REACTING PLUME INVERSION ON URBAN GEOMETRIES THROUGH GRADIENT BASED DESIGN METHODOLOGIES

By

Nicholas G. Currier

REACTING PLUME INVERSION ON URBAN GEOMETRIES THROUGH GRADIENT BASED DESIGN METHODOLOGIES

By

Nicholas G. Currier

A Dissertation Submitted to the Faculty of the University of
Tennessee at Chattanooga in Partial Fulfillment of the
Requirements of the Degree of Doctor of Philosophy
in Computational Engineering

The University of Tennessee at Chattanooga
Chattanooga, Tennessee

August 2014

ABSTRACT

An increased focus on domestic security in recent years has brought attention to several important application areas where computational fluid dynamics (CFD) has the ability to make a significant impact. In particular, disaster mitigation and post-event forensic activities are of interest. This work investigates a procedure built on gradient based design methods to allow for the solution of the so-called inverse chemistry problem in urban environments. The inverse chemistry problem consists of computing a release location based on the sensing of chemical byproducts of the release and the ability to compute an accurate flow field on the geometry of interest. In this study, Washington DC is simulated under conditions of a hazardous plume. A CFD solver is implemented which allows for the solution of the preconditioned finite-rate Navier-Stokes equations as well as the *in situ* computation of design gradients.

DEDICATION

I would like to dedicate this dissertation to my beautiful wife, Maria. Without her unwavering support and sacrifice this document might never have come to fruition.

ACKNOWLEDGMENTS

A very sincere thanks goes to the acting adviser for a majority of this work, Dr. Daniel Hyams. His mentorship, guidance, and unending patience will forever be deeply appreciated. Without his help, this dissertation would not have been possible.

I would also like to thank Dr. Jim Newman, III for agreeing to take on a new student in need of an adviser on such short notice. The well of knowledge available through him has been instrumental in my success. He has also made the transition to new advisement easy and for this I wish to express my gratitude.

I wish to express my thanks to Dr. Sreenivas as well for his willingness to aid my project at a moments notice. I also thank my other committee members, Dr. Matt Matthews, Dr. Kyle Anderson, and Dr. Robert Webster for their guidance regarding this work. Dr. Tim Swafford deserves my appreciation for his guidance and good nature as department head, interim director, and mentor. Despite all of the demands we students make of him, he has always been a jovial and grounded individual with good advice and an open door.

Last but certainly not least I wish to express my heartfelt gratitude towards my colleagues and friends at the SimCenter during my time here. Cameron Druyor, Wally Edmondson, Ethan Hereth, Bruce Hilbert, Anshul Mittal, Matthew O'Connell, and Scott Richardson deserve special mention. You have all been incredibly supportive and instrumental in my success here. Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

$\alpha_{r,k}$	collisional efficiency for species k and reaction r
β	preconditioner parameter
Δq_i^n	nonconservative variable update for cell i and time level n
Δt	time-step
δ	variable jump vector
$\dot{W}$	chemical source term vector
$\dot{w}_i$	mass production rate for species i
Γ	effective concentration
γ	specific heat ratio
$\hat{n}_x, \hat{n}_y, \hat{n}_z$	face area vector components in x, y, and z directions
$\hat{\vec{n}}$	normalized face area vector
$\hat{q}$	heat flux
$\int_\Omega$	control volume integral
$\int_{\partial\Omega}$	control volume surface integral

λ	eigenvalue
$\mathcal{V}$	volume
μ	molecular viscosity
μ_t	turbulent viscosity
$\nu''_{i,r}$	product stoichiometric coefficient for reaction r and species i
$\nu'_{i,r}$	reactant stoichiometric coefficient for reaction r and species i
$\Re$	residual
ρ	mixture density
ρ_i	density of species i
θ	covariant velocity, $\hat{n}_x u + \hat{n}_y v + \hat{n}_z w + a_t$
$\tilde{k}$	local heat transfer coefficient
$\vec{n}$	control volume face area vector
$\vec{V}$	control volume face velocity vector, $V_x\hat{i} + V_y\hat{j} + V_z\hat{k}$
ξ_1, ξ_2	temporal accuracy parameters
$a_1, a_2, a_3, a_4, a_5, a_6$	thermochemistry curvefit coefficients
a_t	control volume face velocity in direction of normal
B_g	buoyancy source term vector

c_L, c_R	speed of sound to the left and right of a face
c_p	specific heat at constant pressure
c_v	specific heat at constant volume
D_T	thermal diffusion coefficient
D_{ij}	binary diffusion coefficient for species pair i and j
Da	local Damköhler number
e_t	specific total energy
Ea	activation energy
F	conservative inviscid flux vector
G	conservative viscous flux vector
g_x, g_y, g_z	gravity force components
H	unsteady residual
h	specific enthalpy
h_i^o	standard state enthalpy for species i
h_t	specific total enthalpy
J_e	energy diffusion flux
J_i	mass diffusion flux for species i

$K_{b,r}$	backward reaction rate for reaction r
$K_{c,r}$	equilibrium rate constant for reaction r
$K_{f,r}$	forward reaction rate for reaction r
M_k	molecular weight of species k
NS	number of species
p	pressure
P_p	preconditioner for pressure based nonconservative variable vector
p_{std}	standard pressure = 1 atm
Pr_t	turbulent Prandtl number
Q	conservative variable vector
q	nonconservative variable vector
R	specific mixture gas constant
R_i	specific gas constant
R_{univ}	universal gas constant
s	specific entropy
s_i^o	standard state entropy for species i
S_L, S_R, S_*	wave speed estimates

T	temperature
u, v, w	flow field velocity components in x, y, and z directions
U_i	diffusion velocity for species i
V_x, V_y, V_z	grid velocities in x, y, and z directions
x, y, z	Cartesian coordinate directions
x_i	mole fraction for species i
Y_i	mass fraction
z''	sum of product stoichiometric coefficients
z'	sum of reactant stoichiometric coefficients

CHAPTER 1

INTRODUCTION

An increased focus on domestic security in recent years has brought attention to several important application areas where computational fluid dynamics (CFD) has the ability to make a significant and lasting impact. In particular, disaster mitigation and post-event forensic activities are of interest. An engineering solution to the so-called "inverse chemistry" problem has the potential to aid in forensic activities related to plume dispersal in high occupancy areas.

Unfortunately, these incidents, whether intentional or accidental are not at all uncommon. In 1994 over 600 individuals were exposed to Sarin gas when a truck with a heater and a fan was used as a delivery mechanism in a residential area of Matsumoto, Japan [1]. Vapors from Sarin-filled containers in a Tokyo, Japan subway exposed several thousand in 1995 [2]. The same agent was used in Damascus, Syria, in 2013 during civil war. Chlorine and other industrial chemicals are also of grave concern. In 2005, 90 tons of chlorine gas was accidentally vented into Graniteville, South Carolina following a train derailment with mass injury occurring [3]. Advanced simulation and in particular the ability to study complicated releases with precise outcomes, i.e. design, promises to increase the knowledge-base surrounding these tragedies and assist in enhanced informed planning on the part of security personnel. Allocation of resources to better protect currently unidentified critical locations could have profound effects in preventing or mitigating disaster situations.

The technologies necessary for simulation of chemical dispersal in an urban environment have been present in literature for some time. These include preconditioning for low-Mach flow present in atmospheric plumes, finite-rate chemistry for tracking chemical content with reactions, and parallel computational techniques necessary for the resolution requirements of complicated urban terrain.

Given a forward solution (or physical measurement) from a disaster situation which provides fixed sensor data for measurable chemical species, an engineering estimate with useful error on a dispersant location is desired. Sensor data provided for a particular release will be dependent on release concentration, local weather (flow field), release duration, etc. These variables are natural choices for study.

Several researchers have investigated this problem previously. Markov chain Monte-Carlo (MCMC) methods have been used to determine source plume locations in complex urban environments under uncertainty [4]. A recommended resource for MCMC methods is given in [5]. However, these methods require a very high number of forward simulations to build a statistical database. The methods rely on the forward simulation of a flow field with chemistry changes absent and the solution of a scalar transport equation with the flow field imposed. Current research has therefore been commonly limited to convective-diffusion equations without finite-rate chemistry [6]. Greedy hidden Markov models have also been used to estimate both the plume source as well as source parameters based on binary sensor output [7]. Huang *et al* [8] studied parameter estimation using PDE based plume models and MCMC methods. Their results indicate that position of a release can be determined to useful levels using convection-diffusion models. Primary disadvantages of these methods are the relative inflexibility and lack of fidelity to resolve flow features in complicated environments. While the addition of source terms to account for terrain steering effects is possible in theory, it appears to be cumbersome and error-prone in general. Michaelides and Panayiotou utilize a nonlinear least squares method to perform source inversion [9]. This work uses a constant radial concentration model with varying noise that does not model the effects of convection. However, the model does represent diffusion in some sense. Interestingly, utilizing the nonlinear least squares procedure under low levels of temporally varying noise, increasing the number of sensors does not always increase the source location predictions. Errors reported are on the order of 50-100 meters for a 1 km by 1 km solution space with less than 100 sensors deployed.

The present research proposes to utilize gradient based design methods to provide an engineering "best-guess" for plume source location with in-plume reactivity. The method has the potential to be significantly more accurate and flexible than building the MCMC databases which are considered state of the art at the time of this writing. Because the gradient based method requires a full simulation of potential source locations, it also permits much higher resolution models to be utilized. The work of Chow [4] requires nearly 2,560 forward simulations in an Oklahoma City model environment. For a simple geometry, a single building in crossflow, the source location was predicted very well. However, for a complex urban environment such as Oklahoma City, the source location was localized to within around 100 meters. Clearly this is a difficult problem with the complexities of real geometry. For a total geometry size of 300 meters by 330 meters with a total height of around 100 meters (580,000 elements), the construction of the flow field database took approximately 17 days on 32 2.4 GHz Xeon processors [4]. Computationally, the above method is intractable with finite-rate chemistry on adequately resolved meshes. Hence, the work presented here demonstrates an increase in the fidelity of predictions in more complicated urban environments. More importantly, the methods presented herein only require the simulation of potential source locations as the optimization problem progresses. This has the potential to be significantly cheaper for a given problem. However, querying the database given in Chow for given sensor data only takes approximately 5 minutes on two processors [4]. The method presented here does not have the potential to demonstrate a turn around time this short because computation is not started until sensor data is acquired. However, a similar database could be constructed with this method for traversal. This response time optimization is not performed here.

Gradient based design optimization methodologies have been shown to be quite robust in applications of fairly complex engineering problems [10]. Both the direct and adjoint methods show the ability to compute high accuracy gradients of output functions with respect to parametric variables from physics-based solvers. It has been shown that these methods produce derivatives comparable to computing flow solutions in complex arithmetic at a fraction of the cost. These methods are of particular interest here because of their increased accuracy when compared to finite difference methods external to the flow solver. Due to the low concentrations as well as the low speed flow in urban environments the problem of interest would be intractable in the presence of accuracy limitations. That is, the inverse problem has high sensitivity to gradient accuracy due to the difficulty in solving the forward problem (CFD).

In this work, a fully parallel compressible Navier-Stokes solver is developed. The desired capabilities include the simulation of low speed reacting plumes in complicated urban environments. However, this work also seeks to develop and demonstrate the ability to compute accurate and cost efficient sensitivity information of a sensor field sample with respect to a dispersal location. Several methods of computing these gradients are compared and their accuracy evaluated for numerous test problems.

Thus, the goal of this study is to provide a proof of concept study demonstrating the appropriate technologies required to perform forensic investigation of a hazardous release. While the complete "inverse chemistry" problem is not attempted on a fully resolved urban geometry, a surrogate test case investigates the feasibility of using these methods. Results of forward simulations without sensitivity are also presented on a full geometry of Washington DC. Though computational power has continued to increase exponentially, the resources available to this researcher are still roughly an order of magnitude too low for reasonable use of these techniques on real geometries. However, machines are in existence with the throughput to investigate disasters on useful time scales.

CHAPTER 2

COMPUTATIONAL FORMULATION

This chapter describes the governing equations of interest and the numerical discretization used in their solution. Within this discussion the preconditioning required for the solution of reacting flows at low Mach numbers is presented. Additionally discretization in time and space, as well as practical implementation issues, are described in detail.

2.1 Governing Equations

The governing equations of interest in the present work are those representing time-dependent, compressible, turbulent, chemically reacting flows. In this regard, the Reynolds averaged Navier-Stokes equations utilizing finite-rate chemistry are solved with turbulence being modeled in a loosely coupled manner.

2.1.1 Representation of Flow Variables

Several variable sets are considered throughout this work. In many cases a particular choice makes certain calculations considerably less difficult. The variable sets used here are

$$Q = [\rho_1 \; ... \; \rho_{NS} \; \rho u \; \rho v \; \rho w \; \rho e_t]^T \tag{2.1}$$

where Q is the conservative variable set of species density (ρ_i), momentum ($\rho u, \rho v, \rho w$), and total energy (ρe_t). Two nonconservative variables sets are also used. These are constructed with either pressure (p), or temperature (T) representing the energy content of the flow and both replacing momentum with flow velocity (u, v, w). These are

$$q_p = [\rho_1 \, ... \, \rho_{NS} \; u \; v \; w \; p]^T \tag{2.2}$$

$$q_T = [\rho_1 \, ... \, \rho_{NS} \; u \; v \; w \; T]^T \tag{2.3}$$

Conversion between any particular set may be accomplished via transformation Jacobians. For example the map from pressure based to temperature based variables is performed by

$$dq_T = \left[\frac{\partial q_T}{\partial q_p}\right] dq_p \tag{2.4}$$

2.1.2 Compressible Navier-Stokes with Finite-Rate Chemistry

The Navier-Stokes equations with finite-rate chemistry are written as

$$\frac{\partial}{\partial t}\int_{\Omega} Q \, d\mathcal{V} + \int_{\partial\Omega} F \cdot \hat{\vec{n}} \, dA = \frac{1}{Re}\int_{\partial\Omega} G \cdot \hat{\vec{n}} \, dA + \int_{\Omega} \dot{W} d\mathcal{V} + \int_{\Omega} B_g \, d\mathcal{V} \tag{2.5}$$

$$Q = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_{NS} \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix}, \dot{W} = \begin{bmatrix} \dot{w}_1 \\ \dot{w}_2 \\ \vdots \\ \dot{w}_{NS} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, B_g = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ g_x(\rho - \rho_{ref}) \\ g_y(\rho - \rho_{ref}) \\ g_z(\rho - \rho_{ref}) \\ 0 \end{bmatrix} \tag{2.6}$$

where Q is the vector of conservative variables, $\dot{W}$ is the chemical source term associated with the mass production for each tracked species, and B_g is the gravity buoyancy source which will be subsequently discussed.

The inviscid and viscous fluxes may be expressed as

$$F \cdot \hat{\vec{n}} = \begin{bmatrix} \rho_1(u - V_x) \\ \rho_2(u - V_x) \\ \vdots \\ \rho_{NS}(u - V_x) \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ \rho h_t(u - V_x) + pV_x \end{bmatrix} \hat{i} + \begin{bmatrix} \rho_1(v - V_y) \\ \rho_2(v - V_y) \\ \vdots \\ \rho_{NS}(v - V_y) \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ \rho h_t(v - V_y) + pV_y \end{bmatrix} \hat{j} + \begin{bmatrix} \rho_1(w - V_z) \\ \rho_2(w - V_z) \\ \vdots \\ \rho_{NS}(w - V_z) \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ \rho h_t(w - V_z) + pV_z \end{bmatrix} \hat{k} \tag{2.7}$$

$$G \cdot \hat{\vec{n}} = \begin{bmatrix} J_{1,x} \\ \vdots \\ J_{NS,x} \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - \hat{q}_x + J_e \end{bmatrix} \hat{i} + \begin{bmatrix} J_{1,y} \\ \vdots \\ J_{NS,y} \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{yx} + v\tau_{yy} + w\tau_{yz} - \hat{q}_y + J_e \end{bmatrix} \hat{j} + \begin{bmatrix} J_{1,z} \\ \vdots \\ J_{NS,z} \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ u\tau_{zx} + v\tau_{zy} + w\tau_{zz} - \hat{q}_z + J_e \end{bmatrix} \hat{k} \quad (2.8)$$

where τ is the shear stress tensor with

$$\tau_{xx} = (\mu + \mu_t)\left(2u_x - \frac{2}{3}\mathcal{D}\right) \quad (2.9)$$

$$\tau_{yy} = (\mu + \mu_t)\left(2v_y - \frac{2}{3}\mathcal{D}\right) \quad (2.10)$$

$$\tau_{zz} = (\mu + \mu_t)\left(2w_z - \frac{2}{3}\mathcal{D}\right) \quad (2.11)$$

$$\tau_{xy} = \tau_{yx} = (\mu + \mu_t)(u_y + v_x) \quad (2.12)$$

$$\tau_{xz} = \tau_{zx} = (\mu + \mu_t)(u_z + w_x) \quad (2.13)$$

$$\tau_{yz} = \tau_{zy} = (\mu + \mu_t)(v_z + w_y) \quad (2.14)$$

$$\mathcal{D} = u_x + v_y + w_z \quad (2.15)$$

where molecular viscosity is defined by μ and the turbulent eddy viscosity by μ_t. $\hat{q}$ represents the heat flux defined as

$$\hat{q} = -\left(\tilde{k} + \frac{c_p \mu_t}{Pr_t}\right)\nabla T \quad (2.16)$$

and $\tilde{k}$ is a local heat transfer coefficient. J_i is a vector of diffusion fluxes for species i.

An additional term, accounting for the energy transfer must be included in the diffusion flux terms, and is given by

$$J_e = \sum_{i=1}^{NS} \rho_i h_i U_i \tag{2.17}$$

where U_i is the species diffusion velocity. The above equation set is valid for a Newtonian fluid in local thermodynamic equilibrium obeying Stokes' hypothesis.

2.1.3 Buoyancy Source Terms

Of particular interest in this study is the simulation of heavy gas plumes. Sinking of a heavy gas plume to street level, and the continued propagation in the boundary layer region of the mesh, are critical for accurately modeling this physical phenomena. Without resolving this effect, all plumes, regardless of density, rise until the vertical momentum imparted is negated by cross flow and continue to propagate at a relatively stable height over the environment. In the presence of gravity source terms, however, the simulated path of contaminants is more physically representative and complex. These effects have been shown to be fundamental to CFD modeling of heavy gases [11, 12].

In the current implementation a buoyancy source term is applied to the momentum equations as follows

$$B_g = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ g_x(\rho - \rho_{ref}) \\ g_y(\rho - \rho_{ref}) \\ g_z(\rho - \rho_{ref}) \\ 0 \end{bmatrix} \tag{2.18}$$

where ρ_{ref} is a reference value associated with the bulk density with the purpose of improving the numerics of the source term addition. This factor is not strictly necessary but allows the value added into the source term to consider only the difference between cells rather than full magnitude. Because the buoyancy source is integrated across all cells in the mesh, ρ_{ref} is simply a constant in the numerical integration. The gravity vector, $\vec{g}$, is specified based on the grid orientation and ρ is the bulk fluid density in the control volume.

2.1.4 Chemical Source Terms

The chemical source term for each species in the Equation 2.5 is defined as a combination of the rate contributions from each reaction in which that species participates. It is written as

$$\dot{w}_i = M_i \sum_{r=1}^{NR} (\nu''_{i,r} - \nu'_{i,r}) \Gamma \left[K_{f,r} \prod_{k=1}^{NS} \left(\frac{\rho_k}{M_k} \right)^{\nu'_{k,r}} - K_{b,r} \prod_{k=1}^{NS} \left(\frac{\rho_k}{M_k} \right)^{\nu''_{k,r}} \right] \tag{2.19}$$

where $M_{i/k}$ is the molecular weight of each species indexed either i or k. The terms $\nu'_{i/k,r}$ and $\nu''_{i/k,r}$ are the left and right-hand chemical stoichiometric coefficients respectively for a species indexed either i or k. The coefficient $K_{f,r}$ is the forward rate of a particular reaction and $K_{b,r}$ is the backward rate. It is worth noting that $K_{f,r}$ and $K_{b,r}$ have units that are dependent on the stoichiometric coefficients. That is, $K_{f,r}$ has units of $(\frac{mol}{m^3})^{1-z'}\frac{1}{s}$ and $K_{b,r}$ has units of $(\frac{mol}{m^3})^{1-z''}\frac{1}{s}$, where z' is the sum of the left-hand side coefficients and z'' is the sum of the right-hand side coefficients for a particular reaction. The term Γ is used as a modifier for the rate of progress of a reaction if a third body is present. A third body is any species which has the potential to modify the reaction rate either as a catalyst or through rate reduction. It is considered an effective concentration of a species for cases where there is enhanced collisional efficiency $\alpha_{r,k}$ [13]. This effective concentration can be higher than the actual concentration. If all species contribute equally to a reaction, $\alpha_{r,k} = 1$. This term can be computed via the expression

$$\Gamma = \sum_{k=1}^{NS} \alpha_{r,k} \frac{\rho_k}{M_k} \tag{2.20}$$

The backward rate of a reaction is typically determined from the combination of a forward rate ($K_{f,r}$), which is computed as a function of temperature, and the equilibrium constant ($K_{c,r}$), which is computed via standard thermodynamic properties of the involved reactants and products. A full description can be found in Section 2.14.3.

2.1.5 Diffusion Flux Calculation

The Stefan-Maxwell equations are a model for describing the inter-species diffusion in a multicomponent gas mixture. Following Ramshaw [14] these equations are

$$\sum_j \frac{x_i x_j}{D_{ij}}[U_j - U_i] = G_i \tag{2.21}$$

where x_i is the mole fraction of species i, D_{ij} is the binary diffusion coefficient for the species pair i and j. The driving forces G_i are given as

$$G_i = \nabla x_i + (x_i - Y_i)\nabla ln\ p + K_i \nabla ln\ T - \frac{1}{p}[\rho_i F_i - Y_i \rho_j F_j] \tag{2.22}$$

where p is pressure, T is temperature, and F_i is the body force per unit mass acting on species i. The mass fraction for species i is given by Y_i. K_i is a function of species thermal diffusion coefficients and species mass/mole fractions. It is given by

$$K_i = \sum_j \frac{x_i x_j}{\rho D_{ij}} \left(\frac{D_{T,i}}{Y_i} - \frac{D_{T,j}}{Y_j} \right) \tag{2.23}$$

where $D_{T,i}$ are the thermal diffusion coefficients. The above equations are referred to as the generalized Stefan-Maxwell equations. In the case where the only driving addition to G_i is the mole-fraction gradients, ∇x_i, the Stefan-Maxwell equations are recovered [15]. This assumption is made here. The summation of G_i over all species is equal to zero. That is,

$$\sum_i G_i = 0 \tag{2.24}$$

The mass diffusion flux for species i is thus defined as

$$J_i = \rho Y_i U_i \tag{2.25}$$

which allows closure for the mass diffusion fluxes given in Equation 2.5.

It should be noted that though the above equation is written in terms of NS components, there are only $NS - 1$ independent equations. These equations also reduce to Fick's Law in the case of two species diffusion only.

While not developed here, an excellent discussion of several methods for solving the reduced form of the Stefan-Maxwell equations under varying assumptions is given by Subramaniam [16].

2.2 Preconditioner for Low Mach Flows

The numerical difficulty of solving the compressible Navier-Stokes equations at low Mach numbers arises from the spectral radius of the discretized linear system becoming large. This manifests itself as numerical "stiffness" and slow convergence properties. More nefarious is that the convergence of iterative methods used to solve the linear system is strictly dependent on the spectral radius of the iteration matrix being less than unity. For example, the left hand side of the linear system, A, can be written as

$$A = D + R \tag{2.26}$$

where D is the diagonal of the matrix and R is the remainder. The Jacobi method requires the following condition for convergence

$$\rho(D^{-1}R) < 1 \tag{2.27}$$

where D^{-1} is the inverse of the diagonal. The results are similar for other iterative solution methods such as Symmetric Gauss Seidel [17].

The eigenvalues of concern correspond to the convective and acoustic wave speeds. As the Mach number of the flow is decreased towards the limit of zero, the eigenvalue associated with the acoustic wave speed changes little. The convective eigenvalue, however, approaches zero. This large disparity causes the spread in eigenvalues described above. For the non-preconditioned equations, these eigenvalues are

$$\begin{aligned} \lambda_{conv} &= \theta \\ \lambda_{acoustic} &= \theta + c \end{aligned} \tag{2.28}$$

where θ is the velocity normal to a face and c is a mixture speed of sound.

A method is developed to "condition" the equations and in turn, reduce the spectral radius. This allows for the convergence of the discretized partial differential equations in the low Mach number limit.

