution Algorithms for the Navier Stokes Equations for Generalized Discretizations.* PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, Dec. 1992.

[27] Van Leer, B., "Towards the ultimate conservative difference scheme. IV. A second order sequel to Godunov's method," *Journal of Computational Physics*, vol. 23, pp. 101–136, 1979.

[28] Thomas, J. L. and Walters, R. W., "Upwind Relaxation Algorithms for the Navier Stokes Equations," AIAA 85-1501, July 1985.

[29] Thomas, J. L., Van Leer, B., and Walters, R. W., "Implicit Flux-Split Schemes for the Euler Equations," AIAA 85-1680, July 1985.

[30] Frink, N. T., Parikh, P., and S., P., "A Fast Upwind Solver for the Euler Equations on Three Dimensional Unstructured Meshes," AIAA 91-0102, Jan. 1991.

[31] Barth, T. J. and Frederickson, P. O., "Higher Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction," AIAA 90-0013, Jan. 1990.

[32] Barth, T. J., "Recent Developments in High Order K-Exact Reconstruction on Unstructured Meshes," AIAA 93-0668, Jan. 1993.

[33] Mitchell, C. R. and Walters, R. W., "K-Exact Reconstruction for the Navier Stokes Equations on Arbitrary Grids," AIAA 93-0536, Jan. 1993.

[34] Barth, T. J., "Numerical Aspects of Computing Viscous High Reynolds Number Flows on Unstructured Meshes," AIAA 91-0721, Jan. 1991.

[35] Frink, N. T., "Recent Progress Toward a Three-Dimensional Unstructured Navier-Stokes Flow Solver," AIAA 94-0061, Jan. 1994.

[36] Schetz, J. A., *Foundations of Boundary Layer Theory for Momentum, Heat, and Mass Transfer.* Prentice-Hall, 1984. ISBN 0-13-329334-4.

[37] Wilcox, D. C., *Turbulence Modeling for CFD.* DCW Industries, Inc., 1993. ISBN 0-9636051-0-0.

[38] Coakley, T. J., "Turbulence Modeling Methods for the Compressible Navier-Stokes Equations," AIAA 83-1693, July 1983.

[39] Rumsey, C. L., "A Comparison of the Predictive Capabilities of Several Turbulence Models Using Upwind and Central-Difference Computer Codes," AIAA 93-0192, Jan. 1993.

[40] Baldwin, B. S. and Barth, T. J., "A One-Equation Turbulence Transport Model for High-Reynolds Number Wall-Bounded Flows," AIAA-91-0610, Jan. 1991.

[41] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence model for Aerodynamic Flows," AIAA 92-0439, Jan. 1992.

[42] Spalart, P. R. and Allmaras, S. R., "A one-equation turbulence model for aerodynamic flows," *La Recherche Aérospatiale*, vol. 1, pp. 5–21, 1994.

[43] McBride, B. J., Heimel, S., Ehlers, J. G., and S., G., "Thermodynamic Properties to 6000K for 210 Substances Involving the First 18 Elements," NASA SP 3001, 1963.

[44] Burden, R. L. and Faires, D. J., *Numerical Analysis.* PWS-Kent Publishing Company, 1989. ISBN 0-53491-585-X.

[45] Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal of Scientific Statistical Computing*, vol. 7, no. 3, 1986.

[46] Vincenti, W. G. and Kruger, C. H., *Introduction to Physical Gas Dynamics.* Robert E. Krieger, 1965. ISBN 0-88275-309-6.

[47] Anderson Jr., J. D., *Hypersonic and High Temperature Gas Dynamics.* McGraw-Hill, 1989. ISBN 0-07-001671-2.

[48] Park, C., "On Convergence of Computation of Chemically Reacting Flows," AIAA 85-0247, 1985.

[49] Millikan, R. C. and White, D. R., "Systematics of Vibrational Relaxation," *Journal of Chemical Physics*, vol. 39, no. 12, p. 3209, 1963.

[50] Blottner, F. G., Johnson, M., and Ellis, M., "Chemically Reacting Viscous Flow Program for Multi-Component Gas Mixtures," Sandia Laboratories, SC-RR-70-754, 1971.

[51] Wilke, C. R., "A Viscosity Equation for Gas Mixtures," *The Journal of Chemical Physics*, vol. 18, no. 4, 1950.

[52] Mavriplis, D. and Jameson, A., "Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes," ICASE Contractor Report 87-53, July 1987.

[53] Mavriplis, D. and Jameson, A., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," AIAA 87-0353, Jan. 1987.

[54] Anderson, W. K., *Implicit Multigrid Algorithms for the Three-Dimensional Flux Split Euler Equations.* PhD thesis, Mississippi State University, Mississippi State, Mississippi, Aug. 1986.

[55] Brandt, A., "Introductory Remarks on Multigrid Methods," tech. rep., The Weizmann Institute of Science, Department of Applied Mathematics, Rehovot, Isreal".

[56] Wigton, L. B., Yu, N. J., and Young, D. P., "GMRES Acceleration of Computational Fluid Dynamics Codes," AIAA 85-1494, 1985.

[57] Hixon, R. and Sankar, L. N., "Application of a Generalized Minimal Residual Method to 2D Unsteady Flows," AIAA 92-0422, Jan. 1992.

[58] McGrory, W. D., *Generalized Spatial Discretization Techniques for Space-Marching Algorithms.* PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, Jan. 1991.

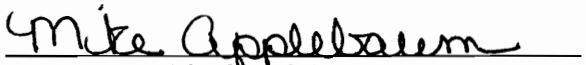
[59] Townsend, J. C., Howell, D. T., Collins, I. K., and Hayes, C., "Surface Pressure Data on a Series of Analytic Forebodies at Mach Numbers from 1.70 to 4.50 and Combined Angles of Attack and Sideslip," NASA TM-80062, June 1979.

[60] Kang, S. W. and Dunn, M. G., "Theoretical and Measured Electron-Density Distributions for the RAM Vehicle at High Altitudes," AIAA 72-689, 1972.

[61] Papay, M. L., *A General Reverse Design Procedure for Aerodynamic Bodies.* PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, May 1994.

Vitae

The author was born on September 8th, 1967 in Rockville, Maryland. He began his engineering studies at Embry Riddle Aeronautical University and transferred to the Aerospace Engineering program at Virginia Polytechnic Institute and State University. He received a Bachelor of Science in Aerospace Engineering, graduating Magna Cum Laude in May 1990. Upon completion of his Bachelor of Science degree, he entered the graduate program at Virginia Polytechnic Institute and State University. The next four years were spent in pursuit of a Doctorate in Aerospace Engineering.



Michael Paul Applebaum