2.3 Preconditioner for Finite-Rate Navier-Stokes

Following Gupta, the energy conservation equation is written for a given species i in terms of entropy for fluid undergoing compression [18]. The equation is written under the assumption of constant entropy. The work that follows is a multi-species extension to Eriksson's constant entropy preconditioner [19].

$$dQ_{i,rev} = Tds_i = de_i + p_i dv_i \tag{2.29}$$

Specific volume v_i is related to density

$$v_i = \frac{1}{\rho_i} \tag{2.30}$$

and the differential of v is then

$$dv_i = -\frac{1}{\rho_i^2} d\rho_i \tag{2.31}$$

Substituting Equation 2.31 into Equation 2.29 results in

$$Tds_i = de_i - \frac{p_i}{\rho_i^2} d\rho_i \tag{2.32}$$

The total derivative of internal energy with respect to the two state variables $e_i = e_i(\rho_i, T)$ is given by

$$de_i = \left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} dT + \left.\frac{\partial e_i}{\partial \rho_i}\right|_{p} d\rho_i \tag{2.33}$$

Temperature can be expressed as a function of pressure and density while holding other species densities constant $T = T(p, \rho_i)$, which leads to the differential

$$dT = \left.\frac{\partial T}{\partial p_i}\right|_{\rho_i} dp_i + \left.\frac{\partial T}{\partial \rho_i}\right|_{p} d\rho_i \tag{2.34}$$

Equation 2.34 is substituted into Equation 2.33 and the result into Equation 2.32 yield

$$\begin{aligned}
Tds_i &= \left.\tfrac{\partial e_i}{\partial T}\right|_{\rho_i} dT + \left.\tfrac{\partial e_i}{\partial \rho_i}\right|_{p} d\rho_i - \tfrac{p_i}{\rho_i^2} d\rho_i \\
Tds_i &= \left.\tfrac{\partial e_i}{\partial T}\right|_{\rho_i} \left[\left.\tfrac{\partial T}{\partial p_i}\right|_{\rho_i} dp_i + \left.\tfrac{\partial T}{\partial \rho_i}\right|_{p} d\rho_i \right] + \left.\tfrac{\partial T}{\partial \rho_i}\right|_{p} d\rho_i + \tfrac{p_i}{\rho_i^2} d\rho_i \\
Tds_i &= \left.\tfrac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\tfrac{\partial T}{\partial p_i}\right|_{\rho_i} dp_i + \left[\left.\tfrac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\tfrac{\partial T}{\partial \rho_i}\right|_{p} + \left.\tfrac{\partial e_i}{\partial \rho_i}\right|_{T} - \tfrac{p_i}{\rho_i^2} \right] d\rho_i
\end{aligned} \tag{2.35}$$

A parameter β is introduced to scale the changes in pressure and limit the spectral radius of the flux Jacobian as

$$dp_i^* = \beta dp_i \tag{2.36}$$

In this work β is a global constant and is chosen to be

$$\beta = \begin{cases} \text{Mach}^2 & \text{for Mach} < 1 \\ 1 & \text{for Mach} \geq 1 \end{cases} \tag{2.37}$$

This choice provides well behaved eigenvalues for the test cases examined. The terms dp_i and $d\rho_i$ are replaced with the conditioned dp_i^* and $d\rho_i^*$ in Equation 2.35 to arrive at

$$Tds_i = \left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial p_i}\right|_{\rho_i} dp_i^* + \left[\left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial \rho_i}\right|_{p} + \left.\frac{\partial e_i}{\partial \rho_i}\right|_{T} - \frac{p_i}{\rho_i^2} \right] d\rho_i^* \tag{2.38}$$

Subtracting Equation 2.38 from Equation 2.35 and enforcing zero species entropy change Tds_i between the conditioned and unconditioned expressions results in

$$0 = \left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial p_i}\right|_{\rho_i} (dp_i - dp_i^*) + \left[\left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial \rho_i}\right|_{p} + \left.\frac{\partial e_i}{\partial \rho_i}\right|_{T} - \frac{p_i}{\rho_i^2} \right] (d\rho_i - d\rho_i^*) \tag{2.39}$$

Substituting Equation 2.36 into Equation 2.38 and solving for the conditioned change in species density ρ_i^* defines

$$d\rho_i^* = d\rho_i + \frac{\left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial p_i}\right|_{\rho_i} (1-\beta) dp_i}{\left[\left.\frac{\partial e_i}{\partial T}\right|_{\rho_i} \left.\frac{\partial T}{\partial \rho_i}\right|_{p} + \left.\frac{\partial e_i}{\partial \rho_i}\right|_{T} - \frac{p_i}{\rho_i^2} \right]} \tag{2.40}$$

Making judicious use of the cyclic rule of partial derivatives, this can be rewritten as

$$d\rho_i^* = d\rho_i - \frac{(1-\beta)}{c_i^2} dp_i \tag{2.41}$$

where c_i^2 is the square of the species speed of sound. This is calculated for ideal gases as

$$c_i^2 = \gamma_i R_i T \tag{2.42}$$

where γ_i is the ratio of specific heats i and R_i is the specific gas constant for species i. Note that the change in preconditioned density is related to the change in partial pressure where the total pressure can be written as

$$p_i = \frac{\rho_i}{\rho} p \tag{2.43}$$

That is, the change in density related to the change in total pressure involves a mass fraction term. The exclusion of this term results in the generation of spurious mass sources and sinks. Though of small magnitude in the cases examined, these sources and sinks greatly hinder the convergence properties and are certainly non-physical in nature. Therefore, the change in species density may be written as

$$d\rho_i^* = d\rho_i - \frac{(1-\beta)}{c_i^2} Y_i dp = d\rho_i - \frac{(1-\beta)\rho_i}{c_i^2 \rho} dp \tag{2.44}$$

and in matrix form the differential changes become

$$\begin{bmatrix} d\rho_i^* \\ \vdots \\ d\rho_{NS}^* \\ du \\ dv \\ dw \\ dp^* \end{bmatrix} = \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & 0 & -\frac{(1-\beta)\rho_1}{c_1^2 \rho} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 & 0 & -\frac{(1-\beta)\rho_{NS}}{c_{NS}^2 \rho} \\ 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & \beta \end{bmatrix} \begin{bmatrix} d\rho_i \\ \vdots \\ d\rho_{NS} \\ du \\ dv \\ dw \\ dp \end{bmatrix} \tag{2.45}$$

The preconditioning matrix is derived from the transformation matrix in Equation 2.45. The above is expressed in $[\rho\ u\ p]^T$ primitive variables, and this is denoted via the subscript p as

$$P_p = \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & 0 & -\frac{(1-\beta)\rho_1}{c_1^2\rho} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 & 0 & -\frac{(1-\beta)\rho_{NS}}{c_{NS}^2\rho} \\ 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & \beta \end{bmatrix} \tag{2.46}$$

The inverse of the preconditioner is required in the implicit solution procedure, which may be shown to be

$$P_p^{-1} = \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & 0 & \frac{(1-\beta)\rho_1}{\beta c_1^2\rho} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 & 0 & \frac{(1-\beta)\rho_{NS}}{\beta c_{NS}^2\rho} \\ 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & \frac{1}{\beta} \end{bmatrix} \tag{2.47}$$

2.4 Preconditioned Eigensystem

Implementation of characteristic boundary conditions requires the eigensystem of the preconditioned equations. Additionally, this allows for the extension of the flow solver to make use of other approximate Riemann solvers which require a full eigensystem. The derivations are performed using the pressure based nonconservative variable set. This choice in particular highlights the wave speeds that should be conditioned and provides for simpler mathematical manipulation. The interested reader is directed to Gupta [18] for a parameterized formulation of many other available preconditioners. The flux Jacobian derived in the pressure based nonconserved variable set is given by

$$A_p = \frac{\partial F}{\partial q_p} = \begin{bmatrix} \theta & \cdots & 0 & \rho_1 n_x & \rho_1 n_y & \rho_1 n_z & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \theta & \rho_{NS} n_x & \rho_{NS} n_y & \rho_{NS} n_z & 0 \\ 0 & \cdots & 0 & \theta & 0 & 0 & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & 0 & \theta & 0 & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & 0 & 0 & \theta & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & \rho c^2 n_x & \rho c^2 n_y & \rho c^2 n_z & \theta \end{bmatrix} \tag{2.48}$$

Applying the preconditioner to the flux Jacobian matrix results in

$$P_pA_p = \begin{bmatrix} \theta & \cdots & 0 & \frac{\rho_1 n_x(c_1^2-c^2(1-\beta))}{c_1^2} & \frac{\rho_1 n_y(c_1^2-c^2(1-\beta))}{c_1^2} & \frac{\rho_1 n_z(c_1^2-c^2(1-\beta))}{c_1^2} & -\frac{\rho_1\theta(1-\beta)}{\rho c_1^2} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \theta & \frac{\rho_{NS} n_x(c_{NS}^2-c^2(1-\beta))}{c_{NS}^2} & \frac{\rho_{NS} n_y(c_{NS}^2-c^2(1-\beta))}{c_{NS}^2} & \frac{\rho_{NS} n_z(c_{NS}^2-c^2(1-\beta))}{c_{NS}^2} & -\frac{\rho_{NS}\theta(1-\beta)}{\rho c_{NS}^2} \\ 0 & \cdots & 0 & \theta & 0 & 0 & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & 0 & \theta & 0 & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & 0 & 0 & \theta & \frac{n_x}{\rho} \\ 0 & \cdots & 0 & \beta\rho c^2 n_x & \beta\rho c^2 n_y & \beta\rho c^2 n_z & \beta\theta \end{bmatrix} \tag{2.49}$$

Furthermore, the preconditioned flux Jacobian can be decomposed into the eigensystem

$$P_pA_p = T\Lambda T^{-1} \tag{2.50}$$

where T are the right eigenvectors, T^{-1} are the left eigenvectors, and Λ is a diagonal matrix whose entries are the eigenvalues of the system. These eigenvalues are

$$\lambda_1 = \lambda_2 = ... = \lambda_{NS+2} = \theta \tag{2.51}$$

$$\lambda_{NS+3} = \frac{1}{2}(\beta+1)\theta + c' \tag{2.52}$$

$$\lambda_{NS+4} = \frac{1}{2}(\beta+1)\theta - c' \tag{2.53}$$

The selection of the eigenvectors requires great care be taken due to the repeat $NS + 2$ eigenvalues. Construction of a basis for the preconditioned flux Jacobian which is nonsingular when inverted is nontrivial. Additionally, the eigensystem must remain in reasonable scale even in the presence of trace species with very low density [20]. That is, the matrix must not approach singularity in the presence of trace species. The right eigenvectors of this system are selected to be

$$T = \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & -\frac{\rho_1(c_1^2+c^2(\beta-1)-\theta X_m(\beta-1))}{c_1^2 X_m} & \frac{\rho_1(c_1^2+c^2(\beta-1)-\theta X_p(\beta-1))}{c_1^2 X_p} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 & -\frac{\rho_{NS}(c_{NS}^2+c^2(\beta-1)-\theta X_m(\beta-1))}{c_{NS}^2 X_m} & \frac{\rho_{NS}(c_{NS}^2+c^2(\beta-1)-\theta X_p(\beta-1))}{c_{NS}^2 X_p} \\ 0 & \cdots & 0 & l_x & m_x & n_x & -n_x \\ 0 & \cdots & 0 & l_y & m_y & n_y & -n_y \\ 0 & \cdots & 0 & l_z & m_z & n_z & -n_z \\ 0 & \cdots & 0 & 0 & 0 & -\rho X_m & \rho X_p \end{bmatrix} \tag{2.54}$$

where $\hat{\vec{l}}$ and $\hat{\vec{m}}$ are mutually perpendicular normal vectors constructed to form a basis with the face normal vector $\hat{\vec{n}}$. The nonsingular left eigenvectors are

$$T^{-1} = \begin{bmatrix} 1 & \cdots & 0 & -\frac{\tilde{K}_1(l_y m_z - l_z m_y)}{c_1^2 X_m X_p} & \frac{\tilde{K}_1(l_x m_z - l_z m_x)}{c_1^2 X_m X_p} & -\frac{\tilde{K}_1(l_x m_y - l_y m_x)}{c_1^2 X_m X_p} & \frac{\rho_1(c_1^2+c^2(\beta-1))}{\rho c_1^2 X_m X_p} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & -\frac{\tilde{K}_{NS}(l_y m_z - l_z m_y)}{c_{NS}^2 X_m X_p} & \frac{\tilde{K}_{NS}(l_x m_z - l_z m_x)}{c_{NS}^2 X_m X_p} & -\frac{\tilde{K}_{NS}(l_x m_y - l_y m_x)}{c_{NS}^2 X_m X_p} & \frac{\rho_{NS}(c_{NS}^2+c^2(\beta-1))}{\rho c_{NS}^2 X_m X_p} \\ 0 & \cdots & 0 & (m_y n_z - m_z n_y) & -(m_x n_z - m_z n_x) & (m_x n_y - m_y n_x) & 0 \\ 0 & \cdots & 0 & -(l_y n_z - l_z n_y) & (l_x n_z - l_z n_x) & -(l_x n_y - l_y n_x) & 0 \\ 0 & \cdots & 0 & \frac{X_p(l_y m_z - l_z m_y)}{2c'} & -\frac{X_p(l_x m_z - l_z m_x)}{2c'} & \frac{X_p(l_x m_y - l_y m_x)}{2c'} & \frac{1}{2\rho c'} \\ 0 & \cdots & 0 & \frac{X_m(l_y m_z - l_z m_y)}{2c'} & -\frac{X_m(l_x m_z - l_z m_x)}{2c'} & \frac{X_m(l_x m_y - l_y m_x)}{2c'} & \frac{1}{2\rho c'} \end{bmatrix} \tag{2.55}$$

where the following definitions are made

$$\tilde{K}_i = -\rho_i(c_i^2(X_m + X_p) - (\beta - 1)X_m X_p \theta + (\beta - 1)(X_m + X_p)c^2) \tag{2.56}$$

$$\beta_m = \frac{1-\beta}{2} \tag{2.57}$$

$$X_p = \theta\beta_m + c' \tag{2.58}$$

$$X_m = \theta\beta_m - c' \tag{2.59}$$

A nonsingular eigensystem for the non-preconditioned finite-rate Navier-Stokes equations is available in the literature [20, 21].

2.5 Preconditioned HLLC Flux

Nonlinear hyperbolic partial differential equations admit solutions which contain shocks and contact discontinuities. These cases can be reduced to the solution of a Riemann problem. Using the theory of characteristics several different methods have been proposed for the upwinding of fluxes in such equation sets. One such method is given by Roe [22]. However, the approximate Riemann solver of Harten, Lax, and van Leer (HLL) [23] with contact wave (HLLC) corrections is utilized in the current work due to a considerably simpler implementation. It was found by Buvaneswari [24] that the differences between the flux formulation of Roe and HLLC are almost nonexistent as applied to both equilibrium and nonequilibrium chemically active flows. The development below follows the discussion of Toro [25].

First, the maximum and minimum eigenvalues of the preconditioned system must be found by

$$\lambda_{(NS+3)R} = \theta_R + c'_R \tag{2.60}$$

$$\lambda_{(NS+4)L} = \theta_L - c'_L \tag{2.61}$$

$$\lambda_{(NS+3)} = \bar{\theta} + \bar{c}' \tag{2.62}$$

$$\lambda_{(NS+4)} = \bar{\theta} - \bar{c}' \tag{2.63}$$

where

$$\theta = \hat{n}_x u + \hat{n}_y v + \hat{n}_z w + a_t \tag{2.64}$$

and the preconditioned speed of sound is

$$c' = \sqrt{\theta^2 \beta_m^2 + \beta c^2} \tag{2.65}$$

where c'_L, c'_R are the speed of sound based on either the left or right state at the control volume face and $\bar{c}'$ is some averaged speed of sound between the two states. Possibilities for $\bar{c}'$ include either a simple arithmetic average or a Roe averaged state. The term θ has been defined previously, only here the averaged version $\bar{\theta}$ is used. Next, wave speed estimates S_L, S_R, and S_* are defined as

$$S_L = \min(\lambda_{(NS+4)L}, \lambda_{(NS+4)}) \tag{2.66}$$

$$S_R = \max(\lambda_{(NS+3)R}, \lambda_{(NS+3)}) \tag{2.67}$$

$$S_* = \frac{p_R - p_L + \rho_L\theta_L(S_L - \theta_L) - \rho_R\theta_R(S_R - \theta_R)}{\rho_L(S_L - \theta_L) - \rho_R(S_R - \theta_R)} \tag{2.68}$$

and the HLLC state is determined from

$$Q^{HLLC} = \begin{cases} Q_L & \text{if } 0 \leq S_L \\ Q_{*L} & \text{if } S_L \leq 0 \leq S_* \\ Q_{*R} & \text{if } S_* \leq 0 \leq S_R \\ Q_R & \text{if } 0 \geq S_R \end{cases} \tag{2.69}$$

where Q_L and Q_R indicate that the entire state is derived from either the left or the right side of the interface. This is the condition that arises from a supersonic flow, in which the speed of sound is not sufficient to overcome the flow speed, where all values are upwinded. For the cases in which the HLLC state is defined as Q_{*K}, where K is either L or R, the following relations are appropriate

$$\rho_K^{HLLC} = \frac{S_K - \theta_K}{S_K - S_*}\rho \tag{2.70}$$

and is applicable to bulk density as well as species density. The pressure in this region is written as

$$p_* = p_K + \rho_K(\theta_K - S_K)(\theta_K - S_*) \tag{2.71}$$

Since a conservative formulation is utilized, the variables must be transformed to conservative variables before flux evaluation. Therefore, the remaining conservative variables are determined as

$$\rho u_K^{HLLC} = \frac{S_K - \theta_K}{S_K - S_*}(\rho_K u_K) + \frac{p_* - p_K}{S_K - S_*}\hat{n}_x \tag{2.72}$$

$$\rho v_K^{HLLC} = \frac{S_K - \theta_K}{S_K - S_*}(\rho_K v_K) + \frac{p_* - p_K}{S_K - S_*}\hat{n}_y \tag{2.73}$$

$$\rho w_K^{HLLC} = \frac{S_K - \theta_K}{S_K - S_*}(\rho_K w_K) + \frac{p_* - p_K}{S_K - S_*}\hat{n}_z \tag{2.74}$$

$$\rho e_t^{HLLC} = \frac{S_K - \theta_K}{S_K - S_*}\rho e_{t_K} + \frac{p_* S_* - p_K \theta_K}{S_K - S_*} \tag{2.75}$$

2.6 Practical Issues With Preconditioner

Several practical issues are of concern when utilizing the above preconditioner. Namely, the issue of the choice of β as well as some modifications to the idealized eigensystem to ensure stability are examined.

2.6.1 Choice of β

The choice of β is of particular interest for preconditioning schemes. Sreenivas [26] showed that the choice of a global β parameter equal to Mach^2 was sufficient to provide satisfactory convergence and accuracy results for a single species variable Mach formulation. Gupta [18] utilized a local β parameter and noted that for stagnation regions β must be limited to avoid numerical instability with an optimal value of Mach number squared also used in well-behaved regions. Unrau [27] also used a local β and proposed the preconditioning parameter should consist of a free parameter multiplied by the local Mach number squared, i.e. ϕMach^2 where $\phi < 3$. Instabilities associated with the acoustic modes of the flow field approaching the local diffusion velocity were observed in [27].

Additionally, the specification of a local von Neumann number, $\mu\Delta t/\rho\Delta x^2$ must be considered. Unrau [27] noted that in low Reynolds number regions, the time step must be set to limit the von Neumann number to unity. In the current work, this criterion was found to be far too restrictive to allow for useful simulation times. It was also found that this criterion is not necessary with the reduction of the β parameter. It is suspected that the effect noted by Unrau [27] is actually related to the increase in the mandated critical time-step as the local eigenvalues approach unity in a preconditioned flow solver.

It should be noted that in the above cited works, none considered multi-species simulations. In this work it is noted that highly converged solutions for very small values of β were not always possible using inadequately resolved meshes. However, by applying a preconditioning parameter which was not precisely equal to Mach^2 in these cases resulted in greatly enhanced convergence. It is suspected that the trace elements being modeled in many of the cases of interest result in increased "stiffness" which are beyond the performance threshold of the applied preconditioner. Unrau [27] showed, however, that despite limiting β from the ideal value, acceptable accuracy results were still obtained.

Figure 2.1 shows how the eigenvalues associated with acoustic and convective wavespeeds approach a constant value as the preconditioning parameter, β, is reduced towards Mach^2. This reduction in magnitude of the acoustic wavespeeds is the primary benefit of the preconditioning method demonstrated. By rescaling the eigenvalues, the spectral radius is reduced and convergence enhanced.

It has been reported by others [26] that replacing total pressure with gauge pressure in flux calculation aids in convergence at low Mach numbers. This modification has the effect of

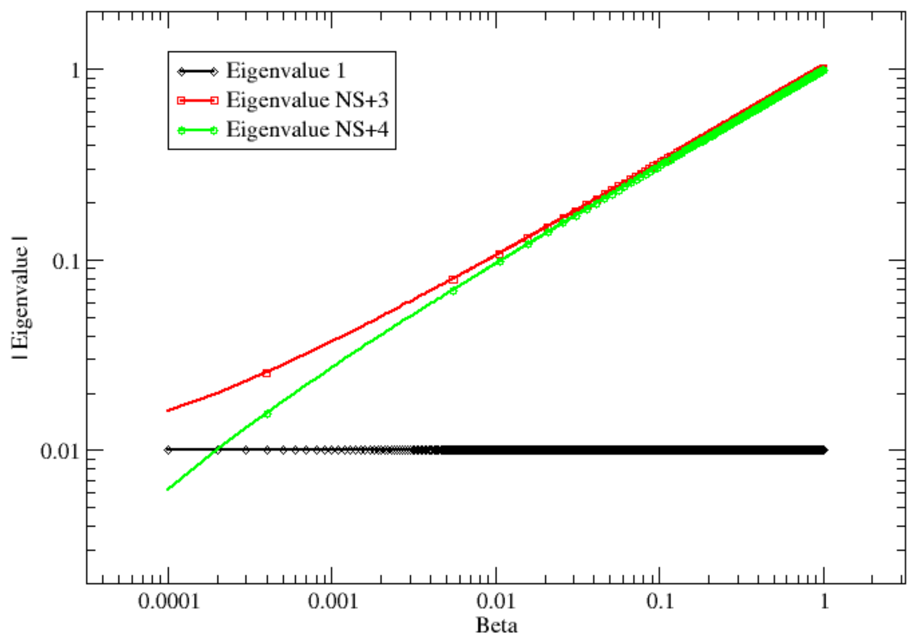


Figure 2.1 Illustration of eigenvalue scaling at Mach 0.01 as β is varied from 1 (no preconditioning) to Mach^2 (full preconditioning)

minimizing accumulation of roundoff errors associated with the surface integral of pressure. At low Mach numbers this contribution becomes poorly scaled in relation to other the flux components and utilizing gauge pressure assists in mitigating this issue.

2.6.2 Modification of Eigensystem for Stability

The replacement of the species speed of sound, c_i, with the mixture speed of sound, c, greatly enhances convergence. This is believed to be due to the decoupling in densities allowed with the use of the species speed of sound in the preconditioner. The modification made using the mixture speed of sound appears to drive the system towards convergence in a more coupled way, thus avoiding numerical stability problems.

2.7 Spatial Discretization

In the current work, spatial discretization is achieved using a node-centered (median-dual), finite-volume method on mixed-element unstructured grids. The median-dual is constructed by connecting the element centroids to edge midpoints and face centroids to form a closed cell. References available to describe this implementation in more detail may be found in [28, 29].

The discretization of spatial terms over each control volume can be expressed as

$$\mathcal{V}_i \frac{\partial Q_i}{\partial t} + \Re_i = 0 \tag{2.76}$$

where $\Re$ is the spatial residual and contains all contributions from numerical fluxes and source terms where applicable [28]. In this work $\Re$ can be represented by

$$\Re_i = \Re_{i,inviscid} + \Re_{i,viscous} - \mathcal{V}_i B_{g,i} - \mathcal{V}_i \dot{W}_i \tag{2.77}$$

This quantity is defined for every control volume i and $\frac{\partial Q_i}{\partial t}$ here represents the change of the conservative variables with respect to a particular step in time. This expression must be generalized to solve directly for the nonconservative variables via

$$\mathcal{V}_i \frac{\partial Q_i}{\partial q_i} \frac{\partial q_i}{\partial t} + \Re_i = 0 \tag{2.78}$$

where $\frac{\partial Q_i}{\partial q_i}$ can be thought of as a mapping of the conservative solution into nonconservative variable space. These terms arise on the diagonal of the linear system and are given in appendix A.

The discrete form of the residual on an unstructured mesh is expressed as a sum of the normal flux over each face enclosing the control volume

$$\Re_{i,inviscid} = \sum_{j=0}^{nfaces} F \cdot \hat{\vec{n}} \, dA \tag{2.79}$$

where dA is the subface area associated with a particular control volume face normal.

Using Equation 2.7, the conservative inviscid flux normal to these control volume faces may be written as

$$F \cdot \hat{\vec{n}} = \begin{bmatrix} \rho_1 \theta \\ \rho_2 \theta \\ \vdots \\ \rho_{NS} \theta \\ \rho u \theta + P \hat{n}_x \\ \rho v \theta + P \hat{n}_y \\ \rho w \theta + P \hat{n}_z \\ \rho h_t \theta - a_t P \end{bmatrix} \tag{2.80}$$

$$\theta = \hat{n}_x u + \hat{n}_y v + \hat{n}_z w + a_t \tag{2.81}$$

and the grid velocity term a_t is defined as

$$a_t = -[\hat{n}_x V_x + \hat{n}_y V_y + \hat{n}_z V_z] \tag{2.82}$$

where V_x, V_y, V_z are components of $\vec{V}$, the control volume face velocity vector.

The viscous fluxes are discretized by

$$\Re_{i,viscous} = \frac{1}{Re} \sum_{j=0}^{nfaces} G(\nabla Q_{ij}) \cdot \hat{\vec{n}} \, dA \tag{2.83}$$

where $G(\nabla Q_{ij}) \cdot \hat{\vec{n}}$ is the viscous flux evaluated with the solution gradient at a face between control volumes i and j.

The flux associated with the viscous stresses is discretized according to the directional derivative method. Following Hyams [28] the gradients at the control volume faces are approximated via gradient information available at control volume centroids as well as information regarding the geometric location of the control volume face. Therefore, the gradient at a face defined by the edge ij can be written as a normal and tangential contribution

$$\nabla Q_{ij} = \nabla Q_{ij,norm} + \nabla Q_{ij,tan} \tag{2.84}$$

If a directional derivative is used to approximate the normal component along the edge and the average of centroid gradients is used to approximate the tangential component,

$$(\nabla Q_{ij} \cdot \hat{s})\hat{s} \approx \frac{Q_j - Q_i}{|\Delta s|}\hat{s} \tag{2.85}$$

$$(\nabla Q_{ij} \cdot \hat{t})\hat{t} \approx \bar{\nabla Q} - (\bar{\nabla Q} \cdot \hat{s})\hat{s} \tag{2.86}$$

where $\hat{s}$ is a unit vector in the direction of the face and $\hat{t}$ is a unit vector in a direction perpendicular to the direction of the face, $\bar{\nabla Q}$ is the average of the centroid gradient values, and $\vec{\Delta s} = \vec{x_j} - \vec{x_i}$, where i is the index of the control volume of interest and j is the index of a neighbor. By combining Equation 2.85 and Equation 2.86 the following formula for a gradient at a face location is derived

$$\nabla Q_{ij} \approx \bar{\nabla Q} + [Q_j - Q_i - \bar{\nabla Q} \cdot \vec{\Delta s}]\frac{\vec{\Delta s}}{|\vec{\Delta s}|^2} \tag{2.87}$$

Thus, the only requirement for the evaluation of the viscous fluxes is the nodal gradient value and information regarding the geometry of control volumes on either side of a particular control volume face.

2.8 Higher Order Accuracy

A second order spatially accurate method is implemented by extrapolating the control volume centroid data to the control volume face location. In this work, a weighted least squares method is used to compute the solution gradients. The variables at a control volume face can be written as a truncated Taylor series expansion

$$Q_{face} = Q_i + \nabla Q_i \cdot \vec{r} \tag{2.88}$$

where $\vec{r}$ is a position vector from the control volume centroid to the control volume face. Development of the least squares gradient method used to compute the variable gradients, ∇Q_i, is found in Appendix B.

2.8.1 Limiters

Equation 2.88 represents a higher order solution reconstruction procedure. However, in the presence of shocks or other high gradient regions of the flow field, this reconstruction procedure is non-monotone. A limiting function must be introduced to reduce the extrapolation back to first order in the presence of strong flow features or the solution will become oscillatory. These oscillations, if not damped, may result in the divergence of the numerical method or non-physical solution behavior. The limited extrapolated method is

$$Q_{face} = Q_i + \phi_i \nabla Q_i \cdot \vec{r} \tag{2.89}$$

where ϕ_i is a limiting function with a value less than or equal to one. This method is higher order in smooth regions of the solution and total variation diminishing (TVD). Limiters are known to hinder convergence [30]; in this work, however, some test cases warranted their use for stability reasons.

Several limiters are implemented within the fluid solver. Two of note are those of Barth and Jespersen [31], and Venkatakrishnan [32]. The Barth limiter is implemented as follows:

1. Find largest negative difference $\delta Q_{i,min} = \min(Q - Q_i)$ and positive difference $\delta Q_{i,max} = \max(Q - Q_i)$ between the solution in the current control volume and its immediate neighbors.

2. Compute the unlimited reconstruction at each control volume face

   $Q_{face} = Q_i + \nabla Q_i \cdot \vec{r}$

3. Compute a maximum allowable value of ϕ_i for each face j

$$\phi_{ij} = \begin{cases} \min\left(1, \frac{\delta Q_{i,max}}{Q_{face} - Q_i}\right), & \text{if } (Q_{face} - Q_i) > 0 \\ \min\left(1, \frac{\delta Q_{i,min}}{Q_{face} - Q_i}\right), & \text{if } (Q_{face} - Q_i) < 0 \\ 1, & \text{if } (Q_{face} - Q_i) = 0 \end{cases} \tag{2.90}$$

4. Select $\phi_i = \min(\phi_{ij})$

The Venkatakrishnan limiter replaces the function $\min(1, y)$ with a smooth alternative

$$\psi(y) = \frac{y^2 + 2y}{y^2 + y + 2} \tag{2.91}$$

The version most often used in this work is the modification introduced by Venkatakrishnan to eliminate the limiter's effect for regions of smooth flow. This modifies the limiter in the case where $(Q_{ij} - Q_i) > 0$. For these regions

$$\phi_{ij} = \frac{1}{\Delta_-}\left[\frac{(\Delta_+^2 + \epsilon^2)\Delta_- + 2\Delta_-^2\Delta_+}{\Delta_+^2 + 2\Delta_-^2 + \Delta_-\Delta_+ + \epsilon^2}\right] \tag{2.92}$$

where $\Delta_- = Q_{ij} - Q_i$, $\Delta_+ = \delta Q_{i,max}$, and $\epsilon^2 = (K\Delta x)^3$. K is set to one in most cases and is a tunable parameter, Δx is any characteristic length scale for the control volume.

The Barth limiter is non-differentiable which has ramifications with regards to the gradient based design code as well as convergence [33]. For these reasons, all results shown make use of the modified Venkatakrishnan limiter where necessary.

2.9 Temporal Discretization

Introducing the volume averaged value of the state-vector as

$$\bar{Q} = \frac{\int_{\mathcal{V}} Q d\mathcal{V}}{\mathcal{V}} \tag{2.93}$$

The conservation equations given in Equation 2.5 become

$$\frac{\partial(\bar{Q}\mathcal{V})}{\partial t} + \int_{\partial\Omega} F \cdot \hat{n}\, dS - \frac{1}{Re}\int_{\Omega} G \cdot \hat{\vec{n}}\, dA - \int_{\Omega} \dot{W}\, d\mathcal{V} - \int_{\Omega} B_g\, d\mathcal{V} = 0 \tag{2.94}$$

and defining the spatial residual as

$$\frac{\partial(\bar{Q}\mathcal{V})}{\partial t} = -\Re \tag{2.95}$$

where

$$\Re = \int_{\partial\Omega} F \cdot \hat{n}\ dA - \frac{1}{Re}\int_{\partial\Omega} G \cdot \hat{\vec{n}}\, dA - \int_{\Omega} \dot{W}\, d\mathcal{V} - \int_{\Omega} B_g\, d\mathcal{V} \tag{2.96}$$

An implicit solution method requires the spatial residual to be evaluated at the current time level.

$$\frac{\partial\bar{Q}\mathcal{V}}{\partial t} + \Re^{n+1} = 0 \tag{2.97}$$

Using the standard backward differentiation formula (BDF2) for the temporal derivative of the equation, the above may be written as

$$\frac{1}{\Delta t}[\phi_{n+1}(\bar{Q}\mathcal{V})^{n+1} + \phi_n(\bar{Q}\mathcal{V})^n + \phi_{n-1}(\bar{Q}\mathcal{V})^{n-1}] = -\Re(Q^{n+1}) \tag{2.98}$$

or subtracting $\bar{Q}^n$ from each term, and factoring, yields

$$\begin{aligned} \tfrac{1}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1} - \bar{Q}^n)\mathcal{V}^{n+1} + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^n)\mathcal{V}^{n-1}] &\quad + \\ \tfrac{\bar{Q}^n}{\Delta t}(\mathcal{V}^{n+1}\phi_{n+1} + \mathcal{V}^n\phi_n + \mathcal{V}^{n-1}\phi_{n-1}) &\quad = \quad -\Re(Q^{n+1}) \end{aligned} \tag{2.99}$$

For BDF2 the integration coefficients are

$$\begin{aligned}\phi_{n+1} &= 3/2 \\ \phi_n &= -2 \\ \phi_{n-1} &= 1/2\end{aligned} \tag{2.100}$$

Note that the sum of these coefficients is zero. In the case of constant volume through time, the last term, the geometric conservation law (GCL) contribution, cancels which gives

$$\frac{\mathcal{V}}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1} - \bar{Q}^n) + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^n)] = -\Re(Q^{n+1}) \tag{2.101}$$

It is important to note that $\Re$ can be quite challenging to linearize exactly. This process introduces a linearization error into the solution. Also, there is error associated with the rescaling of the wave-speeds via preconditioning. Therefore, the concept of pseudo-time τ is introduced to allow these errors to be removed. This results in

$$\mathcal{V}\left(\frac{\partial Q}{\partial \tau}\right)^{n+1} + \frac{1}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1} - \bar{Q}^n) + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^n)] = -\Re^{n+1} \tag{2.102}$$

Note that this modification does not affect the temporal accuracy of the scheme as long as $\partial Q/\partial \tau$ vanishes for large values of τ. This term is now discretized using a first order backwards difference about a pseudo-time level $m+1$. Note that previous values of the solution nor the GCL depends on τ.

$$\mathcal{V}\frac{\bar{Q}^{n+1,m+1} - \bar{Q}^{n+1,m}}{\Delta \tau} + \frac{1}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1,m+1} - \bar{Q}^n) + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^n)] = -\Re^{n+1,m+1} \tag{2.103}$$

The residual at pseudo-time $m+1$ is linearized about the m^{th} time level, again using a first order approximation. This gives

$$\Re^{n+1,m+1} = \Re^{n+1,m} + \frac{\partial \Re}{\partial Q}^{n+1,m} (\bar{Q}^{n+1,m+1} - \bar{Q}^{n+1,m}) \tag{2.104}$$

Defining $\Delta Q^{n+1,m+1} = (\bar{Q}^{n+1,m+1} - \bar{Q}^{n+1,m})$ and making the appropriate substitutions yields

$$\begin{gathered} \mathcal{V}\frac{\Delta Q^{n+1,m+1}}{\Delta\tau} + \Re^{n+1,m} + \frac{\partial \Re}{\partial Q}^{n+1,m} \Delta Q^{n+1,m+1} = \\ -\frac{1}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1,m+1} - \bar{Q}^{n}) + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^{n})]] \end{gathered} \tag{2.105}$$

The right hand side still has a $\bar{Q}^{m+1,n+1}$ term present which cannot be evaluated. Subtract $\phi_{n+1}\bar{Q}^{n+1,m}\mathcal{V}/\Delta t$ from both sides and proceed formally to obtain

$$\left[\left(\frac{\mathcal{V}}{\Delta\tau} + \frac{\phi_{n+1}\mathcal{V}}{\Delta t} \right) I + \frac{\partial \Re}{\partial Q}^{n+1,m} \right] \Delta Q^{n+1,m+1} = -H \tag{2.106}$$

where H is the unsteady residual, defined by

$$H = \Re^{n+1,m} + \frac{1}{\Delta t}[\phi_{n+1}(\bar{Q}^{n+1,m} - \bar{Q}^{n}) + \phi_{n-1}(\bar{Q}^{n-1} - \bar{Q}^{n})] \tag{2.107}$$

Since the terms dependent on τ vanish as τ gets large, any procedure that will evolve the solution in pseudo-time will result in the convergence of H to the temporally correct residual. That is, τ is evolved using local time-stepping while t must remain fixed for every control volume. This greatly accelerates the solution at each unsteady time-step, t, and has no effect on the temporal accuracy provided in real-time. Using the backward differentiation formula integration scheme, the solver is formally second order accurate in time.

Note that Equation 2.106 is written in terms of conserved variables Q. The preconditioner derived in section 2.3 is applied to the time-stepping scheme from Equation 2.106 as

$$\left[\left[\frac{\partial Q}{\partial q_T}\right] P_T^{-1} \left(\frac{\mathcal{V}}{\Delta\tau} + \frac{\phi_{n+1}\mathcal{V}}{\Delta t}\right) I + \frac{\partial \Re}{\partial q_T}^{n+1,m} \right] \Delta q_T^{n+1,m+1} = -H \tag{2.108}$$

where P_T^{-1} is the inverse preconditioning matrix as applied to the $[\rho\ u\ T]^T$ variable set. The transformation to the temperature-based variable set from the pressure-+based set is straightforward and can be written as

$$P_T^{-1} = \left[\frac{\partial q_T}{\partial q_p}\right] P_p^{-1} \left[\frac{\partial q_p}{\partial q_T}\right] \tag{2.109}$$

Making this substitution results in the fully implicit, second order time-accurate method

$$\left[\left[\frac{\partial Q}{\partial q_T}\right] \left[\frac{\partial q_T}{\partial q_p}\right] P_p^{-1} \left[\frac{\partial q_p}{\partial q_T}\right] \left(\frac{\mathcal{V}}{\Delta\tau} + \frac{\phi_{n+1}\mathcal{V}}{\Delta t}\right) I + \frac{\partial \Re}{\partial q_T}^{n+1,m} \right] \Delta q_T^{n+1,m+1} = -H \tag{2.110}$$

2.10 Boundary Conditions

The Navier-Stokes equations can be classified as an initial-boundary value problem. As such, the specification of boundary conditions is necessary for solution. In this section, a set of boundary conditions applicable to the problems studied in this work are developed.

2.10.1 Isothermal No-Slip Wall

For all viscous problems studied in this work an isothermal no-slip wall boundary condition was utilized at solid surfaces. The derivation of this boundary condition follows the work by Anderson and Bonhaus [34]. These boundary conditions are applied implicitly via the modification of the block row corresponding to nodes lying on the viscous surface. The ghost nodes as well as the interior nodes which lie on the viscous surface are also set to the wall temperature and wall velocity explicitly in boundary condition enforcement. The modified diagonal block for a node lying on a viscous surface is

$$\begin{bmatrix} B_{1,1} & \cdots & B_{1,NS} & B_{1,NS+1} & B_{1,NS+2} & B_{1,NS+3} & B_{1,NS+4} \\ & & & & & & \\ B_{NS,1} & \cdots & B_{NS,NS} & B_{NS,NS+1} & B_{NS,NS+2} & B_{NS,NS+3} & B_{NS,NS+4} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta\rho_1 \\ \vdots \\ \Delta\rho_{NS} \\ \Delta u \\ \Delta v \\ \Delta w \\ \Delta T \end{bmatrix} = \begin{bmatrix} \Re_1 \\ \vdots \\ \Re_{NS} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.111}$$

In diagonal block matrix rows where the residual and the off-diagonal entries are zeroed, the off diagonal blocks must also have the appropriate entries zeroed. This ensures that during the solution of the linear system, no contributions are added to the right hand side of the equation. In short, this modification prevents updates from being non-zero in all momentum and energy equations for nodes on viscous walls.

2.10.2 Characteristic Variable Boundary Conditions

Farfield boundaries are enforced via characteristic variable boundary conditions. The governing equations can be written in 1-D for a direction normal to the boundary, η, as

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial \eta} = S \tag{2.112}$$

where S is a source term. This equation, denoting the flux Jacobian $A = \frac{\partial F}{\partial Q}$, may be expressed as

$$\frac{\partial Q}{\partial t} + A\frac{\partial Q}{\partial \eta} = S \tag{2.113}$$

A similarity transformation is used to diagonalize the Jacobian A as $A = T\Lambda T^{-1}$.

$$\frac{\partial Q}{\partial t} + T\Lambda T^{-1}\frac{\partial Q}{\partial \eta} = S \tag{2.114}$$

The matrix Λ is diagonal and contains the eigenvalues of A. T is a matrix whose columns are a the right eigenvectors of A and T^{-1} the left eigenvectors. Multiplying the above through by T^{-1} evaluated at constant conditions gives

$$\frac{\partial W_0}{\partial t} + \Lambda_0\frac{\partial W_0}{\partial \eta} = T_0^{-1} S \tag{2.115}$$

The result is a decoupled hyperbolic partial differential equation where W_0 are the characteristic variables defined as $W_0 = T^{-1}Q$. The slope of characteristics in η space is given by the eigenvalues. Therefore, $\frac{\partial \eta}{\partial t} = \Lambda$ and the above can be further simplified as

$$\frac{dW_0}{dt} = T_0^{-1} S \tag{2.116}$$

Discretizing at a boundary, b, of interest yields

$$W_{0,b} = W_{0,r} + T_0^{-1} S \Delta t \tag{2.117}$$

For the purposes of this work $S = 0$ which gives

$$W_{0,b} = W_{0,r} \tag{2.118}$$

where $W_{0,b}$ are the characteristic variables on the boundary and $W_{0,r}$ are the characteristic variables evaluated at a reference condition specified by the sign of the eigenvalue. That is, $W_{0,r}$ is evaluated from internal information for positive eigenvalues and evaluated from the outside of the domain for negative eigenvalues.

Recall, $W_{0,b} = T_0^{-1} Q_b$ and $W_{0,r} = T_0^{-1} Q_r$, and making this substitution results in

$$T_0^{-1} Q_b = T_0^{-1} Q_r \tag{2.119}$$

Multiplying through by the right eigenvalues gives

$$Q_b = T_0 T_0^{-1} Q_r \tag{2.120}$$

For example, if the first $NS+3$ eigenvalues are positive (i.e. $\theta+c$) and the last eigenvalue if negative (i.e. $\theta-c$), which is the case for subsonic outflow ($\theta < c$), the system takes the following form

$$Q_b = [T_0] \begin{bmatrix} T_{0,1}^{-1} & q_{in} \\ \vdots & \\ T_{0,NS}^{-1} & q_{in} \\ T_{0,NS+1}^{-1} & q_{in} \\ T_{0,NS+2}^{-1} & q_{in} \\ T_{0,NS+3}^{-1} & q_{in} \\ T_{0,NS+4}^{-1} & q_{ext} \end{bmatrix} \tag{2.121}$$

where $T_{0,i}^{-1}$ is the ith row of T_0^{-1} or the ith left eigenvector, q_{in} is the solution vector from the interior of the domain, and q_{ext} is the solution vector from an externally specified state. Due to the fact that the eigensystem is developed in the pressure based nonconservative variable set, the resulting output and input, q, is in terms of the pressure based nonconservative variable vector. The application of an equation of state allows this pressure to be converted readily to temperature.

2.10.3 Characteristic Impermeable Wall

The characteristic impermeable wall boundary conditions are imposed at inviscid surfaces in a similar manner to the characteristic farfield boundary conditions. Since there is no information available except from the internal portion of the flowfield, all characteristics at the ghost node are applied from internally

$$RHS_{CVBC} = \begin{bmatrix} T^{-1}_{0,1} & q_{in} \\ \vdots & \\ T^{-1}_{0,NS} & q_{in} \\ T^{-1}_{0,NS+1} & q_{in} \\ T^{-1}_{0,NS+2} & q_{in} \\ T^{-1}_{0,NS+3} & q_{in} \\ & a_t \end{bmatrix} \tag{2.122}$$

The following linear system is then solved with the last row in T^{-1} replaced to enforce a $\theta = 0$ condition. This modification can be written as

$$T^{-1}_{CVBC}\, Q_b = RHS_{CVBC} \tag{2.123}$$

where T^{-1}_{CVBC} is T^{-1} above with the last equation replaced with the following

$$\begin{bmatrix} 0 & \dots & 0 & \hat{n}_x & \hat{n}_y & \hat{n}_z & 0 \end{bmatrix} [P_b] = [a_t] \tag{2.124}$$

This has the effect of allowing the pressure at the wall to float as appropriate while strictly enforcing the no mass flow constraint at the surface.

2.10.4 Reflective Impermeable Wall

The reflective impermeable wall boundary condition is implemented as a simple reflection of the nodal velocities such that the effect is zero mass flow through the wall. This reflected velocity is

$$U_{ghost} = U_{in} - 2.0(U_{in} \cdot \vec{\hat{n}}) \tag{2.125}$$

Density and temperature at the ghost node are directly imposed from the internal node values.

2.10.5 Symmetry Plane

The symmetry plane boundary condition is implemented as being identical to the particular impermeable wall boundary condition which is active plus a gradient correction. The gradient correction is simply the removal of any normal gradient component which exist on symmetry enforced walls. This eliminates any small errors which may be present in the gradient calculation and has been shown to greatly increase the numerical stability of this boundary condition. The correction is

$$\nabla Q = \nabla Q - \nabla Q \cdot \vec{\hat{n}}_{wall} \tag{2.126}$$

where $\vec{\hat{n}}_{wall}$ is the normal face vector at the symmetry wall surface.

2.11 Jacobian Calculation

When implementing an implicit solution algorithm, the Jacobian or linearization of the residual must be available. The full Jacobian is represented by a sparse block matrix. A diagonal block of this matrix for control volume i is represented as

$$\frac{\partial R_i}{\partial Q_i} = \begin{bmatrix} \frac{\partial R_1}{\partial Q_1} & \cdots & \frac{\partial R_1}{\partial Q_{NS}} & \frac{\partial R_1}{\partial Q_{NS+1}} & \frac{\partial R_1}{\partial Q_{NS+2}} & \frac{\partial R_1}{\partial Q_{NS+3}} & \frac{\partial R_1}{\partial Q_{NS+4}} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial R_{NS}}{\partial Q_1} & \cdots & \frac{\partial R_{NS}}{\partial Q_{NS}} & \frac{\partial R_{NS}}{\partial Q_{NS+1}} & \frac{\partial R_{NS}}{\partial Q_{NS+2}} & \frac{\partial R_{NS}}{\partial Q_{NS+3}} & \frac{\partial R_{NS}}{\partial Q_{NS+4}} \\ \frac{\partial R_{NS+1}}{\partial Q_1} & \cdots & \frac{\partial R_{NS+1}}{\partial Q_{NS}} & \frac{\partial R_{NS+1}}{\partial Q_{NS+1}} & \frac{\partial R_{NS+1}}{\partial Q_{NS+2}} & \frac{\partial R_{NS+1}}{\partial Q_{NS+3}} & \frac{\partial R_{NS+1}}{\partial Q_{NS+4}} \\ \frac{\partial R_{NS+2}}{\partial Q_1} & \cdots & \frac{\partial R_{NS+2}}{\partial Q_{NS}} & \frac{\partial R_{NS+2}}{\partial Q_{NS+1}} & \frac{\partial R_{NS+2}}{\partial Q_{NS+2}} & \frac{\partial R_{NS+2}}{\partial Q_{NS+3}} & \frac{\partial R_{NS+2}}{\partial Q_{NS+4}} \\ \frac{\partial R_{NS+3}}{\partial Q_1} & \cdots & \frac{\partial R_{NS+3}}{\partial Q_{NS}} & \frac{\partial R_{NS+3}}{\partial Q_{NS+1}} & \frac{\partial R_{NS+3}}{\partial Q_{NS+2}} & \frac{\partial R_{NS+3}}{\partial Q_{NS+3}} & \frac{\partial R_{NS+3}}{\partial Q_{NS+4}} \\ \frac{\partial R_{NS+4}}{\partial Q_1} & \cdots & \frac{\partial R_{NS+4}}{\partial Q_{NS}} & \frac{\partial R_{NS+4}}{\partial Q_{NS+1}} & \frac{\partial R_{NS+4}}{\partial Q_{NS+2}} & \frac{\partial R_{NS+4}}{\partial Q_{NS+3}} & \frac{\partial R_{NS+4}}{\partial Q_{NS+4}} \end{bmatrix} \tag{2.127}$$

There are also entries in each matrix row, i, for each control volume, j, that i is connected to, $\partial R_i/\partial Q_j$. In this way the matrix is symmetric in structure but not in value. Also, for the full second order accurate or viscous linearization, there is also an entry for each neighbor of j in row i. This is referred to as the second order stencil and the matrix itself becomes considerably less diagonally dominant when these entries are included. The matrix also becomes considerably less sparse and therefore, more expensive to store. These entries need not be included for the computation of a flow field and are only necessary when considering gradient computation for design or sensitivity analysis.

These Jacobians can be computed in one of several different ways. The first method is by analytic hand differentiation. The analytic method is the least computationally expensive in most cases as a closed form derivative evaluation is performed. However, due to the complexity of the residual routine, approximations must often be made in order to differentiate analytically. In the case of the viscous contributions to the residual, only nearest neighbor contributions are made to the Jacobian. All viscous contributions are of the analytic type and are approximate. The following approximation is made for taking the derivative of any gradient at a face location

$$\begin{aligned} \frac{\partial}{\partial Q}(\nabla Q_{ij}) &\approx \frac{\partial}{\partial Q}\left([Q_j - Q_i]\frac{\vec{\Delta s}}{|\vec{\Delta s}|^2}\right) && (2.128) \\ &\approx \frac{\vec{\Delta s}}{|\vec{\Delta s}|^2} && (2.129) \end{aligned}$$

That is, only the directional derivative contribution is applied. It should be noted that this introduces a considerable amount of inaccuracy into the Jacobian related to viscous terms. However, it has been demonstrated that this approximation greatly enhances the convergence of viscous problems while maintaining sparsity in the full Jacobian matrix. This approximation is not appropriate for computation of exact linearizations which are necessary for gradient based design methods.

In most cases, numerical Jacobians are preferable for several reasons. The foremost reason is that no additional development is required when the residual routine changes or additional source terms are added. This is beneficial as it greatly increases the modularity and permits plug and play type functionality. Additionally, numerical Jacobians require no approximations be made to allow for compact analytic expressions. In this way, numerical Jacobians can be more accurate than approximate analytic expressions. Forward finite difference, central finite difference, and complex Taylor series expansion (CTSE) numerical Jacobians are available in the flow solver. Computation is very straightforward and simply involves a loop where each value of Q is perturbed, the flux computed, and the resulting derivative taken. The result is placed in the appropriate column of the Jacobian. Forward finite differences involve a single perturbation and are first order accurate, central finite differences require two perturbations and are second order accurate. Both of these are subject to subtractive cancellation errors. The CTSE method involves a single perturbation in the complex plane and exhibits true second order accuracy while not being subject to subtractive cancellation errors [35]. The only caveat is computing the diagonal contribution from boundary conditions. The boundary condition must be updated, and converged to steady state if exact contributions are required, then the flux and derivative computed. The procedure is as follows:

1. Perturb Q_j at node i as required for derivative method
2. Update boundary conditions
3. Compute boundary flux at node i
4. Take derivative (forward finite difference or complex only)

Central differences are incongruous with this method and are not used for boundary node Jacobian contributions. These numerical methods applied to constructing gradient information are discussed in more detail in Section 3.1.

2.12 Time Step Calculation

The solver can be run in several temporal modes. The first is time accurate. In this mode the time-step is specified by the user to capture relevant flow features and is applied as constant across all cells. The second mode is steady or pseudo-time-stepping. In both the steady and pseudo-time-stepping modes the parameter $\Delta\tau$ is set locally for every control volume based on a Courant-Friedrichs-Lewy (CFL) number. This is termed local-time-stepping. In the pseudo-time-stepping mode Δt is set as constant just as in the time accurate mode. This mode is also time accurate. See Section 2.9 for a deeper discussion of this scheme.

The $\Delta\tau$ value in each control volume is computed based on the eigenvalues local to that cell. The time step for a control volume is computed by

$$\Delta\tau = \text{CFL}\frac{\mathcal{V}}{\sum_{i=0}^{nfaces} |\lambda_i|_{max}\|n\|} \tag{2.130}$$

where λ_{max} is the maximum eigenvalue at a particular volume subface and $\|n\|$ is the associated face area. In the case of the Navier-Stokes equations with finite-rate chemistry, the maximum eigenvalue will be either λ_{NS+3} or λ_{NS+4}, given in Equation 2.52 and Equation 2.53, as these are the acoustic wave-speeds. Figure 2.2 illustrates how $\Delta\tau$ values scale while varying the preconditioning parameter β. This is important to recognize from a practical standpoint. For instance, in a typical flow solve, a CFL of unity might take a very long time to converge. However, in a preconditioned flow solve at Mach 0.01 the same CFL value represents a time-step which is over fifty times larger. Therefore, more conservative CFLs are often required to maintain stability in the non-convective terms of the flux, namely

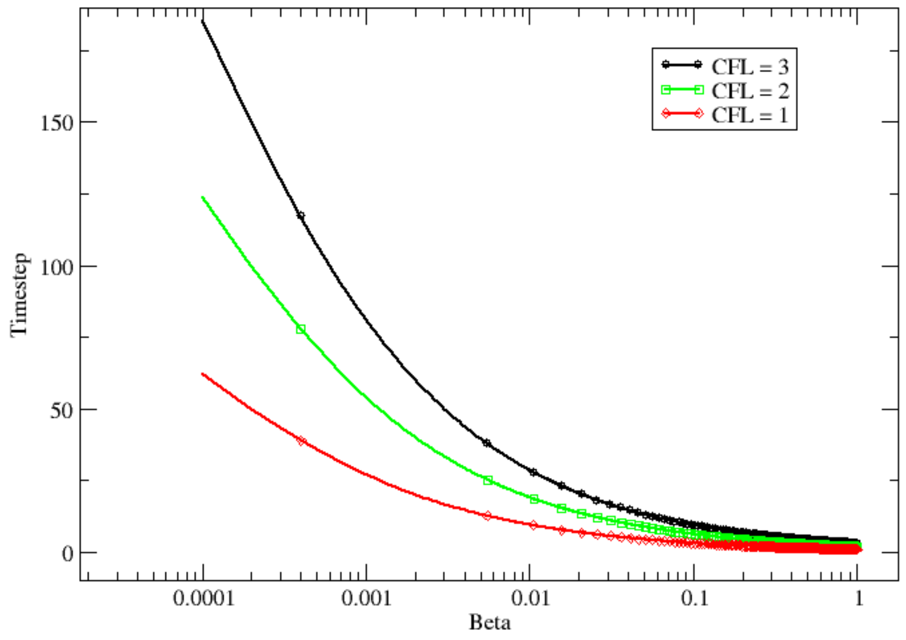


Figure 2.2 Illustration of time-step scaling at Mach 0.01 as β is varied from 1 (no preconditioning) to Mach^2 (full preconditioning) at several CFL numbers

the viscous terms. This effect is particularly pronounced due to the difficulty in properly linearizing these terms to begin with. See Section 2.11 for a discussion of the linearization used here.

2.13 Implicit Solution Algorithm

Consider the discretized form of the Navier-Stokes equations as written in Equation 2.106.

$$\left[\left(\frac{\mathcal{V}}{\Delta\tau} + \frac{\phi_{n+1}\mathcal{V}}{\Delta t}\right) I + \frac{\partial\Re}{\partial Q}^{n+1,m} \right] \Delta Q^{n+1,m+1} = -H$$

It is obvious that the above falls into the familiar form of a linear system of equations with the matrix being represented by

$$A \equiv \left[\left(\frac{\mathcal{V}}{\Delta\tau} + \frac{\phi_{n+1}\mathcal{V}}{\Delta t}\right) I + \frac{\partial\Re}{\partial Q}^{n+1,m} \right] \tag{2.131}$$

Rewriting the linear system above making this substitution gives

$$A\,\Delta Q = -H \tag{2.132}$$

This linear system must be solved in order to advance the solve in time or in pseudo-time in the case of steady calculations. For steady calculations H is simply replaced directly with the spatial residual $\Re$.

The solution to this system can be achieved by various methods. The two classes of linear system solution are direct or iterative. Direct methods are intractable due to the sparse structure of the matrix, the expected fill-in and consequently memory usage constraints are far too high for realistic scale problems. Thus, iterative methods are used. Two classes of iterative methods exist; nonstationary and stationary. Both Krylov subspace (nonstationary) and stationary iterative methods are useful but the smoothing properties of stationary methods are attractive in the study of computational fluid dynamics for many reasons. In particular, the symmetric Gauss-Seidel method is used in solving this linear system.

Given a matrix A a decomposition can be performed such that the lower triangular, diagonal, and upper triangular components can be addressed separately.

$$A = (L + D + U) \tag{2.133}$$

where L contains all lower blocks, D all diagonal blocks, and U all upper blocks. The solution to the linear system is thus iteratively obtained by

$$\Delta Q_j^{k+1} = D_j^{-1}[-H_j - (L_j + U_j)\Delta Q_j^k] \tag{2.134}$$

where k is the current iteration level in solution of the linear system and j is the current block row for which the solution is being updated.

A procedure for inverting the block D_j is still required. Here, a partial pivoted LU factorization is used. $PD_j = \hat{L}\hat{U}$ where P is a permutation matrix which permutes rows only with $\hat{L}$ and $\hat{U}$ being lower and upper triangular matrices for the block. Doolittle's method with row pivoting is used so that the computations are performed in-place for performance and memory consumption reasons. The only additional memory cost for this implementation is the storage of an integer vector which is equal in length to the number of blocks multiplied by the block rank. This decomposition is performed once per time-step. Inverting these blocks amounts to the solution of an additional block linear system for each row in the large linear system.

$$D_j \, \Delta Q_j^{k+1} = -H_j - (L_j + U_j)\Delta Q_j^k \tag{2.135}$$

This system is solved simply because of the decomposition in one forward and one backwards pass. The forward pass is

$$\hat{L}x^* = P\left[-H_j - (L_j + U_j)\Delta Q_j^k\right] \tag{2.136}$$

and the backwards pass is

$$\hat{U} \, \Delta Q_j^{k+1} = x^* \tag{2.137}$$

This procedure computes the updates ΔQ to advance the solution in time or pseudo-time. After the updates are applied to the solution vector, the residual and matrix are recomputed for the next step. This process is repeated iteratively until the residual, $\Re$, or temporal residual, H, are reduced to the required level of solution accuracy.

2.14 Chemistry

The chemistry components are largely modularized from the compressible Navier-Stokes equations solver in order to allow quick changes of chemistry models and thermodynamic databases for individual species. This section describes the methods utilized within the chemistry library for this research.

2.14.1 Thermodynamic Data

Determining thermodynamic properties of individual chemical species is not a trivial task. One method of determining these properties is through the use of NASA 7/9 coefficient polynomials. Here, the 7 coefficient variety is used exclusively, though for most species the 9 coefficient versions are also available. These 9 coefficient versions are more accurate. However, the 7 coefficient version of the polynomials typically exhibit one-tenth of one percent to one percent error at peak temperatures [36].

The thermodynamic curve fit coefficients are valid over two semi-standard ranges, 200-1000K and 1000-6000K, with some researchers [37] adding additional coefficients for ranges beyond these tables. The thermodynamic properties of a given species can be derived from these curve fits as a function of temperature. The relation used for specific heats is

$$\frac{c_{p_i}}{R_i} = a_1 + a_2 T + a_3 T^2 + a_4 T^3 + a_5 T^4 \tag{2.138}$$

which is the specific heat at constant pressure. The constant volume specific heat can be calculated as

$$c_{p_i} - c_{v_i} = T \left(\frac{\partial p}{\partial T} \right) \bigg|_v \left(\frac{\partial v}{\partial T} \right) \bigg|_p \tag{2.139}$$

An example of this calculation for the ideal gas equation of state is shown in Section 2.14.4. Specific enthalpy is calculated via

$$\frac{h_i}{R_i T} = a_1 + \frac{a_2 T}{2} + \frac{a_3 T^2}{3} + \frac{a_4 T^3}{4} + \frac{a_5 T^4}{5} + \frac{a_6}{T} \tag{2.140}$$

Specific entropy can be calculated with

$$\frac{s_i}{R_i} = a_1 \ln T + a_2 T + \frac{a_3 T^2}{2} + \frac{a_4 T^3}{3} + \frac{a_5 T^4}{4} + a_7 \tag{2.141}$$

Gibbs free energy, which can be used as a fundamental indication of an equilibrium state by minimizing the total free energy in a mixture, can be found by

$$\frac{g_i}{R_i T} = \frac{h_i}{R_i T} - \frac{s_i}{R_i} = a_1(1 - \ln T) - \frac{a_2}{2}T - \frac{a_3}{6}T^2 - \frac{a_4}{12}T^3 - \frac{a_5}{20}T^4 + \frac{a_6}{T} - a_7 \tag{2.142}$$

All of the above properties are calculated on a per species basis and are not very useful with regards to a CFD solver. What is desired are the bulk fluid properties in a particular control volume. The species mass fraction is given as

$$Y_i = \frac{\rho_i}{\rho} \tag{2.143}$$

Using the following relations, several useful properties can be computed for a localized flow field

$$c_p = \sum_{i=1}^{NS} Y_i c_{p_i} \tag{2.144}$$

$$\gamma = \frac{c_p}{c_v} \tag{2.145}$$

where c_v is defined in Equation 2.139 and is dependent on the equation of state selected. The species specific gas constant is

$$R_i = \frac{R_{univ}}{M_i} \tag{2.146}$$

and the mixture specific gas constant is defined by

$$R = \sum_{i=1}^{NS} Y_i R_i \tag{2.147}$$

Specific total enthalpy for the mixture can be calculated as

$$h_t = \sum_{i=1}^{NS} Y_i h_i + \frac{1}{2}\|U\|^2 \tag{2.148}$$

and likewise, specific total energy can be calculated via

$$e_t = \sum_{i=1}^{NS} Y_i e_i + \frac{1}{2}\|U\|^2 \tag{2.149}$$

More useful, however, is the computation of total energy which is represented by

$$E_t = h_t \sum_{i=1}^{NS} \rho_i - P \tag{2.150}$$

The square of the speed of sound is defined as the partial derivative of pressure with respect to density and can be written as

$$c^2 = \left(\frac{\partial P}{\partial \rho}\right)_s \tag{2.151}$$

This can be evaluated analytically as was done by Cox [38]. However, the full evaluation can be quite expensive to compute. Here, the assumption of ideal gas behavior was used. This can be written as

$$c^2 = \gamma RT \tag{2.152}$$

where γ and R are both defined for the total mixture of gases [39].

2.14.2 Thermal Transport Properties

The NASA reports by McBride and Gordan represent the methodology used within this work in the determination of thermal transport properties [40, 41]. The data provided is in the form of temperature-dependent four coefficient curve-fits.

$$\left.\begin{array}{l} ln\ \mu \\ \\ ln\ \tilde{k} \end{array}\right\} = A\ ln\ T + \frac{B}{T} + \frac{C}{T^2} + D \tag{2.153}$$

where viscosity, μ, is reported in micropoise and the thermal conductivity in units of microwatts per centimeter-kelvin. The mixture thermal conductivity is computed via a Wilkes mixture rule:

$$\tilde{k} = \sum_{i=1}^{NS} \frac{x_i \tilde{k}_i}{\sum_{j=1}^{NS} x_j \Phi_{ij}} \tag{2.154}$$

with

$$\Phi_{ij} = \frac{\left[1 + \left(\frac{\mu_i}{\mu_j}\right)^{0.5} + \left(\frac{M_j}{M_i}\right)^{0.25}\right]^2}{\sqrt{8}(1 + \frac{M_i}{M_j})^{0.5}} \tag{2.155}$$

where x_i is the mole fraction of species i which can be computed by

$$x_i = \frac{\rho_i}{\rho}\frac{1}{M_i} = \frac{Y_i}{M_i} \tag{2.156}$$

Likewise, the mixture viscosity is computed via a similar mixture rule

$$\mu = \sum_{i=1}^{NS} \frac{x_i \mu_i}{\sum_{j=1}^{NS} x_j \Phi_{ij}} \tag{2.157}$$

Though many species are provided in the literature, it is exceedingly difficult to find data on every species of interest. In many cases, experimental data is not available nor are predicted values based on molecular structure. In the absence of information regarding a species of interest, the implementation described here reduces to the thermal transport properties for air. These properties are given by White in the form of a Sutherland type curve-fit [42].

$$\frac{\mu}{\mu_0} = \left(\frac{T}{T_0}\right)^{3/2} \frac{T_0 + S}{T + S} \tag{2.158}$$

$$\frac{\tilde{k}}{k_0} \approx \left(\frac{T}{T_0}\right)^{3/2} \frac{T_0 + S}{T + S} \tag{2.159}$$

The appropriate mixture rules still apply to compute bulk transport properties in these cases.

2.14.3 Reaction Rates

The reaction rates for a particular reaction can be computed via several methods. In this research, the Arrhenius and modified Arrhenius forms are used. All reactions in this work are assumed to be laminar. That is, turbulent chemistry interactions are not resolved via probability density functions [43] or other modeling techniques. The Arrhenius equation is

$$K = Ae^{-E_a/R_{univ}T} \tag{2.160}$$

where A is a prefactor and E_a is the activation energy of the reaction in question. The modified Arrhenius form is

$$K = A\left(\frac{T}{T_0}\right)^n e^{-E_a/R_{univ}T} \tag{2.161}$$

where T_0 is some reference temperature and n is a unitless power. Other forms do exist but are not used here.

Reaction rates are typically given in terms of forward reaction rate, $K_{f,r}$, only. It is then left to the researcher to derive the backward reaction rate, $K_{b,r}$. This can be done with

$$K_{b,r} = \frac{K_{f,r}}{K_{c,r}} \tag{2.162}$$

where the equilibrium coefficient $K_{c,r}$ must be determined first. It can be computed from thermodynamic properties alone [13].

$$K_{p,r} = e^{\left(\frac{\Delta s^o_{r,r}}{R_{univ}} - \frac{\Delta h^o_{r,r}}{R_{univ}T}\right)} \tag{2.163}$$

where

$$\Delta s^o_{r,r} = \sum_{i=1}^{NS}\left(\nu''_{i,r} - \nu'_{i,r}\right) s^o_i \tag{2.164}$$

and

$$\Delta h^o_{r,r} = \sum_{i=1}^{NS}\left(\nu''_{i,r} - \nu'_{i,r}\right) h^o_i \tag{2.165}$$

in which s_i^o and h_i^o are the standard state entropy and enthalpy, respectively. From this the reaction rate based on concentration can be derived, which is

$$K_{c,r} = K_{p,r} \left(\frac{p_{std}}{R_{univ} T} \right)^{\sum_{i=1}^{NS} \left(\nu''_{i,r} - \nu'_{i,r} \right)} \tag{2.166}$$

where P_{std} is 1 atmosphere.

2.14.4 Equation of State

Throughout this work, the equation of state which is selected is the ideal gas law. This is given by

$$p\mathcal{V} = nR_{univ}T \tag{2.167}$$

or rearranged to use the specific gas constant, R,

$$p = \rho RT \tag{2.168}$$

However, any equation of state may be used which has several functions defined. Since most chemical databases give specific heat in terms of constant pressure, the equation of state model must be able to convert this to specific heat at constant volume.

The equation of state must also be able to return a pressure value given density, mixture properties, and temperature. In this work, due to the implementation of the preconditioners in the $[\rho\ u\ p]^T$ variable set, the equation of state should be able to return temperature given pressure, density, and mixture properties instead. The constant volume specific heat give in

Equation 2.139 can be calculated as

$$c_{p_i} - c_{v_i} = T \left(\frac{\partial p}{\partial T}\right)\bigg|_v \left(\frac{\partial v}{\partial T}\right)\bigg|_p \tag{2.169}$$

The partial derivatives take the following values for the ideal gas equation of state only

$$\frac{\partial p}{\partial T} = \frac{R}{\mathcal{V}} \tag{2.170}$$

$$\frac{\partial v}{\partial T} = \frac{R}{p} \tag{2.171}$$

This results in Mayer's relation for ideal gases

$$c_p - c_v = R \tag{2.172}$$

Several functions are also required for the linearization of the viscous flux. An example of these are

$$\frac{\partial T}{\partial p},\ \frac{\partial T}{\partial R},\ \frac{\partial R}{\partial \rho_i},\ \text{and}\ \frac{\partial v}{\partial T}$$

Finally, the transformation from conservative to nonconservative variables in the mapping Jacobian requires

$$\frac{\partial \rho e_t}{\partial p},\ \frac{\partial \rho e_t}{\partial \rho_i},\ \text{and}\ \frac{\partial p}{\partial T}$$

These requirements are a product of assumptions made about fluid behavior during the implementation of the solver. However, by placing all equation of state calculations within a replaceable module, tremendous flexibility is possible when fluid behavior deviates from the ideal gas law.

2.15 Spalart-Allmaras Turbulence Model

All turbulent flows computed in this work make use of the Spalart-Allmaras turbulence model [44]. The trip term f_{t1}, which is noted in the first reference, is not used here and instead the freestream boundary conditions are modified according to [45]. The turbulence model is enforced in a loosely-coupled fashion after each pseudo-time-step of the core flow solver. The non-dimensionalization used when implemented should be consistent with values of velocity, distance, and time used in the core flow solver. The dimensional form of the Spalart-Allmaras turbulence model may be expressed as

$$\begin{aligned}\frac{\partial \tilde{\nu}}{\partial t} + u_j \frac{\partial \tilde{\nu}}{x_j} = C_{b1}[1 - f_{t2}]\tilde{S}\tilde{\nu} + \frac{1}{\sigma}\left[\nabla \cdot [(\nu + \tilde{\nu})\nabla\tilde{\nu}] + C_{b2}|\nabla\tilde{\nu}|^2\right] - \\ \left[C_{w1} f_w - \frac{C_{b1}}{\kappa^2} f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2\end{aligned} \tag{2.173}$$

where the following quantities are defined

$$\tilde{S} = \Omega + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2};\ \Omega = \sqrt{2W_{ij}W_{ij}} \tag{2.174}$$

$$\tag{2.175}$$

with d defined as the distance from the nearest wall.

$$f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3};\ f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{2.176}$$

$$\tag{2.177}$$

$$W_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}\right);\ f_{t2} = C_{t3} exp(-C_{t4}\chi^2) \tag{2.178}$$

$$\tag{2.179}$$

$$f_w = g\left[\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right]^{1/6};\ g = r + C_{w2}(r^6 - r) \tag{2.180}$$

$$\tag{2.181}$$

$$r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2} \tag{2.182}$$

The turbulent eddy viscosity is computed from the following expression

$$\mu_t = \rho\tilde{\nu} f_{v1};\ \nu = \frac{\mu}{\rho} \tag{2.183}$$

$$\tag{2.184}$$

The boundary conditions are

$$\tilde{\nu}_{wall} = 0;\ \tilde{\nu}_{farfield} = 3\nu_\infty \text{ to } 5\nu_\infty \tag{2.185}$$

The constants are

$$C_{b1} = 0.1355;\ \ \sigma = 0.3;\ \ C_{b2} = 0.622;\ \ \kappa = 0.41$$

$$C_{w2} = 0.3;\ \ C_{w3} = 2;\ \ C_{v1} = 7.1$$

$$C_{t3} = 1.2;\ \ C_{t4} = 0.5;\ \ C_{w1} = \frac{C_{b1}}{\kappa^2} + \frac{1 + C_{b2}}{\sigma}$$

The value of $\tilde{S}$ is limited to be no smaller that 0.3Ω.

2.16 Sensor Sampling Technique

Sampling interpolated point data is not a trivial task to perform efficiently in parallel. A common approach is to use volume coordinate searches; however, experience has shown that choice of a starting element and round off errors tend to make these methods troublesome even in serial computations. There is no guaranteed algorithmic complexity and worse, the method is not guaranteed to walk in an optimum direction. The addition of walking across parallel mesh boundaries adds additional complexity and the possibility of serialization bottlenecks. In this work, the sample is located via an octree based method. Though the octree requires tree traversal, in practice the tree depth is small and bounded. Furthermore, the remaining number of nodes to be searched in each leaf for closeness is fully parallelizable.

The implemented sample operation is performed in two stages. The first stage can be considered preprocessing and consists of finding the owning process and the nearest node to the sample location. This operation builds a coarse octree surrounding all nodes of a domain and searches within the octree for a nearest node to each of the sensor locations indicated in the configuration file. Every process then owns a list of nearest nodes and distances. This list is reduced in parallel to find the global nearest node to each of the requested sample locations.

After each process contains the mesh node nearest to each sample location, the local stencil of the nearest node, i.e. all the nodes connected to it in the mesh, is extracted. The extracted stencil amounts to a local neighborhood of known quantities around an unknown, the sample value. The local stencil solution values are used to build a inverse distance weighted interpolation which takes the form

$$Q(x) = \sum_{i=0}^{npts} \frac{w_i(x) Q_i}{\sum_{j=0}^{npts} w_j(x)} \tag{2.186}$$

where the weights are

$$w_i(x) = \frac{1}{\mathrm{dist}(x, x_i)^m} \tag{2.187}$$

and m is a tunable parameter which controls the weighting given to nodes based on proximity. Higher values weight closer values more heavily. Since, these stencils tend to be very localized, there is minimal sensitivity to this parameter. Therefore, a value of two is most often chosen.

2.17 Gaussian Plume

All plumes discussed in this work were assumed to have a source density distribution which is represented by a two-dimensional Gaussian function. This Gaussian function is always assumed to lie on a boundary. A plume may be relocated along this specified boundary by the optimization code. In the absence of the ability to locally refine the mesh, the total mass inflow has the potential to vary significantly as the plume moves and predictions are updated. Thus, the density values for a boundary node in a plume are specified via mass inflow calculated via higher order quadrature. Details of the integration procedure can be found in Appendix C. This practice insures that the total mass being contributed does not vary significantly as the plume traverses a mesh from fine to coarse regions. The Gaussian function utilized in the current work may be expressed as

$$g(x, y) = A\, exp\left(-\left(\frac{(x - x_0)^2}{2\sigma_x^2} + \frac{(y - y_0)^2}{2\sigma_y^2}\right)\right) \tag{2.188}$$

where (x_0, y_0) is the center of the plume, σ_x^2 and σ_y^2 are the spread of the plume, (x, y) is a sampled location, and A is the amplitude of the plume. The integrated values for the plume source are then mapped back to the nodes using a linear interpolation method. From this information a Dirichlet type boundary condition can be imposed in the region of influence of the plume such that the total mass flux across the surface is constant regardless of location. A normal velocity component may also be specified to allow for injection of the plume material through any low velocity boundary layers.

CHAPTER 3

COMPUTATIONAL DESIGN

In order to compute derivatives of objective functions, I, with respect to design variables, e.g. plume location, Mach number, etc., a general method of computing partial derivatives is required. Analytic differentiation of outputs with respect to inputs is possible. However, this is extremely time intensive, error prone, and adds significant size to a code base. Alternatively, numerical approaches may be adopted. However, real-valued finite differences require, at minimum, two output function evaluations, which may be quite costly. All standard numerical derivatives can also be quite sensitive to the choice of perturbation size. One solution to this problem is to use a complex Taylor series expansion (CTSE) [46, 35, 47]. Using this method, it is possible to get second order accurate derivatives with a step size near machine precision using a single complex-valued function evaluation.

3.1 Numerical Approaches to Gradient evaluation

Any smooth (differentiable) function can be expressed as a Taylor series expansion as

$$f(x+h) = f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + ... \tag{3.1}$$

It follows that $f(x)$ can be subtracted from each side and divided by the perturbation size h which yields

$$\frac{f(x+h)-f(x)}{h} = f'(x) + \frac{f''(x)}{2!}h + \frac{f'''(x)}{3!}h^2 + ... \tag{3.2}$$

Note that if this expansion is truncated after the first derivative, the forward finite difference derivative appears as

$$f'(x) = \frac{f(x+h)-f(x)}{h} + O(h) \tag{3.3}$$

That is, the largest error term is roughly the size of the perturbation h.

Instead of using a real perturbation, a perturbation may be performed in the complex plane where $i^2 = -1$. Rewriting the Taylor series expansion above using the complex perturbation $h\,i$

$$f(x+h\,i) = f(x) + f'(x)(h\,i) - \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}(h\,i)^3 + ... \tag{3.4}$$

Taking the imaginary part of the expression and dividing by the perturbation size gives an approximation to the first derivative of $f(x)$ that has a leading error term of order h^2.

$$f'(x) = \frac{\mathrm{Imag}(f(x+h\,i))}{h} + O(h^2) \tag{3.5}$$

A higher accuracy is achieved for the derivative with a single function evaluation. More remarkable is that function is never perturbed in the real plane to arrive at a derivative evaluation. Instead, it is simple enough on modern computers to perturb the complex part of a variable the derivative is required of and run the code as usual but using complex arithmetic. At the end of the computation, the complex part of the resulting function evaluation is divided by the complex perturbation itself to achieve a second order accurate first derivative. This is without the addition of any routines to compute finite differences. Essentially, the programming effort of computing derivatives is mitigated by using complex arithmetic operations.

An error is added to the real part of the function evaluation. However, this error is of order $O(h^2)$. The careful selection of h on the order of $\sqrt{\epsilon}$, where ϵ is machine zero, will result in a truncation error which is smaller than a number which can be represented by machine accuracy. In short, the additional error is unresolved on a finite precision computer.

3.2 Gradient Based Design Optimization

Consider an objective function I to minimize via the manipulation of several design variables β_j. The functional form of the objective function is

$$I = I(Q(\beta_j), X(\beta_j), \beta_j) \tag{3.6}$$

where $X(\beta_j)$ represents grid dependence on the design variables and $Q(\beta_j)$ is the solution dependence on the same.

Taking the total derivative of the objective function with respect to each of the design variables gives

$$\frac{dI}{d\beta_j} = \frac{\partial I}{\partial Q}\frac{\partial Q}{\partial \beta_j} + \frac{\partial I}{\partial X}\frac{\partial X}{\partial \beta_j} + \frac{\partial I}{\partial \beta_j} \tag{3.7}$$

The residual evaluation at steady-state may be expressed in a similar form as

$$\Re = \Re(Q(\beta_j), X(\beta_j), \beta_j) = 0 \tag{3.8}$$

Again, taking the total derivative with respect to β_j gives

$$\frac{d\Re}{d\beta_j} = \frac{\partial \Re}{\partial Q}\frac{\partial Q}{\partial \beta_j} + \frac{\partial \Re}{\partial X}\frac{\partial X}{\partial \beta_j} + \frac{\partial \Re}{\partial \beta_j} = 0 \tag{3.9}$$

Rearranging this equation results in

$$\frac{\partial Q}{\partial \beta_j} = -\left[\frac{\partial \Re}{\partial Q}\right]^{-1}\left(\frac{\partial \Re}{\partial \beta_j} + \frac{\partial \Re}{\partial X}\frac{\partial X}{\partial \beta_j}\right) \tag{3.10}$$

Given Equations 3.7 and 3.10 several methods can be developed for computing the gradients of the objective function with respect to the design variables.

3.2.1 Complex Arithmetic Mode

Due to the software design, all routines are internally templated such that each can be called in either real-valued or complex modes. Because of the nature of complex arithmetic, as discussed in Section 3.1, perturbing the complex portion of any parametric parameters (Mach number, plume location, node coordinates, etc.) and running the solver to convergence is straightforward. The derivative of the objective function with respect to the parameter of interest may be evaluated as

$$\frac{dI}{d\beta_j} = \frac{\text{Imag}[I(Q_{perturb}, X_{perturb}, \beta_{j,perturb})]}{h} \tag{3.11}$$

Though the cost is high, over two times the cost of a forward solve, the complex arithmetic mode provides a very reliable and concurrently updated benchmark for comparing the derivative accuracy of other modes. The complex arithmetic mode is also invaluable for finding coding errors in the implementation of more complicated methods.

3.2.2 Forward (Direct) Mode

Substituting Equation 3.10 into Equation 3.7 gives the following

$$\frac{dI}{d\beta_j} = -\frac{\partial I}{\partial Q}\left[\frac{\partial \Re}{\partial Q}\right]^{-1}\left(\frac{\partial \Re}{\partial \beta_j} + \frac{\partial \Re}{\partial X}\frac{\partial X}{\partial \beta_j}\right) + \frac{\partial I}{\partial X}\frac{\partial X}{\partial \beta_j} + \frac{\partial I}{\partial \beta_j} \tag{3.12}$$

This is referred to as the forward or direct mode of gradient calculation. It is important to note here that this formulation requires the solution of a complete (and potentially very costly) linear system for each β_j considered. This system is given by Equation 3.10. For a large number of design variables this can be prohibitively expensive.

3.2.3 Reverse (Adjoint) Mode

The transpose of Equation 3.12 may be utilized to rearrange the gradient of the objective as

$$\left(\frac{dI}{d\beta_j}\right)^T = -\left(\frac{\partial\Re}{\partial\beta_j} + \frac{\partial\Re}{\partial X}\frac{\partial X}{\partial\beta_j}\right)^T \left[\frac{\partial\Re}{\partial Q}\right]^{-T} \left(\frac{\partial I}{\partial Q}\right)^T + \left(\frac{\partial X}{\partial\beta_j}\right)^T \left(\frac{\partial I}{\partial X}\right)^T + \left(\frac{\partial I}{\partial\beta_j}\right)^T \tag{3.13}$$

The adjoint vector, Λ, is then defined to be

$$\Lambda = \left[\frac{\partial\Re}{\partial Q}\right]^{-T} \left(\frac{\partial I}{\partial Q}\right)^T \tag{3.14}$$

which can be arranged and determined from the following

$$\left[\frac{\partial\Re}{\partial Q}\right]^{T} \Lambda = \left(\frac{\partial I}{\partial Q}\right)^T \tag{3.15}$$

3.3 Implementation Concerns

There are several concerns of a practical nature when implementing either the direct or adjoint methods for derivative computation. Notably, as the dependence of a control volume extends past its local neighborhood, as in second order accurate or viscous computations, the exact linearization makes the resulting systems very difficult to solve . To mitigate these difficulties, the incremental iterative approach [48] is introduced for both the direct and adjoint methods.

For higher-order spatially accurate sensitivity analysis, the matrix $\left[\frac{\partial \Re}{\partial Q}\right]$ is the exact linearization of the residual vector. The flow solver tolerates approximations in the linearization because of the Newton formulation; however, no approximations are possible for the linear sensitivity equations. In the viscous or second order spatial accuracy cases, this sparse block matrix can have a very large bandwidth causing the storage cost to be high. The higher order linearization is also very stiff numerically and can be quite difficult to solve.

3.3.1 Incremental Iterative Method - Reverse (Adjoint) Mode

In the case of the discrete adjoint method, this problem can be mitigated by recasting the “stiff” linear system in incremental form [48] as

$$\left[\left(\frac{\mathcal{V}}{\Delta\tau}\right) + \frac{\partial \Re}{\partial Q}\right]^T_{approx} \Delta\Lambda = \left(\frac{\partial I}{\partial Q}\right)^T - \left[\frac{\partial \Re}{\partial Q}\right]^T_{exact} \Lambda \tag{3.16}$$

This operation still requires the computation of the full linearization of the second order residual which can be quite large. More importantly this requires the transpose of the full linearization which is quite expensive in terms of memory consumption. Due to the fact that size of the linear system in the finite-rate Navier-Stokes equations scales like $(NS+4)^2$, where NS is the number of tracked species, the adjoint method has not been extended past first order spatial accuracy in this work.

The prime benefit of utilizing the adjoint method is that for a high number of design variables the additional cost for each variable is the cost of a matrix vector product. Since the number of design variables in this study is low, the additional computational expense is not exorbitant when compared to the additional development cost of computing the full higher order linearization.

3.3.2 Incremental Iterative Method - Forward (Direct) Mode

The linear system in Equation 3.10 can also be very "stiff" numerically. The current form requires the explicit construction of the full linearization matrix which exhibits the same memory consumption issues illustrated with this matrix in the adjoint method. Instead of attempting to solve this system directly, the system is manipulated to again take the form of a Newton iterative solve [48, 47]. This form is

$$\left[\left(\frac{\mathcal{V}}{\Delta\tau}\right)+\frac{\partial\Re}{\partial Q}\right]_{approx}\Delta\left(\frac{\partial Q}{\partial\beta_j}\right)=-\left(\frac{\partial\Re}{\partial\beta_j}+\frac{\partial\Re}{\partial X}\frac{\partial X}{\partial\beta_j}\right)-\left[\frac{\partial\Re}{\partial Q}\right]_{exact}\left(\frac{\partial Q}{\partial\beta_j}\right)\tag{3.17}$$

This form shows convergence which is similar to the steady flow solution and shares the same conditioning. Also, the linear solve involving a full second order linearization of the residual now has been transformed into only a matrix vector product involving that same linearization. This provides the opportunity to utilize matrix free methods and eliminate the need to store the higher-order spatially accurate Jacobian.

3.3.3 Matrix Free Matrix-Vector Product

The Jacobian matrix-vector product in Equation 3.17 can be computed utilizing matrix free methods such as the Fréchet derivative. Alternatively, this product can be computed using complex arithmetic via

$$\left[\frac{\partial \Re}{\partial Q}\right]_{exact}\left(\frac{\partial Q}{\partial \beta_j}\right) = \frac{\text{Imag}\left(\Re\left(Q + i\,h\frac{\partial Q}{\partial \beta_j}\right)\right)}{h} \tag{3.18}$$

as demonstrated in Newman, et. al. [35] and Anderson, et. al. [46]. That is, the solution vector, Q, at steady state is perturbed in the complex plane by the product of a perturbation, h, and the vector $\partial Q/\partial \beta_j$. All dependencies on Q are updated in the flowfield and the residual is computed using complex arithmetic. The complex portion of the residual then contains all of the chain rule derivative information. Note that during this operation, the full linearization is never explicitly computed and the additional memory overhead is only the storage required for the computation of a complex arithmetic residual and sensitivity gradient.

3.4 Application of Complex Arithmetic to Chain Rule Computation

The complex Taylor series expansion is used in any instance when a chain rule evaluation must be made. For example, the computation of $(\partial \Re/\partial X)(\partial X/\partial \beta_j)$ can be performed by perturbing the complex part of X in the residual computation by the derivative $\partial X/\partial \beta_j$. The residual is then computed in complex arithmetic and the derivative is encoded in the complex part of the result by

$$\left(\frac{\partial \Re}{\partial X}\right)\left(\frac{\partial X}{\partial \beta_j}\right) = \frac{\text{Imag}\left[\Re\left(Q, X + i\,h\left(\frac{\partial X}{\partial \beta_j}\right), \beta_j\right)\right]}{h} \tag{3.19}$$

The derivative $\partial X/\partial \beta_j$ may itself have been computed using hand differentiation or another complex Taylor series expansion as appropriate. This methodology is extremely flexible and with careful implementation, very complex evaluations can be encoded in only four or five lines of C++ code.

3.5 Linear Elastic Mesh Deformation

It is assumed that the mesh movement can be represented as an isotropic material undergoing a linear deformation. In this way, the mesh acts in such a way as to minimize the crossing of gridlines following a boundary deformation.

The three-dimensional, generalized equations for linear elastic smoothing are

$$\begin{aligned}
\frac{\partial}{\partial x}\left[\alpha_{11}\frac{\partial u}{\partial x}\right] + \frac{\partial}{\partial y}\left[\alpha_{12}\frac{\partial u}{\partial y}\right] + \frac{\partial}{\partial z}\left[\alpha_{13}\frac{\partial u}{\partial z}\right] + \\
\alpha_{11}\Phi\frac{\partial u}{\partial x} + \alpha_{12}\Psi\frac{\partial u}{\partial y} + \alpha_{13}\Omega\frac{\partial u}{\partial z} + \\
2\left[\beta_1\frac{\partial^2 u}{\partial x\partial y} + \beta_2\frac{\partial^2 u}{\partial y\partial z} + \beta_3\frac{\partial^2 u}{\partial x\partial z}\right] + \\
\frac{\partial}{\partial x}\left[\theta_{11}\left(\frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\right] + \frac{\partial}{\partial y}\left[\theta_{12}\frac{\partial v}{\partial x}\right] + \frac{\partial}{\partial z}\left[\theta_{13}\frac{\partial w}{\partial x}\right] = 0
\end{aligned} \tag{3.20}$$

$$\begin{aligned}
\frac{\partial}{\partial x}\left[\alpha_{21}\frac{\partial v}{\partial x}\right] + \frac{\partial}{\partial y}\left[\alpha_{22}\frac{\partial v}{\partial y}\right] + \frac{\partial}{\partial z}\left[\alpha_{23}\frac{\partial v}{\partial z}\right] + \\
\alpha_{21}\Phi\frac{\partial v}{\partial x} + \alpha_{22}\Psi\frac{\partial v}{\partial y} + \alpha_{23}\Omega\frac{\partial v}{\partial z} + \\
2\left[\beta_1\frac{\partial^2 v}{\partial x\partial y} + \beta_2\frac{\partial^2 v}{\partial y\partial z} + \beta_3\frac{\partial^2 v}{\partial x\partial z}\right] + \\
\frac{\partial}{\partial x}\left[\theta_{21}\frac{\partial u}{\partial y}\right] + \frac{\partial}{\partial y}\left[\theta_{22}\left(\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z}\right)\right] + \frac{\partial}{\partial z}\left[\theta_{23}\frac{\partial w}{\partial y}\right] = 0
\end{aligned} \tag{3.21}$$

$$\begin{aligned}\frac{\partial}{\partial x}\left[\alpha_{31}\frac{\partial w}{\partial x}\right]+\frac{\partial}{\partial y}\left[\alpha_{32}\frac{\partial w}{\partial y}\right]+\frac{\partial}{\partial z}\left[\alpha_{33}\frac{\partial w}{\partial z}\right]+\\ \alpha_{31}\Phi\frac{\partial w}{\partial x}+\alpha_{32}\Psi\frac{\partial w}{\partial y}+\alpha_{33}\Omega\frac{\partial w}{\partial z}+\\ 2\left[\beta_1\frac{\partial^2 w}{\partial x\partial y}+\beta_2\frac{\partial^2 w}{\partial y\partial z}+\beta_3\frac{\partial^2 w}{\partial x\partial z}\right]+\\ \frac{\partial}{\partial x}\left[\theta_{31}\frac{\partial u}{\partial z}\right]+\frac{\partial}{\partial y}\left[\theta_{32}\frac{\partial v}{\partial z}\right]+\frac{\partial}{\partial z}\left[\theta_{33}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right]=0\end{aligned} \tag{3.22}$$

These equations are valid for both linear elasticity and Winslow smoothing [49]. In this study, only linear elastic grid smoothing was utilized. For linear elasticity the following definitions are made:

$$\begin{aligned}&\beta_1=\beta_2=\beta_3=0,\quad \Phi=\Psi=\Omega=0\\ &\alpha_{11}=\tfrac{E(1-\nu)}{(1+\nu)(1-2\nu)},\quad \alpha_{12}=\tfrac{E}{2(1+\nu)},\quad \alpha_{13}=\tfrac{E}{2(1+\nu)}\\ &\theta_{11}=\tfrac{E\nu}{(1+\nu)(1-2\nu)},\quad \theta_{12}=\tfrac{E}{2(1+\nu)},\quad \theta_{13}=\tfrac{E}{2(1+\nu)}\\ &\alpha_{21}=\tfrac{E}{2(1+\nu)},\quad \alpha_{22}=\tfrac{E(1-\nu)}{(1+\nu)(1-2\nu)},\quad \alpha_{23}=\tfrac{E}{2(1+\nu)}\\ &\theta_{21}=\tfrac{E}{2(1+\nu)},\quad \theta_{22}=\tfrac{E\nu}{(1+\nu)(1-2\nu)},\quad \theta_{23}=\tfrac{E}{2(1+\nu)}\\ &\alpha_{31}=\tfrac{E}{2(1+\nu)},\quad \alpha_{32}=\tfrac{E}{2(1+\nu)},\quad \alpha_{33}=\tfrac{E(1-\nu)}{(1+\nu)(1-2\nu)}\\ &\theta_{31}=\tfrac{E}{2(1+\nu)},\quad \theta_{32}=\tfrac{E}{2(1+\nu)},\quad \theta_{33}=\tfrac{E\nu}{(1+\nu)(1-2\nu)}\end{aligned} \tag{3.23}$$

where ν is Poisson's ratio. Young's modulus, E, is taken to be the inverse of the volume of the cell and $\nu = 0.2$ as was described in Karman [49]. By selecting E in this manner, cells with a small volume are given a very large stiffness when undergoing deformation. This has the effect of maintaining the aspect ratio of thin, anisotropic cell spacing near the wall while allowing the larger cells in the field to absorb most of the movement near the boundary. In practice this works quite well even for relatively large motion.

3.6 Implementation Issues

Though a method for the computation of derivatives with second order accuracy has been outlined in this chapter mathematically, a choice of platform is still required for the implementation of the above methods into a useful engineering tool. The platform chosen here is C++ due to the combination of the available template feature as well as computational performance associated with compiled, strongly-typed languages. The wide availability and continued development of computational libraries also makes C++ an attractive platform for future development.

3.6.1 C++ Templating and Operator Overloading

One of the primary expenses in computational fluid dynamics simulation is the modification and maintenance costs associated with continuous code development. The cost of development increases significantly in cases where accurate design derivatives are desired. In particular, the linearization of a code by analytic derivatives can take several years to mature into a robust tool [50]. On the other hand, by using the CTSE method and careful selection of software design principles, a code can be numerically differentiated with more reasonable effort and maintain synchronicity with the flow solver in real time as development continues [51].

In particular, the C++ concept of templating allows for the authoring of modular code which is general enough to perform the appropriate operations regardless of the variable type passed to it. Take for example the following piece of code

```
template <class Type>
Type Cubed_Function(Type x)
{
  return x*x*x;
}
```

At compile time, the C++ compiler examines each call to the above function, performs type resolution, and creates a machine code copy of the function for the appropriate types as they appear. In the case of CTSE, both a double precision and complex typed version is created in the executable. While this adds to the complexity in compilation, the effort is moved from the programmer to the compiler. On modern machines, this additional complexity is negligible in relation to the monolithic effort that would be required to maintain both a real-valued and complex-valued version of every function and class in the entire solver. As a more concrete example, the following is a class declaration from the CFD solver implemented here which contains all of the classes which are necessary to compute a field solution on a piece of discretized geometry.

```
//storage class for solution variables, eqnset type and
//mesh associated with a particular simulation region
template <class Type>
class SolutionSpace : public SolutionSpaceBase<Type>
{
public:
  template <class Type2>
  SolutionSpace(const SolutionSpace<Type2>& spaceToCopy);
  SolutionSpace(Param<Type>* param, PObj<Real>* p, std::string names,
                TemporalControl<Real>& temporalControl);
  ~SolutionSpace();

  void Init();
  void AddField(DataInfo dataInfo, Int stateType, Int varLocation);
  void AddField(std::string name);
  void RemoveField(std::string name);
  SolutionField<Type> & GetField(std::string name);
  const SolutionField<Type> & GetField(std::string name) const;
  Type* GetField(std::string name, Int state);
  Bool CheckField(std::string name) const;
  void WriteAvailableFields() const;
  void ValidateRequestedFields() const;
  void InitCRSSystem();

  void PreIterate();
  void PreTimeAdvance();
  void NewtonIterate();
  void PostTimeAdvance();

  void RefreshForParam();
  void ClearSolutionFromFile();
  void WriteSolution();
  void WriteSurfaceVariables(){};
  void WriteRestartFile();
  void ReadRestartFile();
  void OpenOutFiles();
  void CloseOutFiles();
  void PrintTimers();

  Mesh<Type>* m;
  EqnSet<Type>* eqnset;
  EqnSet<RCmplx>* ceqnset;
  Param<RCmplx>* cparam;
  BoundaryConditions<Real>* bc;
  Param<Type> * param;
  Sensors<Type>* sensors;
  TurbulenceModel<Type>* turb;
  Forces<Type>* forces;
  Limiter<Type>* limiter;
  PObj<Type>* p;
  CRS<Type>* crs;
  Gradient<Type>* grad;
  GaussianSource<Type>* gaussian;
```

```
  //pointers for convenience
  Type* q;
  Type* qold;
  Type* qoldm1;
  Type* qgrad;

  Type residual;
  Type residualnm1;
  std::ofstream timerOutFile;
  std::ofstream residOutFile;

private:
  std::vector<SolutionField<Type>*> fields;
  Bool isCopy;
  Type dtmin, residGlobal;
  Int nanflag;
};

//include implementations
#include "solutionSpace.tcc"
```

The entire CFD code receives this treatment and thus, any routine of interest can be used to compute output based on any of a wide variety of input types. In this study, the type of interest is complex. The additional complexity of this syntax is negligible compared to the obvious benefits.

3.6.2 Derivative Accuracy

Due to the functional dependence of the sensitivity equations, the output derivatives are quite sensitive to many small coding errors which are not readily apparent in a forward CFD solution.

One area of concern is the computation of the residual linearization as it relates to boundary conditions. The full linearization of the residual computation, and therefore the design derivatives themselves, is a function of both the interior and ghost nodes. As shown by Hou et. al. [52], the linearization computed must include these contributions or they must be pre-eliminated prior to the solution of either Equation 3.10 or Equation 3.15. An implicit pre-elimination procedure is used in this work by computing an additional contribution to the diagonal Jacobian blocks. This additional contribution is the numerical sensitivity of the boundary flux at a node i to a perturbation in Q at that node. The procedure is as follows:

1. Perturb Q_j at node i in the complex plane.
2. Update boundary conditions iteratively until converged.
3. Compute boundary flux at node i.
4. Take $\mathrm{Imag}[\Re(Q+ih)]/h$ as the additional contribution to the jth column in the diagonal block.

Also, the computation of any sensitivity where a control volume Q value is perturbed in the complex plane requires careful attention be paid to boundary condition updates. In particular, characteristic boundary conditions must be converged fully based on the new Q value [51]. The computation of characteristic boundary conditions requires an eigensystem be computed at some reference state. In most cases this is taken to be the average of the internal and ghost values. As the ghost values are updated, this eigensystem evaluation changes slightly. In a standard CFD solve or a full CTSE derivative computation the boundary conditions are slowly converged over many iterations as the whole solution field converges. In computation of nodal sensitivities this iteration process does not typically take place and a single sensitivity value at each boundary node is sampled instead. Therefore, at locations where nodal sensitivities are required on boundary surfaces with characteristic boundary conditions applied, the computation must be iterated to converge the boundary conditions sufficiently. It has been found that in most cases ten iterations is sufficient and the accuracy of the derivative will be improved by four to six additional significant figures.

Finally, care must be taken in implementation of these methods where Dirichlet boundary conditions are enabled. In particular, the velocity and temperature components of viscous boundary conditions are enforced via the modification of the Jacobian matrix, $\partial\Re/\partial Q$. Due to this modification, the right hand side vector of the linear system solved to compute $\partial Q/\partial\beta$ values must be set to appropriate values on rows where the left hand side has been modified. This implies that for Dirichlet boundary nodes, $\partial Q/\partial\beta$ must be computed outside of the linear system solve. Accuracy of the computed derivatives is not affected through these modifications.

CHAPTER 4

NUMERICAL VALIDATION

This chapter consists of several cases which validate both the low Mach preconditioner as well as the accuracy of design gradients using several different methods. The capabilities of the implemented solver are exercised in an isolated manner such that confidence in the combined methods is achieved.

4.1 Low Mach Preconditioner

The validity and performance of the implemented preconditioner is examined in comparison with a non-preconditioned case. The grid utilized for this study consists of a unit high channel which is 4 units long. A bump with a maximum height of 0.1 units and of unit length is placed on the floor of the channel beginning 1.5 units downstream of the inlet. The roof and floor of the channel are both set to inviscid wall boundary conditions. The inlet and outlet boundaries are set to characteristic boundary conditions. The compressible Euler equations were solved in this case with two species present; N2 is set to 79% and O2 is set to 21% to approximate standard atmosphere. There were no reactions present at the reference temperature of 300K. The inflow Mach number was set to 0.01 and both cases were run for 3000 iterations. Convergence of 4 orders of magnitude was achieved in both cases.

The velocity contours for this test case are shown in Figure 4.1 and Figure 4.2. The preconditioned case in Figure 4.1 shows a correct solution for this problem. The solution shown in Figure 4.2 illustrates a non-physical contours about the bump which is typical of this problem at low Mach number. Figure 4.3 shows convergence history for the bump in channel problem with a constant CFL and varying preconditioning parameter, β. As β approaches the ideal value of Mach^2, convergence is enhanced as expected. These results compare well to those of Gupta [18].

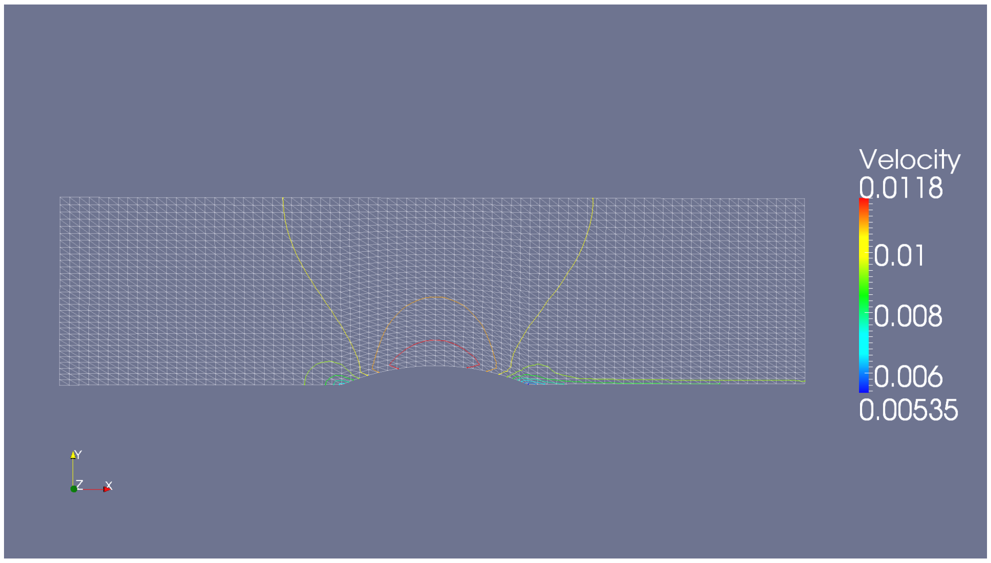


Figure 4.1 Bump in channel with $\beta = \text{Mach}^2$ (Preconditioning on)

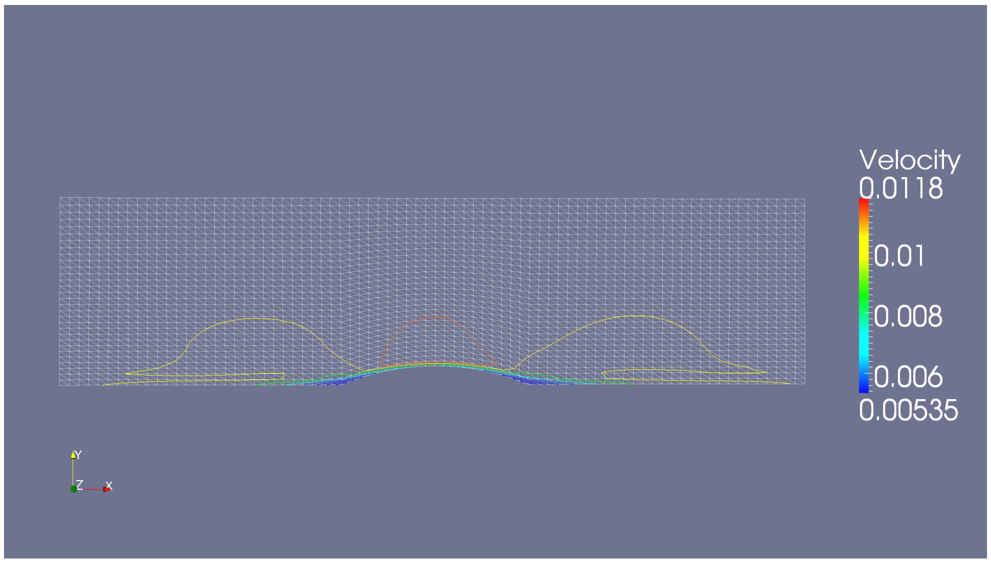


Figure 4.2 Bump in channel with $\beta = 1$ (Preconditioning off)

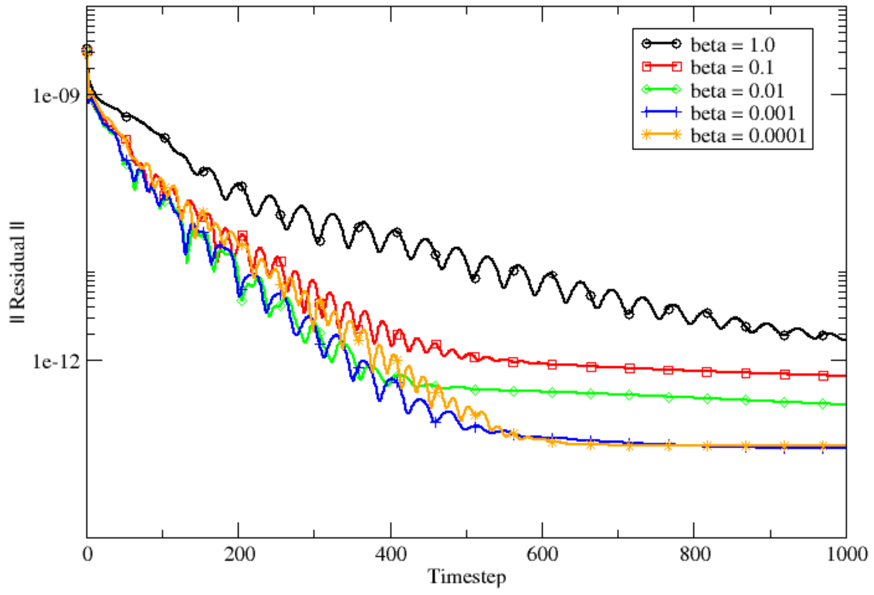
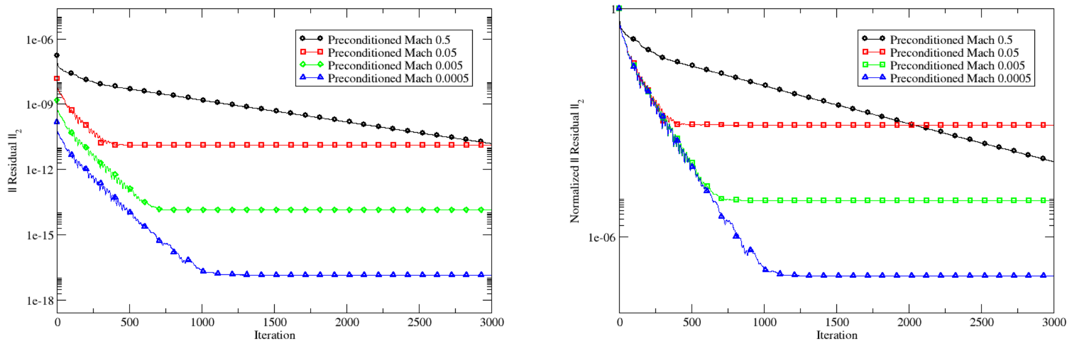


Figure 4.3 Convergence history for bump in channel with constant CFL = 3 and varying β

A behavior observed by many researchers regarding preconditioner performance is convergence which is independent of Mach number [26, 18]. Figure 4.4 shows convergence behavior for the bump in channel test case at several inflow Mach numbers. The preconditioner implemented here does not replicate Mach number independent convergence as shown by others. Convergence is slower for higher Mach numbers. It is possible that reflections are of greater magnitude in the higher Mach number cases which stall convergence. The individual components of the residual were examined and no particular equation is obviously stalling convergence performance. It is worth noting that other work in this area has made the assumption of perfect gas flows. It is conceivable that real gas effects, as well as multiple species, preclude the independent convergence behavior for some, as of yet undetermined, reason. At this point, no definitive conclusions can be made.



(a) Residual in 2 norm **(b)** Residual in normalized 2 norm

Figure 4.4 Convergence behavior for bump in a channel case at several Mach numbers and CFL = 3

Figure 4.5 shows the pressure coefficient on the lower surface of the bump in channel case for both the preconditioned and non-preconditioned solutions. Both solutions were converged three orders of magnitude in the L2-norm of the residual. A zero pressure coefficient both upstream and downstream of the bump is expected. The plot should also be symmetric about the centerline of the bump. The preconditioned solution exhibits both of these characteristics. The non-preconditioned solution exhibits neither. Clearly, the preconditioned solution demonstrates better performance in this test.

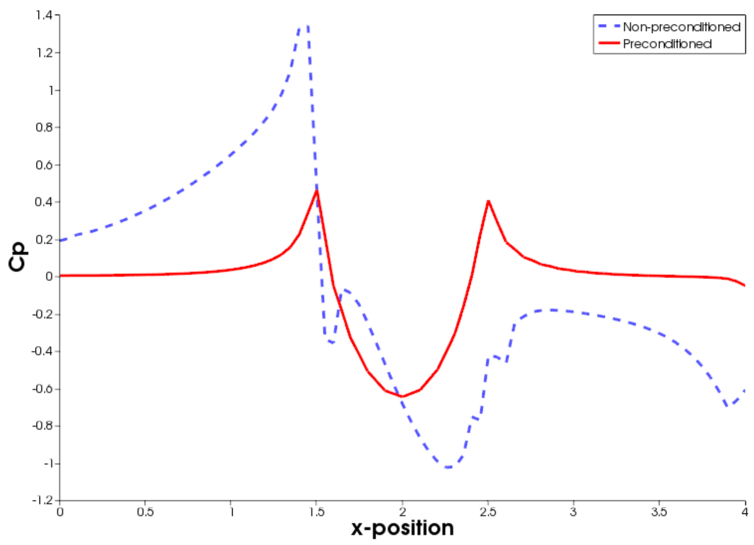


Figure 4.5 Comparison of pressure coefficient for preconditioned and non-preconditioned bump in channel

4.2 Gradient Validation - High Speed Flows

The high speed airfoil test case is designed to examine the accuracy of the computed gradients in the presence of reacting flow. This serves the purpose of validating that the chemistry modeling derivatives are computed correctly in an isolated manner.

4.2.1 Hicks-Henne Functions

The Hicks-Henne function allows for the smooth shape manipulation of a unit long bump via a four variable parameterization. This provides for a simple but relatively powerful method to test both the mesh smoothing and sensitivity derivative routines. The Hicks-Henne function is

$$b(x) = a \left[sin \left(\pi x^{\frac{log\ 5}{log\ t_1}} \right) \right]^{t_2} \quad \text{for } 0 \leq x \leq 1 \tag{4.1}$$

where $b(x)$ is the surface deformation in the normal direction, t_1 controls the location of the maximum height of the bump, t_2 controls the width of the bump in the x-direction, x is allowed to vary from zero to one along the geometry in the streamwise direction, and a is the bump amplitude. This very simple parameterization allows for tremendous flexibility in the shape of the airfoil. A combination of several Hicks-Henne functions which have been restricted to either the front or rear of the airfoil allow for even greater flexibility while reducing the total number of design variables.

4.2.2 Problem Setup

A NACA0012 airfoil shape was placed in Mach 0.8 flow and an angle of attack of 1.25 degrees. This particular simulation was run with the finite-rate Euler equations with a reduced 5-species air model [53]. The inflow conditions were set to standard air mass fractions (non-dissociated) at 2500K. This created a significant production of dissociation products as the flow moved from the inflow and ensured that mass production terms were included in the gradient calculation. The boundary conditions utilized were characteristic farfield, characteristic based impermeable wall on the airfoil, and symmetry conditions in the z-direction. The airfoil is assumed to be one meter in length. The grid consisted of approximately 5,200 nodes and 15,000 elements. At this resolution, the objective was not the accuracy of the flow solution but rather the reduced cost of solution allowing for adequate comparison of all the gradient methods. The objective function of interest was the inverse of lift coefficient for the airfoil. The starting grid is shown in Figure 4.6.

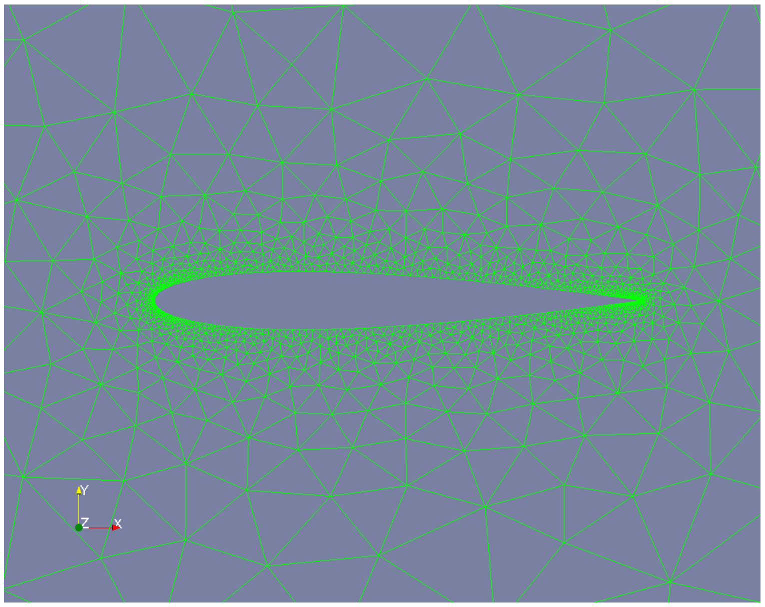


Figure 4.6 Unperturbed NACA0012 airfoil mesh

Table 4.1 shows the derivative results for the first order spatially accurate simulation. As can be seen, all the methods agree well. The design point about which the derivatives were computed was $a = 0.02$, $t_1 = 2.0$, and $t_2 = 2.0$. This ensures that all variables have active contributions. Figure 4.7 shows the grid at the sensitivity derivative evaluation point.

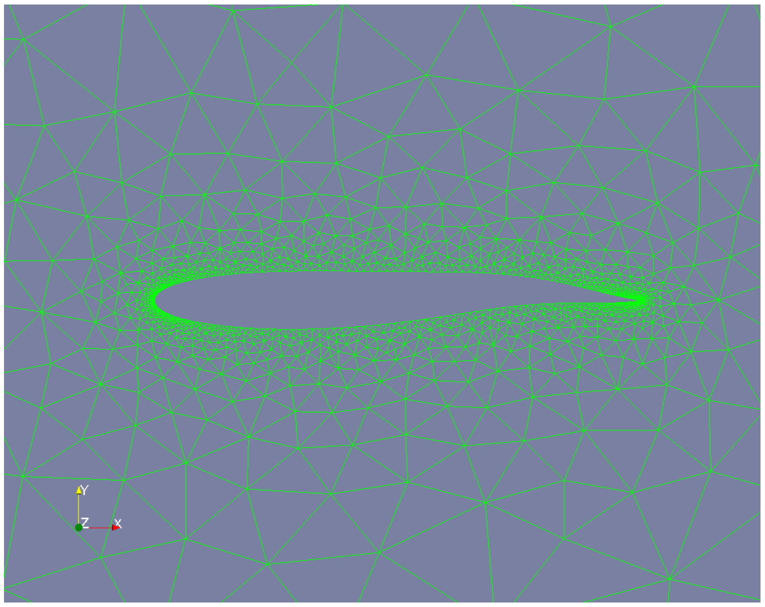


Figure 4.7 NACA0012 Airfoil mesh at gradient evaluation design point

Table 4.1 Comparison of Computed Sensitivity Derivatives - First Order

β	CTSE	Direct	Adjoint
a	-**5.734131344651**742e+01	-**5.734131344651**845e+01	-**5.734131344651**848e+01
t_1	**3.082499605763**175e-02	**3.082499605763**227e-02	**3.082499605763**405e-02
t_2	**3.696702982376**312e-01	**3.696702982376**109e-01	**3.696702982376**109e-01

Table 4.2 shows the derivative results for the second order spatially accurate simulation. Because of the the additional memory requirements and additional complexity the adjoint method was not extended to second order accuracy in this work. The derivatives computed from the CTSE and the direct method agree well. However, the same 13 digits of accuracy which were achieved in the first order analysis were not achieved in second order. The reaction front lies across a poorly resolved region in the mesh. This front is in a slightly different location in the solution which is evolved entirely in complex arithmetic (CTSE) when compared to the starting point for the direct mode derivative computation. This resulted in the objective function linearization being computed about a slightly different

solution location. The results are the slight differences observed. If the solution were computed on a higher resolution mesh or the reaction front passed through a more resolved region, it is expected that more digits of accuracy would be obtained.

Table 4.2 Comparison of Computed Sensitivity Derivatives - Second Order

β	CTSE	Direct	Adjoint
a	**-8.1043**34352528015e+01	**-8.1043**48043995304e+01	N/A
t_1	**1.59968**3455400209e-01	**1.59968**6156764921e-01	N/A
t_2	**6.2873**82995920745e-01	**6.2873**93935134606e-01	N/A

Timings shown in Table 4.3 were run on 24 cores of the SimCenter PunchCard computational cluster. This machine consists of 325 dual-processor 3.0 GHz Xeon (“Woodcrest”) dual-core servers with 1300 cores total. The cluster has a Gigabit Ethernet interconnect arranged in a flat tree topology and has a peak throughput of 15.6 teraflops. Timings are given in seconds and the final column compares the cost of each method normalized with respect to a forward flow solution. The costs are shown for the three design variables described above. The solve column shows the cost of a forward solution for the methods which require one, and the design column shows cost related to the design portion of the simulation only. For the CTSE method, a forward solution is not required as the solver is run entirely in complex mode with the output being the sensitivity derivatives desired. The cost of the direct method gradient evaluation is roughly equal to one additional solution for every design variable of interest. The cost of full complex solution gradient evaluation (CTSE) is roughly two times the cost of a standard flow solution per design variable. The adjoint method exhibits a nearly constant additional cost above the cost of a flow solution related to the cost of a parallel matrix-vector multiplication operation.

Table 4.3 Cost comparison of sensitivity derivative calculation methods

Method	Solve (s)	Design (s)	Total (s)	Normalized Cost
Adjoint	1,702	1,088	2,790	1.66
CTSE	0	13,581	13,581	8.08
Direct	1,680	5,155	6,835	4.06

4.3 Gradient Validation - Low Mach Flows

A bump in channel geometry was chosen for validation of the sensitivity derivatives in a viscous low Mach flow regime. This case also investigates the objective function and design parameters of interest for the study of flows in urban environments.

4.3.1 Problem Setup

The grid utilized for this study consists of a unit high channel which is 4 units long. A bump with a maximum height of 0.1 units and of unit length is placed on the floor of the channel beginning 1.5 units downstream of the inlet. Viscous spacing of approximately 1.0e-5 is present along the lower surface of the mesh which is consistent with $y+$ values of less than one. The inlet, outlet and top surface are set to farfield characteristic boundaries. The lower surface leading up to and away from the bump are set to symmetry planes and the bump itself is set to a constant temperature no-slip surface. A Gaussian plume source is activated on the inflow plane with a strength of 0.001 and centered at a height of 0.1 units for the gradient evaluation case. The plume was introduced through a mass source term at the inflow boundary nodes where the plume was present. The dimensional length of the bump was set to be 0.01 meters with reference viscosity and conductivity of the fluid representing that of air at 300K. The freestream velocity at the inflow was 3.82 m/s. This results in a Reynolds number of approximately 2400. The β parameter was limited to a minimum of 0.01. All cases were converged approximately three orders of magnitude. Further convergence is inhibited due to the combination of preconditioning parameter and numerical stability of the problem setup. This level of convergence was sufficient for sensor samples to stabilize in the fifth significant digit.

4.3.2 Objective Function

The objective function utilized was a root mean squared difference in the sensor values

$$I = \sqrt{(S_{1,target} - S_{1,perturbed})^2 + \cdots + (S_{N,target} - S_{N,perturbed})^2} \tag{4.2}$$

The values $S_{N,target}$ are the sampled density values for the Argon tracer gas and $S_{N,perturbed}$ are the Argon density values at sensor N for the current predicted location of the plume release. In this case 10 samples were taken from a previous run in various location both in and out of the plume. Table 4.4 gives the locations where the density of the plume was sampled.

Table 4.4 Sensor locations for the bump in channel low Mach sensitivity verification case

$S_{\#}$	x	y	z
1	-0.25	0.50	2.50
2	-0.40	0.30	0.50
3	-0.10	0.30	3.20
4	-0.25	0.30	3.60
5	-0.35	0.60	3.80
6	-0.34	0.24	3.90
7	-0.25	0.10	3.40
8	-0.25	0.80	3.00
9	-0.25	0.10	2.50
10	-0.10	0.80	0.40

4.4 Discussion

The converged solution is shown in both Figure 4.8 and Figure 4.9. The plume isosurface in Figure 4.8 shows that the plume maintains a Gaussian distribution until influenced by the region of high pressure preceeding the bump as expected.

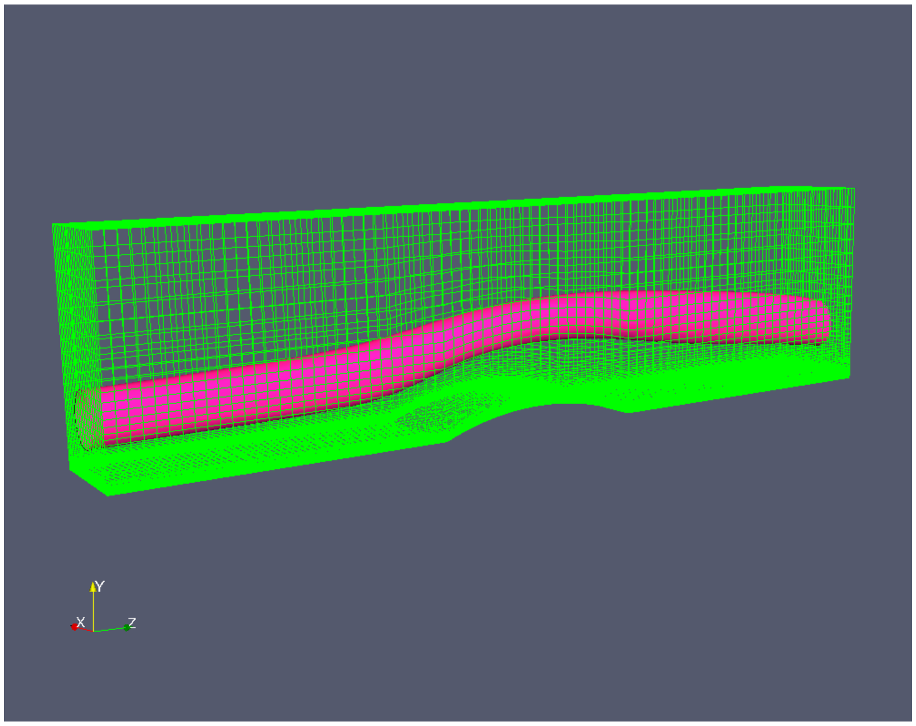


Figure 4.8 Channel bump with passive Argon plume - isosurface at Argon density of 0.1

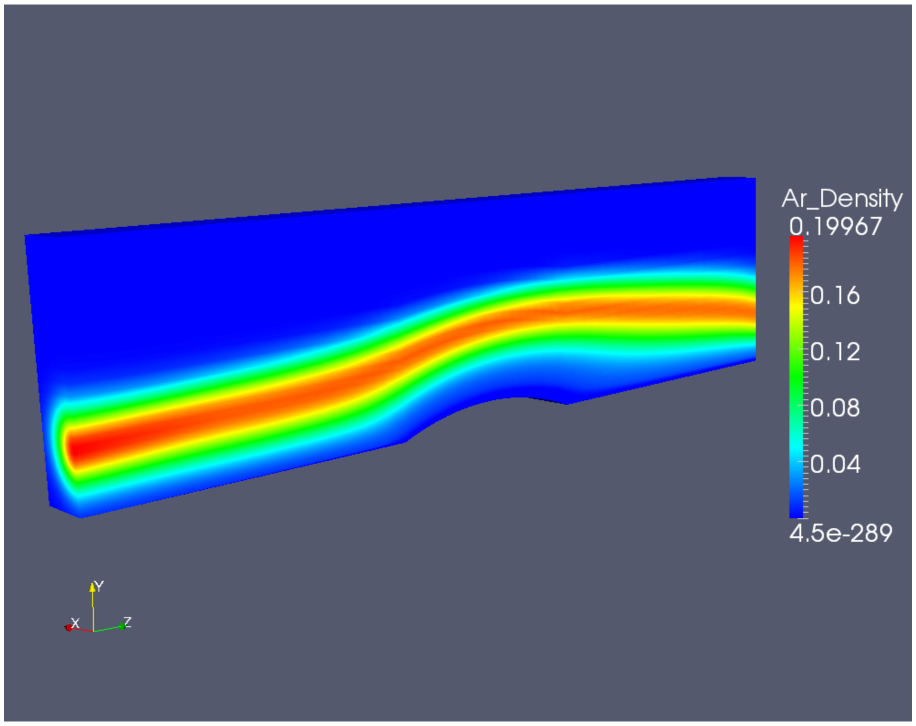


Figure 4.9 Channel bump with passive Argon plume

Table 4.5 shows the comparison of y location sensitivity derivative computed using the full complex (CTSE) approach and the direct method. The computed derivatives agree in the first two digits. This case exhibits limited accuracy due to the trace amount of Argon present in the sampling locations, difficulty in achieving stable sensor readings, and potentially due to the level of achievable convergence. A higher degree of accuracy could not be achieved given the stated problem setup. However, this result is positive as even in the case when only the sign of the derivative is computed correctly, the optimization routine will step the plume back towards the original release location. The magnitude of this derivative assists in the selection of the proper step size within the line search procedure of the optimization algorithm. However, convergence would still be expected, though perhaps not as efficiently.

Table 4.5 Comparison of Computed Sensitivity Derivatives - Second Order

β	CTSE	Direct
y	6.006751220704408e-01	6.0184976143131 26e-01

Design cycles were not attempted on this geometry. This example represents a validation case which utilizes all of the relevant modules necessary for plume release predictions on urban geometries.

CHAPTER 5

SARIN NERVE AGENT

Sarin, or GB, is an organophosphate commonly utilized as a chemical weapon. With a chemical formula of $[(CH_3)_2CHO]CH_3P(O)F$ it is chemically similar to many widely available pesticides. Sarin is both colorless and odorless making it difficult to detect in vapor phase. It is classified as a nerve agent with symptoms ranging from runny nose and disorientation at low concentrations to death at higher vapor concentrations or contact with bulk Sarin in liquid phase. Constant properties are shown in Table 5.1 [54].

Table 5.1 Liquid Sarin (GB) properties

Molecular Weight	140.1 g/mol
Liquid Density @ 25 C	1.089 g/cm^3
Vapor Density (air $\equiv$ 1)	4.86

Though the use of and production of Sarin was banned following the United Nations Chemical Weapons Convention (CWC), which became effective in 1997, it remains of interest from a security standpoint due to the relative ease of production. With relatively little sophistication, a commonly equipped laboratory is capable of producing low purity, though still deadly, Sarin agent. This is evidenced by the Tokyo subway attack where Sarin was produced by a radical religious element in a relatively low sophistication terror attack [2]. Being the most volatile of common nerve agents, the vapor is readily dispersed over wide areas. More importantly, the volatility of Sarin allows for large distances to be traversed in a downwind direction as in the Matsumoto attack by the same group [1]. Even a moderate increase in sophistication could increase the lethality substantially.

The 50 percent toxicity threshold for inhaled Sarin, LCt_{50}, is 100 $mg \cdot min/m^3$. The EPA recommended AEGL-3 (Acute exposure guideline level: life-threatening) for Sarin over 10 minutes is 0.064 parts per million (ppm) or 0.38 $\mathrm{mg/m^3}$ [54]. The AEGL-1 is only 0.0012 ppm or 0.0069 $\mathrm{mg/m^3}$. Therefore, the lethal density of Sarin nerve agent compared to the density of air at sea level is seven orders of magnitude smaller. In numeric terms, this requires that the accuracy of a flow solver should be capable of resolving density features on the order of 3.16e-7 for prediction of survivability limits on dispersed plumes. As the exposure time increases, the exposure concentration for lethality decreases substantially. For example, the AEGL-3 for Sarin exposure of 8 hours is 0.0087 ppm. For the purposes of this study, and to eliminate the need for highly converged solutions, exposure times for continuous plumes are assumed to be for low periods of time, i.e. less than 30 minutes.

Naturally, information regarding the thermochemical and physical properties of Sarin are largely absent from unclassified literature. This presents some difficulty in the analysis of a toolchain to predict dispersal locations based on real world sensor measurements. Using vaporized corn oil as a surrogate is a United States Army standard testing procedure for personal protection equipment used by all military services [55]. In this work however, the database in [36] is used for thermal properties. Surrogates are also used for decomposition reaction rates for Sarin in the presence of common urban pollutants. The report by Elliott, et. al. [56], surveys literature on the decomposition of less toxic organophosphates in urban environments. The limiting rates presented therein are used as a substitute for the Sarin rates due to the lack of available data. Mean residence time based on reaction with the OH radical for several organophosphorous compounds (pesticides) in urban environments ranges from 2.1 days to 48 minutes based on the work of Winer and Atkinson [57]. The measurements provided in [57] make use of a relative rate technique which makes the definition of temperature dependent reaction rates difficult. In this work, the assumption of constant rate was made due to the numerical difficulties associated with the fast production and destruction of the OH radical in infinitesimal quantities. However, should further improvements in the available models be made, incorporating requires little effort.

A more accurate modeling of an agent plume would also include the change in phase from an atomized spray to vapor and back again. At this point, these effects are neglected as well. In particular the vaporization of droplets has a cooling effect on the plume and thus affects its buoyancy.

The Antoine correlation for vapor pressure is derived from experiments performed at the US Army's Edgewood Chemical Biological Center [58]. Vapor pressure measurements are defined to be

$$VP_{GB} = P_{ambient} \frac{n_{GB}}{n_{GB} + n_{carrier}} \tag{5.1}$$

And the Antoine correlation for vapor pressure temperature dependence is given as

$$ln(VP) = a - \frac{b}{c + T} \tag{5.2}$$

where the coefficients for GB are

$$a = 22.720$$

$$b = 4320.8$$

$$c = -41.245$$

For the target temperature of 300K, this relation gives a saturation pressure of 412.23 Pa. Using linear interpolation, the heat of vaporization based on the same data and temperature is 48.47 kJ/mol. This implies that with a molecular weight of 140 g/mol, a 1kW heater is capable of vaporizing 2.89 g/sec of agent. Therefore a continuous incident is capable of vaporizing 10.4 kg/hr with relatively little sophistication. Explosive delivery is capable of introducing considerably more agent into an environment. Continuous releases are studied here due to the restriction of steady-state solutions to find sensitivity information. Techniques exist to perform the unsteady analysis, however, they are beyond the resources available for this work.

CHAPTER 6

TERRAIN ANALOG STUDY

Due to the computational expense of running even a single forward simulation on a real urban geometry, a test case consisting of a notional mountainous topology was constructed. The grid is 10 x 10 x 5 units in dimension with the ground plane having the larger extent. The ground plane was meshed to have a first point off the wall distance sufficient to give $y+$ values less than one. The mesh, shown in Figure 6.1, consists of approximately 4.8 million cells and 1.6 million nodes. Atmospheric boundary layers were imposed at all farfield boundaries with an inflow velocity aligned in the positive x-direction and a magnitude of approximately 3.8 m/s. The grid is nondimensionalized with a reference length of 20 meters. This gives a maximum elevation change of approximately 40 meters across the geometry. A Spalart-Allmaras turbulence model was used to compute turbulent viscosity. Molecular diffusion is not enabled due to large scale turbulence being the dominant driver of interspecies mixing.

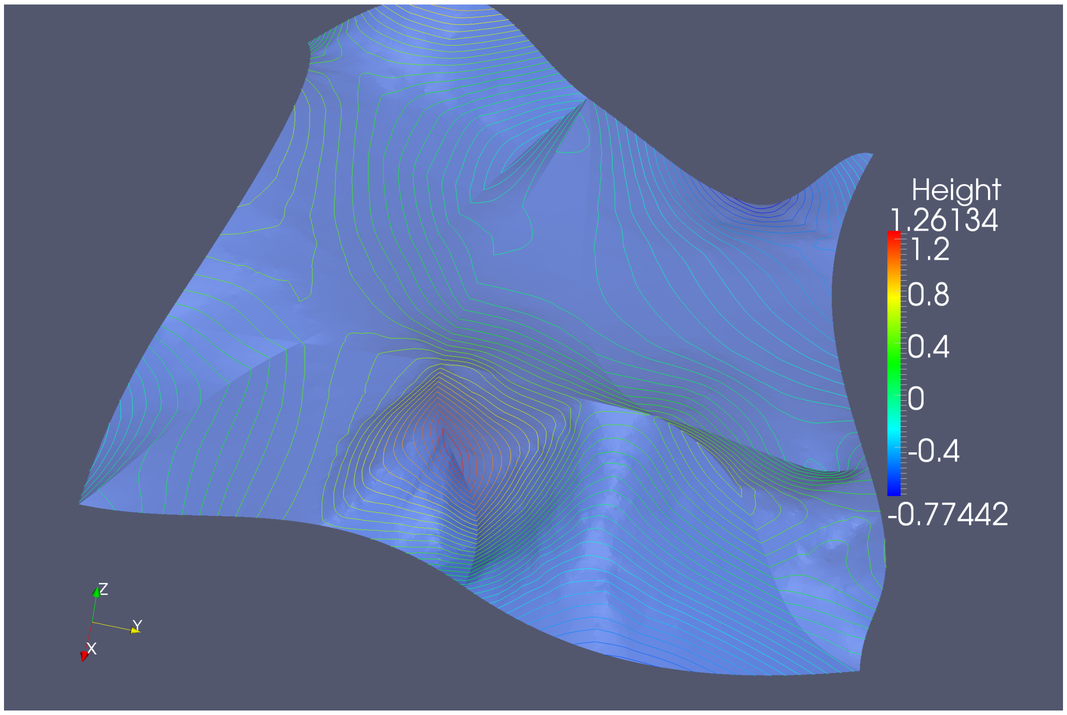


Figure 6.1 Notional mountainous terrain topology

A Sarin plume was introdcuced at the (0,0) location which is centered in the terrain. The density distribution followed a Gaussian distribution with a maximum of 0.001153 kg/m^3 at the origin of the plume. The plume was given a small upward velocity which also followed a Gaussian distribution in the plume region. The maximum upward velocity is given as approximately 1.9 m/s. This velocity is sufficient to insure the plume penetrates the boundary layer region with sufficient momentum to convect downstream in a reasonable amount of computing resources. Figure 6.2 shows the a density isosurface of the Sarin plume at 1.15e-5 kg/m^3. Streamlines are also shown colored by velocity magnitude. Solutions were computed with steady-state conditions only.

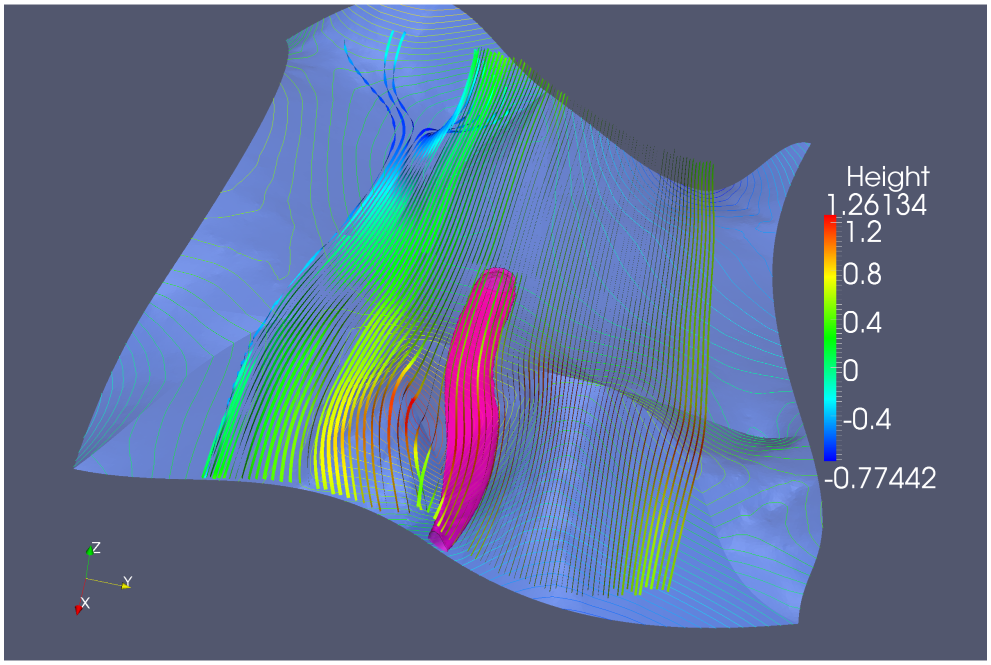


Figure 6.2 Notional mountainous terrain velocity streamlines with Sarin plume

A starting location for the design cycles was chosen arbitrarily to be (-3,-3). Several design cycles were performed and the behavior of the plume movement observed. Immediately the predicted location moved in the negative x-direction with the greatest magnitude. After only 2 design cycles the plume encountered a location which hindered further convergence towards the original plume location. Gradients computed in this demonstration case assumed a frozen turbulence model.

Gradient based design methodologies are sensitive to the phenomenon of local minima. It is readily apparent that for a general spread of randomly place sensors, gradient based methods fail on plume inversion problems due to this difficulty. Figure 6.3 shows the sensor locations relative to the starting location of the design cycles as well as relative to the target location (shown in red).

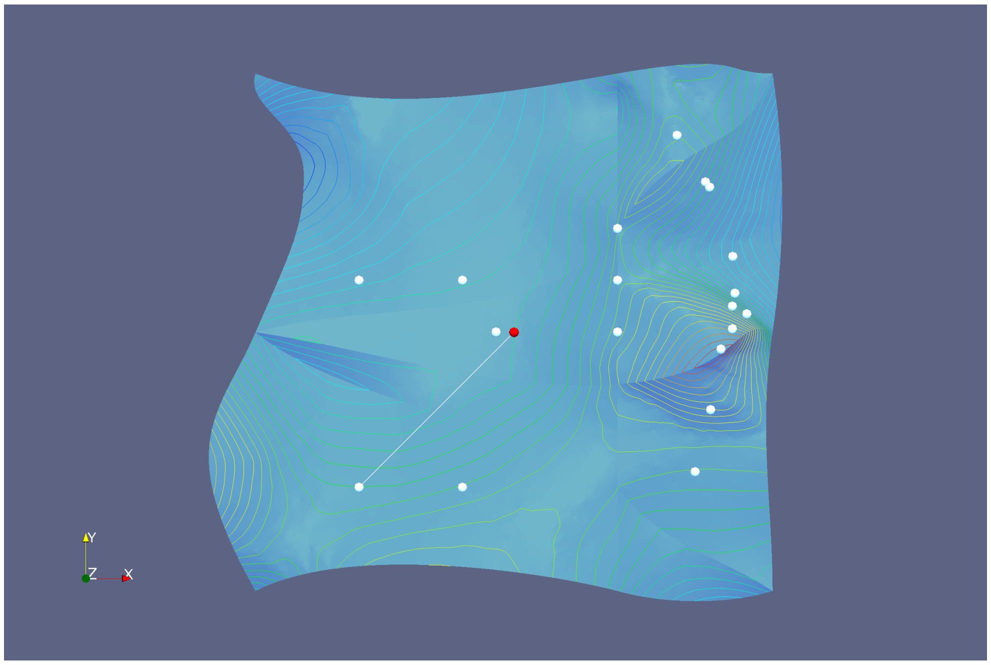


Figure 6.3 Sensor locations and object function sample line for mountainous topology

To further investigate the possibility of encountering local minima, a sample line is taken between the starting location of the design cycles and the target location. Samples of the objective function value are taken at discrete locations along this line. The results of this survey are presented in Figure 6.4. Clearly, a local minima exist along the sample line at (-0.5, -0.5). This minima exists due to the dense packing of sensors near the height of a prominent feature in the terrain shown in Figure 6.3 and the fact that very few of these are exposed in the target location due to terrain steering shown in Figure 6.2. More importantly, however, these minima exist across the entire design space. This implies that with the objective function selected, gradient based design methods are not guaranteed to predict the original plume location given discrete sensor data.

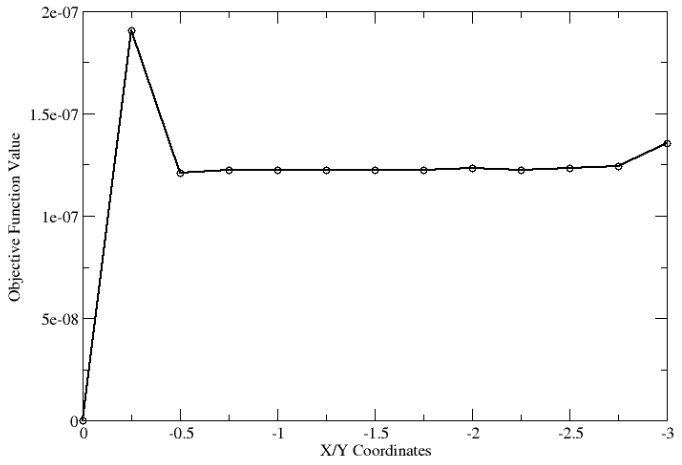


Figure 6.4 Line search objective function values

As an illustrative example of this phenomenon, shown in Figure 6.5, consider a sensor field which contains only 3 sensors. Sensor 1 is directly behind the original plume location and gives a strong signal in the target case. Sensors 2 and 3 at approximately the same downstream distance as sensor 1 but are two plume widths to one side of Sensor 1. Consider a potential starting point for the plume that would give a strong signal at sensor 2 but not at sensors 1 or 3. As the plume moves across the field in the lateral direction the objective function value is high (sensor 3 active), then lower (between sensors 2 and 3), then high (sensor 2 active), lower again (between sensors 1 and 2), and finally minimized as sensor 1 becomes active. The difficulty arises from the discontinuous nature of the plume and the discrete sensing of flowfield parameters. The combination of the two results in an objective function which does not uniformly move towards a minimum at the target location.

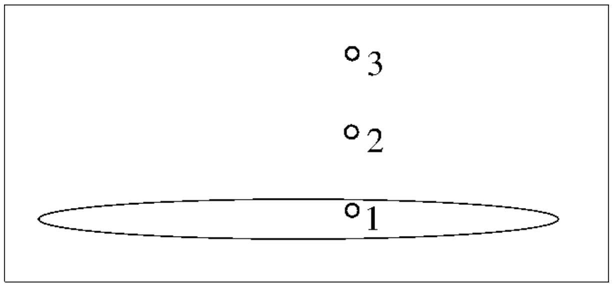


Figure 6.5 Illustration of pathologically constructed example case

Configurations do exist for plume magnitude, terrain, sensor location, wind direction, etc. under which gradient based design methods may be successfully applied to plume inversion. However, the work performed in this study suggest those cases are certainly less common than circumstances for which the method fails. A more global optimization method, such as a genetic algorithm or one based on response surfaces, should be employed due to their ability to traverse local minima.

CHAPTER 7

WASHINGTON DC PLUME STUDY

The target problem is the plume release of Sarin nerve agent near the Washington DC Mall. The geometry used in this case, shown in Figure 7.1 consists of the surface topology as well as the structures which comprise the area surrounding the Mall for several blocks in each direction. The mesh contains approximately 15.5 million nodes and 61 million cells. Resolution from the wall is approximately 1 meter and is shown if Figure 7.2. A Spalart-Allmaras turbulence model is enabled to compute turbulent viscosity. A freestream flow velocity of 3.8 m/s is imposed at all external boundaries. Flow enters from a West-Northwest direction (285 degree azimuth). Atmospheric boundary layer velocity profiles are imposed at these boundaries and follow a power law distribution. This avoids numerical instabilities encountered due to a viscous wall intersecting a farfield boundary condition. Average flow velocity at ground level is approximately 2 m/s due to obstruction effects as shown in Figure 7.3. Molecular diffusion is not simulated here as primary mixing is assumed through large scale turbulent effects. Sarin is introduced following a Gaussian distribution in density upstream of the Washington Monument. The maximum density of Sarin vapor is 1.15e-3 kg/m^3 with Gaussian distribution parameters $\sigma_x = \sigma_y = 6$ m. The maximum injection velocity is set equal to freestream. This set of parameters gives an injection mass flow rate of approximately 1.1 tons per second. A 300 pound TNT equivalent explosive does contain sufficient energy to vaporize this quantity of Sarin agent upon detonation. In this case the simulation is run to steady state with a constant injection mass flow. It is unlikely that a delivery mechanism could be conceived to maintain this mass flow for an extended duration. However, the examples shown here are still of interest to investigate plume behavior.

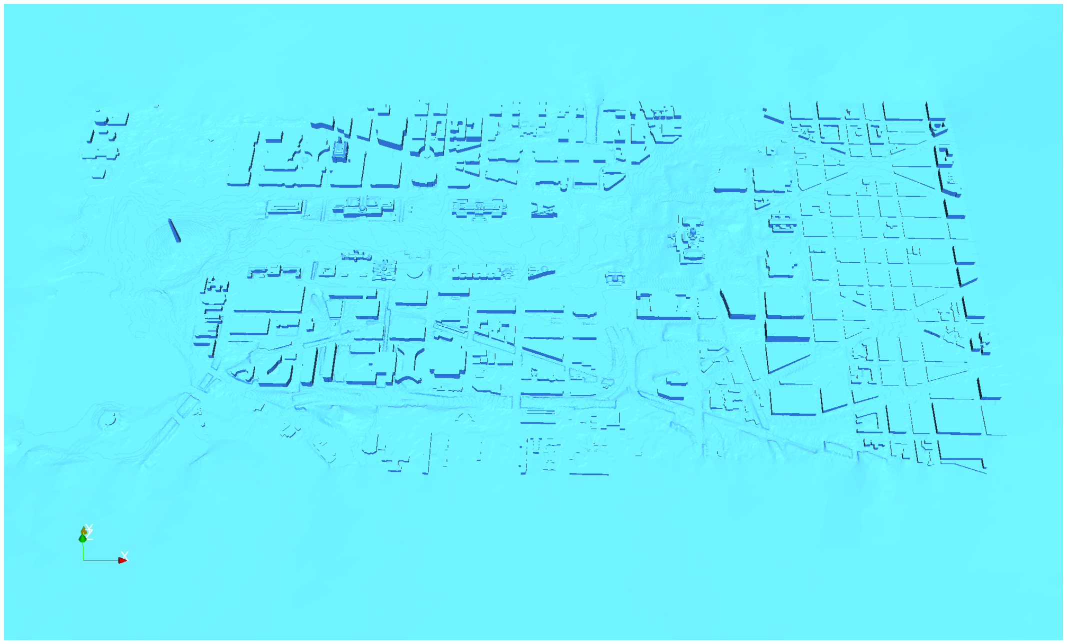


Figure 7.1 Geometry of Washington DC Mall area

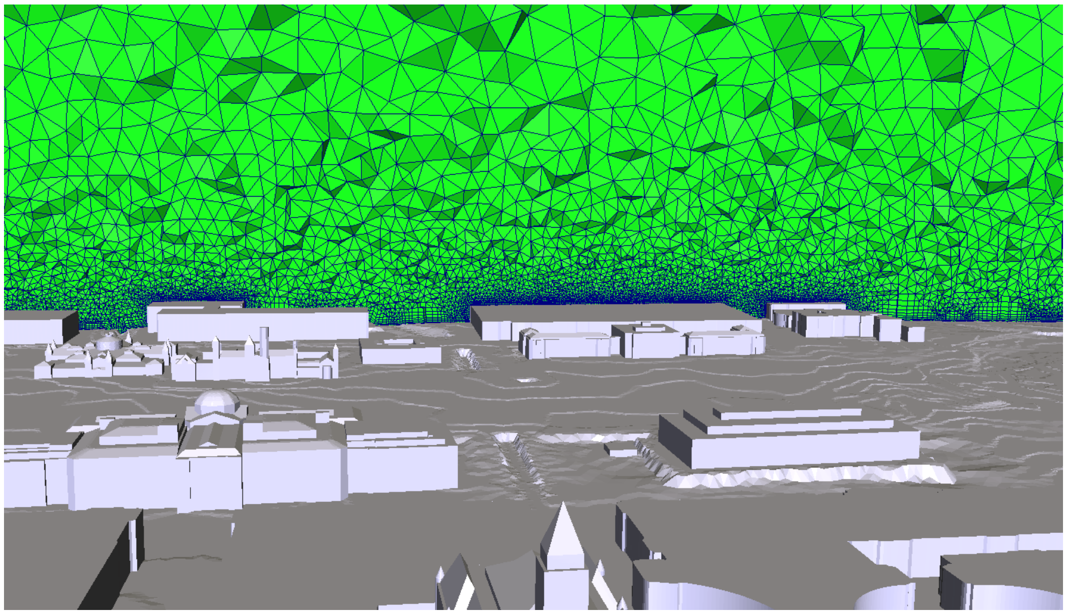


Figure 7.2 Illustration of mesh node clustering near boundary layer

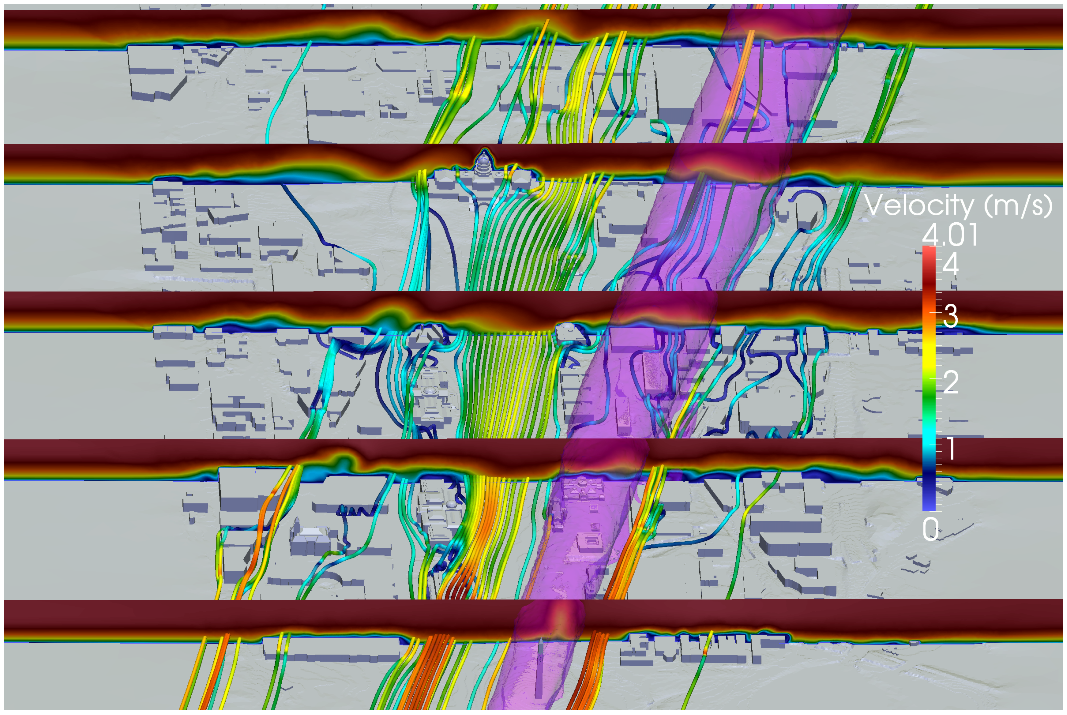


Figure 7.3 Illustration of velocity profile near ground level

Sarin vapor is assumed to transform into a secondary species as a result of exposure to OH radicals in the urban environment. Sarin is chemically similar to many organophosphates utilized as pesticides and a considerable amount of data is available on their decomposition under exposure to OH radicals in urban atmospheres [56, 57]. Chapter 5 covers this particular agent in more detail. The transformation is assumed to take place in an unstable way in this work resulting in a fast rate of 69.7e-6 mol/m^3·s. Figure 7.4 shows density isosurfaces for Sarin and the byproduct at an acute exposure guideline level (AEGL) 1 for 30 minutes [54]. Figure 7.5 shows the density isosurface for Sarin only. It is apparent that the combination of reducing density (due to diffusion) and obstructions act to loft the plume shortly after encountering the leading edge of the cityscape.

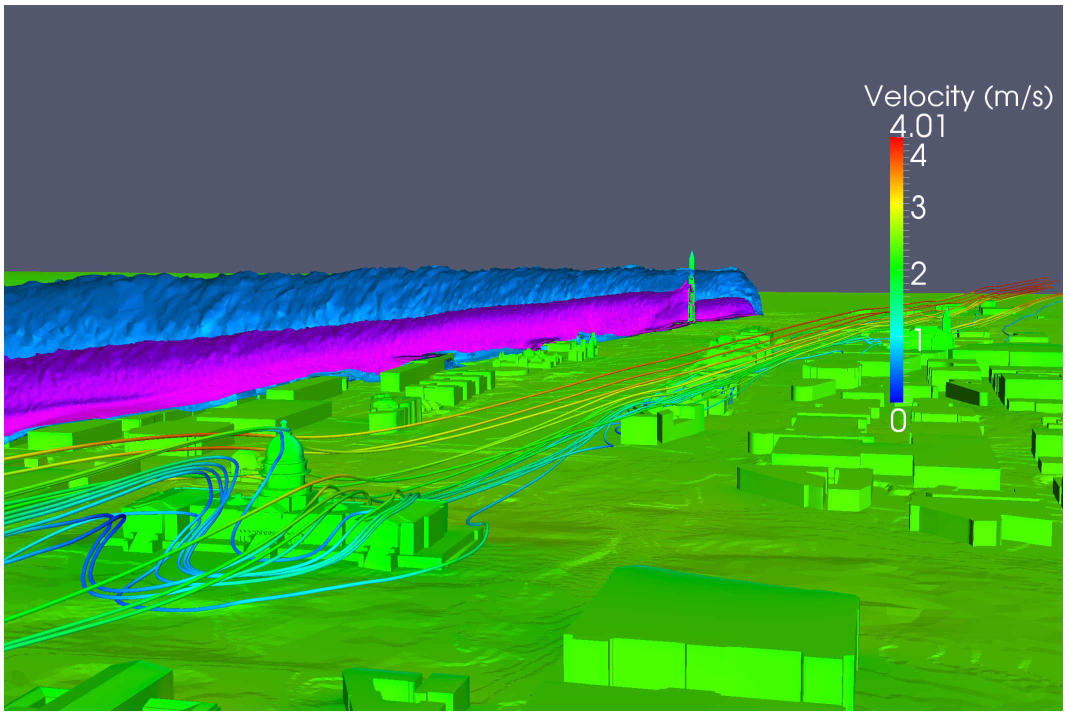


Figure 7.4 Sarin and OH reaction byproduct density isosurfaces - Sarin (blue) and OH reduced byproduct (pink)

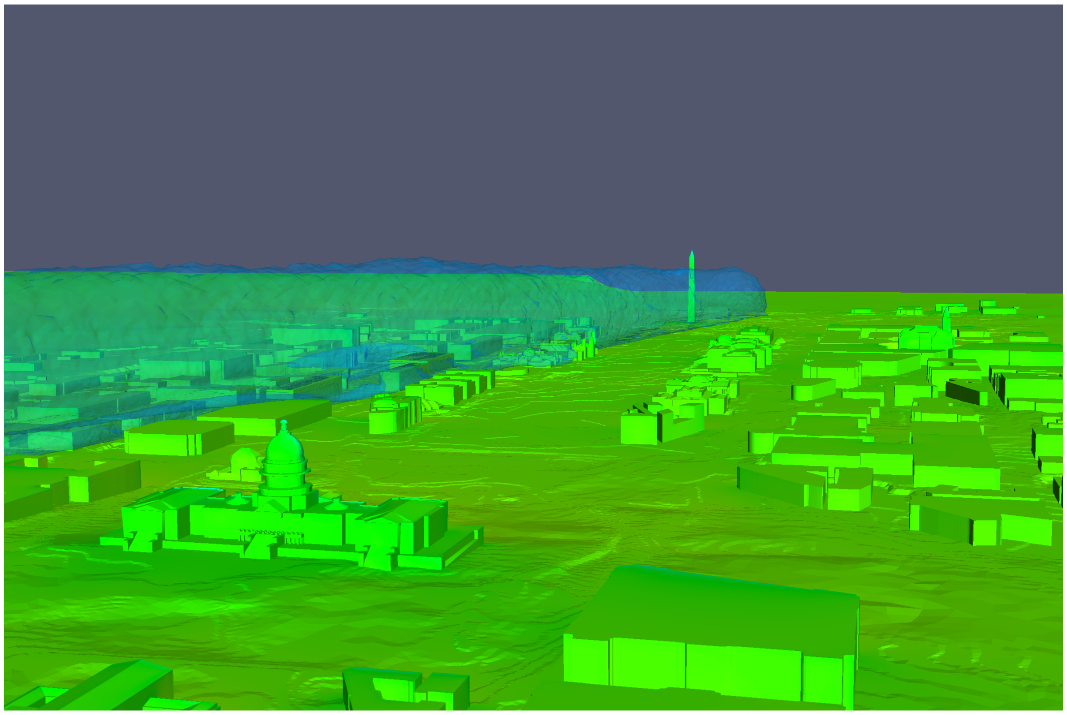


Figure 7.5 Sarin density isosurface at AEGL 1 for 30 minutes

Figure 7.6 shows a visualization of streamlines impinging on the edge of the defined city geometry. The fluid takes a complex and convoluted path between buildings and obstructions making *a priori* transport predictions difficult. Low velocity flow in and around obstructions demonstrates the possibility for heavy gas plumes to become entrapped for extended periods of time in low-lying areas prolonging time for which vapor presents a hazard.

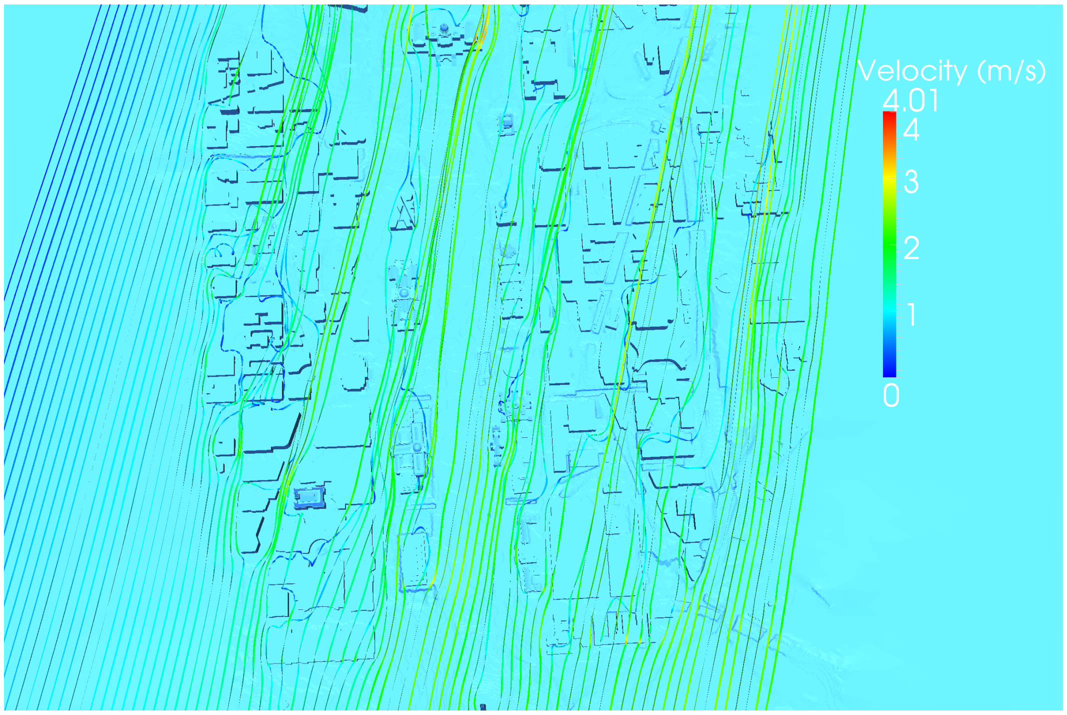


Figure 7.6 Overview of streamline interaction in Washington DC geometry

Figure 7.7 shows streamlines impinging on the edge of the defined geometry. Narrow gaps between structures are shown to convect fluid in a crossflow direction and then reintroduce it into the bulk fluid flow at some distance away. This action demonstrates that contaminants introduced in a particular location have the potential to be transported in a highly non-uniform way. This observation leads to the conclusion that terrain resolving CFD analysis is essential to understand the potential for heavy gas releases to affect surrounding areas. Methods which do not resolve street level features are not capable of correctly predicting this transport.

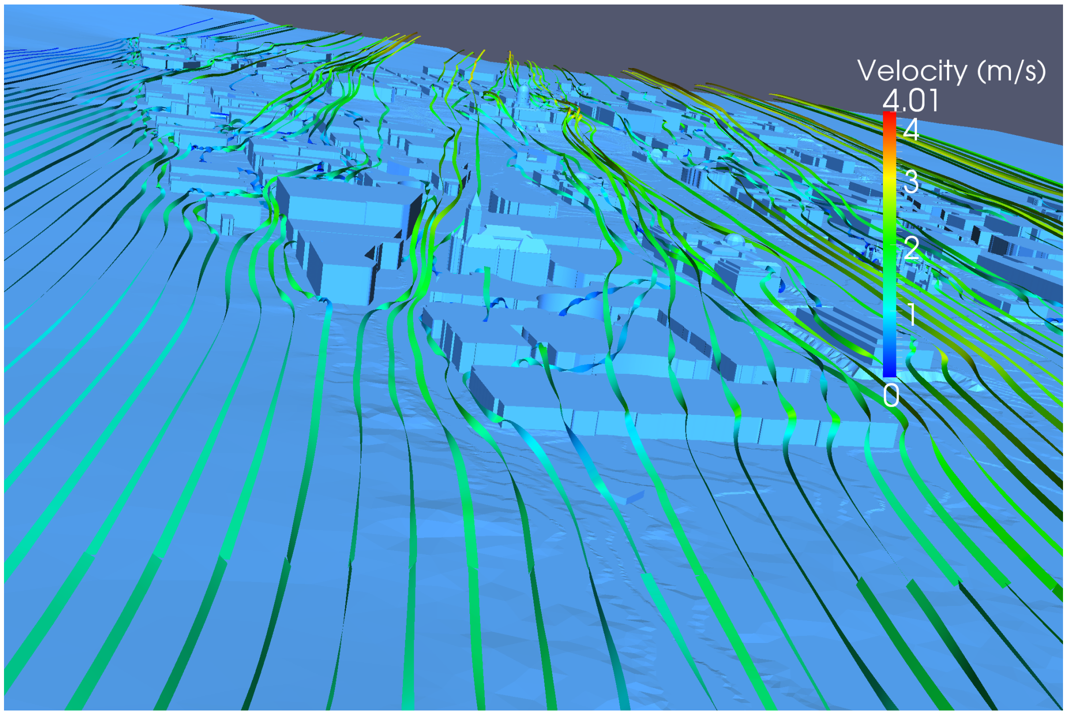


Figure 7.7 Visualization of fluid mixing on city boundary

Figure 7.8 shows AEGLs for a 30 minute sustained exposure at ground level for the simulation described above. AEGL 1 (yellow) corresponds to noticeable symptoms appearing in victims of the disaster. AEGL 2 (orange) corresponds the onset of permanent or lasting effects in response to exposure to an airborne contaminant. AEGL 3 (red) areas correlate to fatal exposure levels in the general population upon sustained contact with an airborne contaminant.

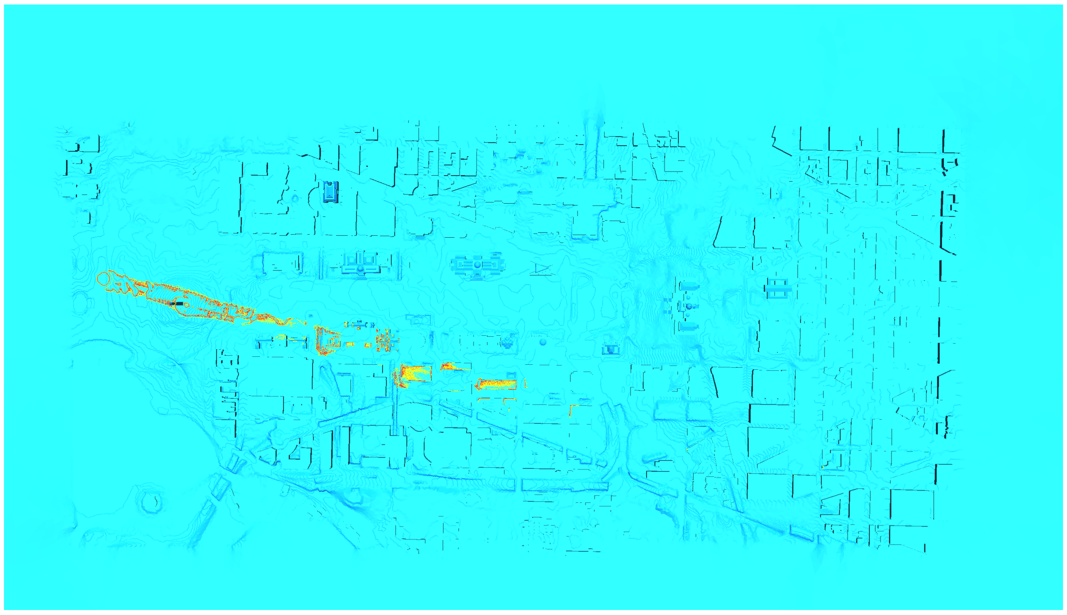


Figure 7.8 Washington DC Sarin attack 30 minute AEGL contour lines.

Figure 7.9 shows AEGLs for a 10 minute sustained exposure at ground level for the above simulation. This exposure period is far shorter and possibly more realistic as individuals are unlikely to remain in the area once the onset of symptoms begins. Unfortunately, the prognosis does not markedly improve with reduced exposure time. Sarin's toxicity level is so high that a reduction of exposure time by 20 minutes only doubles the survivable exposure concentration. Again, wind facing structure with significant height pose the greatest threat to personal safety as contaminant becomes trapped by prevailing winds.

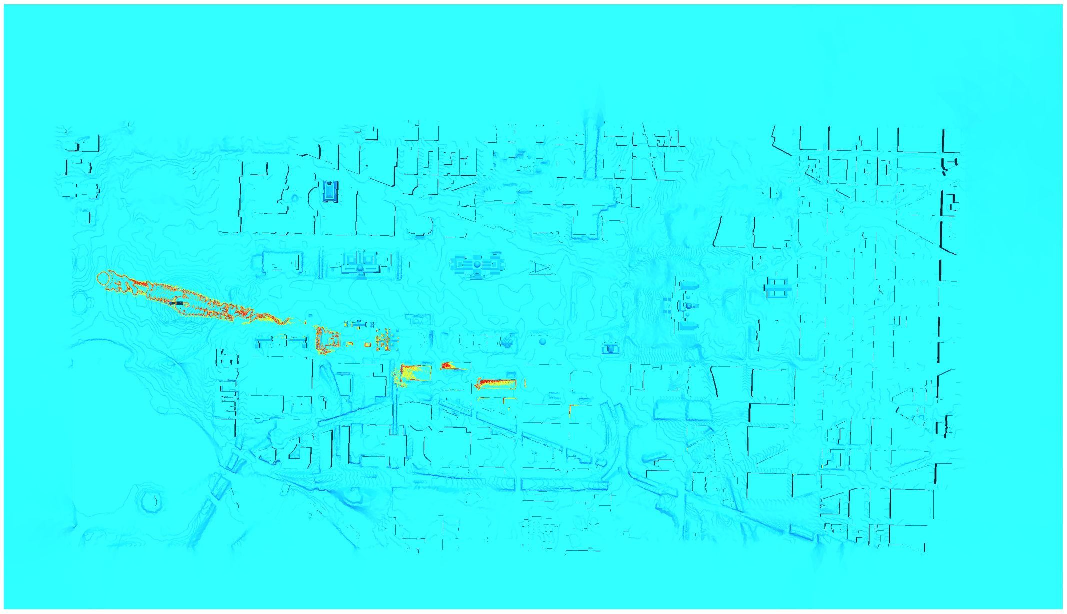


Figure 7.9 Washington DC Sarin attack 10 minute AEGL contour lines.

In this simulation, the presence of terrain as well as structures significantly affects survivability in areas which may be non-obvious. In particular, the upstream face of many buildings far downstream of the plume lofting are not survivable locations. Unexpectedly, several locations very near the release source are, in fact, relatively safe by comparison. However, much of the physics involved in reliable prediction of these zones such as atmospheric inversion, solar heating, etc., are neglected in this simulation. This work does highlight the need, however, for further study with refined physics models in an attempt to better characterize safe zones in the event of such an occurrence.

The high toxicity and relative ease of acquisition of Sarin nerve agent creates a hazard which must be considered carefully in disaster planning exercises undertaken by both federal and local governments. Though the simulation shown here is far beyond the scope of realizability, the exercise does highlight the need for increased understanding and study of complex obstructed flows. The above simulation consumes 160 cores of a 3.0 GHz Xeon ("Woodcrest") cluster for approximately 6 days to obtain plume convergence. Parametric studies on this scale of geometry will require computer resources of at least an order of magnitude larger to become practical. These resources were not available for this work.

CHAPTER 8

CONCLUSIONS

A novel low-Mach preconditioner is developed and applied to the Reynolds averaged finite-rate Navier-Stokes equations. An original formulation for the eigensystem of the finite-rate Navier-Stokes equations has additionally been derived. This work allowed for the simulation of a full urban geometry with reacting plume. By retaining a density-based solver structure, this framework allows for the investigation of explosive events in the presence of low wind velocities in future research. This capability is unique in the literature to the knowledge of this researcher. Additionally, an efficient method for computing design derivatives is presented for reacting flowfields. High accuracy of computed derivatives is demonstrated for several test problems. Finally, an "inverse chemistry" problem is attempted on a complex terrain with the sensitivity information computed. It is found that for random sensor locations the objective function selected has many local minima. This, in turn, makes gradient based design methods potentially unsuitable for the investigation of this class of problems. Global optimization algorithms will likely prove more robust in this regard.

CHAPTER 9

FUTURE WORK

The current research represents a feasibility study to assess the technology necessary to detect, predict, and deploy resources in the event of a chemical spill or terror event. The ability of gradient based design methods to predict dispersal locations based on sensor data alone appears problematic based on the results shown. Global optimization methods such as genetic algorithms, etc. are not as sensitive to local minima and may prove useful in this search. However, the computational cost and lead time shown in this study are prohibitive for this technology to be deployed in that way. It does not appear that the required scale of computational resource will be cost justified in the near future. Precomputed databases can decrease the response lead time in the event of catastrophe. Unfortunately, these methods typically do not reduce the initial cost of performing predictions nor the difficulties in acquiring meaningful and accurate geometry. Therefore, additional research is required to reduce the overall computational costs before this technology may be routinely used to reduce casualties.

The physical fidelity of the toolchain described herein is an area to which significant improvements could be made. Though a great deal of effort was made to consider the physics relevant to the problem of flows in urban environments, many simplifications were made in an effort to reduce the scope of this work. In particular, the number of tracked species very quickly out scaled the available computational resources. Additionally, the removal of ground obstruction interactions (trees, cars, people, etc.), solar heating based on surface finish, turbulence model studies, atmospheric inversion layers, unsteady flow, photochemistry, reaction pathways, and turbulent reaction rate effects could also be significant in the predictive model. While the exclusion of such effects does not invalidate the work presented herein, these effects are necessary for a accurate prediction of plume progression and therefore the correlation to sensor data in a “real” use case. The fidelity of atmospheric models is a well studied problem and the reader is directed to [59, 60, 61] for a discussion of model fidelity.

Finally, the application of the toolchain developed herein to problems more amenable to gradient based design optimization is expected. Shape design for combustors to reduce emissions or improve performance is of particular interest. Both the reactions involved as well as the complexity of the flowfield make this area rich for contribution in the future.

REFERENCES

[1] Okudera, H., Morita, H., Iwashita, T., Shibata, T., Otagiri, T., Kobayahsi, S., and Yanagisawa, N., "Unexpected Nerve Gas Exposure in the City of Matsumoto: Report of Rescue Activity in the First Sarin Gas Terrorism," *American Journal of Emergency Medicine*, Vol. 15, 1997, pp. 527–528. 1, 100

[2] Okumura, T., Takasu, N., Ishimatsu, S., Miyanoki, S., Mitsuhashi, A., Kumada, K., Tanaka, K., and Hinoshara, S., "Report on 640 Victims of the Tokyo Subway Sarin Attack," *Annals of Emergency Medicine*, Vol. 28, 1996, pp. 129–35. 1, 99

[3] Wenck, M., Sickle, D. V., Drociuk, D., Belflower, A., Youngblood, C., Whisnant, M., Taylor, R., Rudnick, V., and Gibson, J., "Rapid Assessment of Exposure to Chlorine Released From a Train Derailment and Resulting Health Impact," Tech. rep., Public Health Report, Nov-Dec 2007. 1

[4] Chow, F. K., Kosovic, B., and Chan, S. T., "Source Inversion for Contaminant Plume Dispersion in Urban Environments Using Building-Resolving Simulations," *86th American Meteorological Society Annual Meeting*, No. UCRL-CONF-216903, 2006. 2, 3, 4

[5] Gelman, A., Karlin, J., Stern, H., and Rubin, D., *Bayesian Data Analysis (2nd Edition)*, Chapman & Hall, Boca Raton, FL, 2003. 2

[6] Enting, I. G., *Inverse Problems in Atmospheric Constituent Transport*, Wiley-Interscience, 2002. 2

[7] Chen, H., "Greedy Methods in Plume Detection, Localization and Tracking," *Advances in Greedy Algorithms*, edited by W. Bednorz, InTech, Vienna, Austria, 2008. 2

[8] Huang, C., Hsing, T., Cressie, N., Ganguly, A., Protopopescu, V., and Rao, N., "Bayesian Source Detection and Parameter Estimation of Plume Model Based on Sensor Network Measurements," *Applied Stochastic Models in Business and Industry*, Vol. 26, 2010. 2

[9] Michaelides, M. and Panayiotou, C., "Plume Source Position Estimation Using Sensor Networks," *13th Mediterranean Conference on Control and Automation*, June 2005. 3

[10] Newman, J. C., *Integrated Multidisciplinary Design Optimization Using Discrete Sensitivity Derivatives for Geometrically Complex Aeroelastic Configurations*, PhD dissertation, Virginia Polytechnic Institute and State University, July 1997. 4

[11] Meroney, R. N., "CFD Prediction of Dense Gas Clouds Spreading in a Mock Urban Environment," *International Symposium on Computational Wind Engineering (CWE2010)*, May 2010. 10

[12] Gavelli, F., Bullister, E., and Kytomaa, H., "Application of CFD (Fluent) to LNG spills into Geometrically Complex Environments," *Journal of Hazardous Materials*, , No. 159, 2008, pp. 158–168. 10

[13] Kee, R. J., Coltrin, M. E., and Glarborg, P., *Chemically Reacting Flow : Theory and Practice*, Wiley-Interscience, 2003. 12, 60

[14] Ramshaw, J., "Self-consistent Effective Binary Diffusion in Multicomponent Gas Mixtures," *Journal of Non-Equilibrium Thermodynamics*, Vol. 15, 1990, pp. 295–300. 13

[15] Williams, F., *Combustion Theory*, Addison-Wesley, 1985. 13

[16] Subramaniam, S., "Minimum Error Fickian Diffusion Coefficients for Mass Diffusion in Multicomponent Gas Mixtures," *Journal of Non-Equilibrium Thermodynamics*, Vol. 24, 1999, pp. 1–39. 14

[17] Quarteroni, A., Sacco, R., and Saleri, F., *Numerical Mathematics*, Springer, 2007. 15

[18] Gupta, A., *Preconditioning Methods for Ideal and Multiphase Fluid Flows*, PhD dissertation, University of Tennessee at Chattanooga, 2013. 16, 21, 27, 86, 88

[19] Eriksson, L., "A Preconditioned Navier-Stokes Solver for Low Mach Number Flows," *Proceedings of the Third ECCOMAS Computational Fluid Dynamics Conference*, Paris, France, 1996. 16

[20] Busby, M. A. and Cinnella, P., "Nonsingular Eigenvectors of the Flux Jacobian Matrix for Reactive Flow Problems," *AIAA Journal*, Vol. 37, No. 3, pp. 398–401. 23, 24

[21] Grossman, B. and Cinnella, P., "Flux-Split Algorithms for Flows with Nonequilibrium Chemistry and Vibrational Relaxation," *Journal of Computational Physics*, Vol. 88. 24

[22] Roe, P. L., "Characteristic-Based Schemes for the Euler Equations," *Annual Review of Fluid Mechanics*, Vol. 18, 1986, pp. 337–365. 24

[23] Harten, A., Lax, P. D., and van Leer, B., "On Upstream Differencing and Godunov-type Schemes for Hyperbolic Conservation Laws," *SIAM Review*, Vol. 25, 1983, pp. 35–61. 24

[24] Buvaneswari, J., *Accurate Upwind Procedures for Real-Gas Flows in Equilibrium and Non-Equilibrium*, Ph.D. thesis, Indian Institute of Science, 1998. 24

[25] Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics - A Practical Introduction*, Springer - Verlag, 1999. 24

[26] Sreenivas, K., Taylor, L. K., and Briley, W. R., "A Global Preconditioner for Viscous Flow Simulations at All Mach Numbers," *24th Applied Aerodynamics Conference*, June 2006. 27, 28, 88

[27] Unrau, D. and Zingg, D., "Viscous Airfoil Computations Using Local Preconditioning," *13th Computational Fluid Dynamics Conference*, June 1997. 27, 28

[28] Hyams, D. G., *An Investigation of Parallel Implicit Solution Algorithms for Incompressible Flows on Unstructured Topologies*, PhD dissertation, Mississippi State University, May 2000. 30, 31, 33, 136

[29] Blazek, J., *Computational Fluid Dynamics: Principles and Applications*, Elsevier Science Ltd., 2001. 30

[30] Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady State Solutions," No. AIAA Paper no. 93-0880, January 1993. 35

[31] Barth, T. and Jespersen, D., "The Design and Application of Upwind Schemes on Unstructured Meshes," No. AIAA Paper No. 89-0366, 9189. 35

[32] Venkatakrishnan, V., "Convergence to Steady-State Solutions of the Euler Equations on Unstructured Grids with Limiters," *Journal of Computational Physics*, Vol. 118, 1995, pp. 120–130. 35

[33] Michalak, K. and Olliver-Gooch, C., "Limiters for Unstructured Higher-Order Accurate Solutions of the Euler Equations," *46th AIAA Aerospace Sciences Meeting*, January 1996. 36

[34] Anderson, W. K. and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers & Fluids*, Vol. 23, No. 1, 1994, pp. 1–21. 41

[35] Newman, III, J., Whitfield, D., and Anderson, W. K., "A Step-Size Independent Approach for Multidisciplinary Sensitivity Analysis," *Journal of Aircraft*, Vol. 40, No. 3, May-June 2003, pp. 566–573. 49, 68, 76

[36] Burcat, A. and Ruscic, B., "Third Millennium Ideal Gas and Condensed Phase Thermochemical Database for Combustion with Updates from Active Thermochemical Tables," Tech. rep., Argonne National Laboratory, 2005. 55, 100

[37] Gupta, R. N., Lee, K.-P., Thompson, R. A., and Yos, J. M., "Calculations and Curve Fits of Thermodynamic and Transport Properties for Equilibrium Air to 30 000 K," Tech. rep., NASA, 1991. 55

[38] Cox, C. F., *An Efficient Solver for Flows in Local Chemical Equilibrium*, Ph.D. thesis, Mississippi State University, 1992. 57

[39] Busby, M. A., *Steps Toward More Accurate and Efficient Simulations of Reactive Flows*, PhD dissertation, Mississippi State University, August 1997. 58

[40] Gordon, S. and McBride, B. J., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications, I," *NASA Reference Publication 1311*, October 1994. 58

[41] Gordon, S. and McBride, B. J., "Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications, II," *NASA Reference Publication 1311*, June 1996. 58

[42] White, F. M., *Viscous Fluid Flow*, McGraw-Hill, 2006. 59

[43] Gaffney, R. L., White, J. A., Girimaji, S. S., and Drummond, J. P., "Modeling Turbulent/Chemistry Interactions Using Assumed PDF Methods," *28th AIAA, SAE, ASME, and ASEE, Joint Propulsion Conference*, July 1992. 59

[44] Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *Recherche Aerospatiale*, , No. 1, 1994, pp. 5–21. 63

[45] Spalart, P., "Trends in Turbulence Treatments," No. AIAA-2000-2306, June 2000. 63

[46] Anderson, W. K., Newman, J. C., Whitfield, D. L., and Nielsen, E. J., "Sensitivity Analysis for Navier-Stokes Equations on Unstructured Meshes Using Complex Variables," *AIAA Journal*, Vol. 39, No. 1, January 2001, pp. 56–63. 68, 76

[47] Burg, C. and Newman, III, J., "Computationally Efficient, Numerically Exact Design Space Derivatives via the Complex Taylor's Series Expansion Method," *Computers & Fluids*, Vol. 32, 2003, pp. 373–383. 68, 75

[48] Sherman, L., Taylor, A., Green, L., Newman, P., and Gene Hou, V. K., "First- and Second-Order Aerodynamic Sensitivity Derivatives via Automatic Differentiation with Incremental Iterative Methods," *Journal of Computational Physics*, Vol. 129, 1996, pp. 307–331. 73, 74, 75

[49] Karman, S., Anderson, W. K., and Sahasrabudhe, M., "Mesh Generation Using Unstructured Computational Meshes and Elliptic Partial Differential Equation Smoothing," *AIAA Journal*, Vol. 44, No. 6, June 2006. 78

[50] Nielsen, E. J. and Kleb, W. L., "Efficient Construction of Discrete Adjoint Operators on Unstructured Grids by Using Complex Variables," *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005. 79

[51] Burdyshaw, C., *Achieving Automatic Concurrency Between Computational Field Solvers and Adjoint Sensitivity Codes*, PhD dissertation, The University of Tennessee at Chattanooga, May 2006. 79, 83

[52] Hou, G.-W., Taylor, III, A., and Korivi, V., "Discrete Shape Sensitivity Equations for Aerodynamic Problems," *International Journal for Numerical Methods in Engineering*, Vol. 13, 1994, pp. 2251–2266. 83

[53] Gupta, R. N., Yos, J. M., Thompson, R. A., and Lee, K.-P., "A Review of Reaction Rates and Thermodynamic and Transport Properties for an 11-Species Air Model for Chemical and Thermal Nonequilibrium Calculations to 30 000 K," Tech. rep., NASA, 1990. 90

[54] Subcommittee on the Acute Exposure Guideline Levels, "Acute Exposure Guideline Levels for Selected Airborne Chemicals," Tech. rep., National Research Council of the National Academies, 2003. 99, 100, 112

[55] Campbell, L. E. and Pappas, A. G., "Domestic Preparedness: Sarin Vapor Challenge and Corn Oil Protection Factor (PF) Testing of Powered Air Purifying Respirator (PAPR) Systems and Cartriges," Tech. rep., Army Research, Development and Engineering Command, Edgewood Chemical Biological Center, March 1999. 100

[56] Elliott, S., Streit, G. E., Gaffney, J. S., Bossert, J. E., Brown, M., Reisner, J., and McNair, L. A., "Pathways for the Oxidation of Sarin in Urban Atmospheres," Tech. Rep. LA-13501-MS, Los Alamos National Laboratory, November 1995. 100, 112

[57] Winer, A. M. and Atkinson, R., *Atmospheric Reaction Pathways and Lifetimes for Organophoshorous Compounds*, Lewis Publishers: Chelsea, MI, 1990. 101, 112

[58] Buchanan, J., Sumpter, K., Abercrombie, P., and Tevault, D., “Vapor Pressure of GB,” Tech. Rep. ECBC-TR-686, US Army Research, Development and Engineering Command: Edgewood Chemical Biological Center, April 2009. 101

[59] Patnaik, G., Boris, J., Grinstein, F., and Iselin, J., “Large Scale Urban Simulations with the Miles Approach,” *16th AIAA Computational Fluid Dynamics Conference*, June 2003. 121

[60] Patnaik, G., Grinstein, F., Boris, J., and Young, T., “Large Scale Urban Contaminant Transport Simulations with Miles,” *Journal of Fluids Engineering*, Vol. 129, 2007. 121

[61] Young, T., Boris, J., Hooper, S., Lind, C., Obenschain, K., and Patnaik, G., “Emergency Assessment with Sensors and Buildings: Advances in CT-ANALYST Technology,” *Chemical Biological Information Systems*, October 2004. 121

APPENDIX A

DEFINITION OF TRANSFORMATION JACOBIANS

This appendix describes the process by which the transformation Jacobians are derived. These terms appear due to the choice of using nonconservative variables in lieu of maintaining the conservative variables that are required for a fully conservative flux formulation.

The mapping Jacobian for the finite-rate regime is written as

$$\frac{\partial Q}{\partial q_p} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \dots & 0 \\ u & u & \dots & \rho & 0 & 0 & 0 \\ v & v & \dots & 0 & \rho & 0 & 0 \\ w & w & \dots & 0 & 0 & \rho & 0 \\ \frac{\partial \rho e_t}{\partial \rho_1} & \frac{\partial \rho e_t}{\partial \rho_2} & \dots & \rho u & \rho v & \rho w & \frac{\partial \rho e_t}{\partial p} \end{bmatrix} \tag{A.1}$$

The total energy term can be expressed as

$$\rho e_t = \sum_{j=1}^{NS} \rho_j h_j + \frac{1}{2} \sum_{j=1}^{NS} \rho_j \|\vec{U}\|^2 - p \tag{A.2}$$

where $\|\vec{U}\|$ is the magnitude of the velocity vector. The partial derivative of total energy with respect to a species partial density is

$$\frac{\partial \rho e_t}{\partial \rho_i} = h_i + \frac{1}{2}\|\vec{U}\|^2 - \frac{\partial}{\partial \rho_i}(p) \tag{A.3}$$

where the partial derivative of pressure with respect to the ith species density with an equation of state of the form $p = p(\rho, R, T)$ is

$$\begin{aligned}\frac{\partial}{\partial \rho_i}(p) &= \frac{\partial p}{\partial T} + \frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial \rho_i} + \frac{\partial p}{\partial R}\frac{\partial R}{\partial \rho_i} \\ &= \frac{\partial p}{\partial \rho} + \frac{\partial p}{\partial R}\frac{\partial R}{\partial \rho_i}\end{aligned} \tag{A.4}$$

and defining

$$R = \sum_{i=1}^{NS} \frac{\rho_i}{\rho} R_i \tag{A.5}$$

with

$$\frac{\partial R}{\partial \rho_i} = R_i \frac{\rho - \rho_i}{\rho^2} - \sum_{j \neq i}^{NS} \frac{\rho_j R_j}{\rho^2} \tag{A.6}$$

Making the appropriate substitutions, without loss of generality for varying equations of state, yields

$$\frac{\partial \rho e_t}{\partial \rho_i} = h_i - \frac{1}{2}\|\vec{U}\|^2 - \frac{\partial p}{\partial \rho} - \frac{\partial p}{\partial R}\left(R_i \frac{\rho - \rho_i}{\rho^2} - \sum_{j \neq i}^{NS} \frac{\rho_j R_j}{\rho^2}\right) \tag{A.7}$$

In the case of an ideal gas

$$p = \rho R T \tag{A.8}$$

The following derivatives can be defined for the ideal gas equation of state

$$\frac{\partial p}{\partial \rho} = RT \tag{A.9}$$

$$\frac{\partial p}{\partial R} = \rho T \tag{A.10}$$

and making the substitutions into Equation A.7 yields

$$\frac{\partial \rho e_t}{\partial \rho_i} = h_i - \frac{1}{2}\|\vec{U}\|^2 - RT - \rho T\left(R_i \frac{\rho - \rho_i}{\rho^2} - \sum_{j \neq i}^{NS} \frac{\rho_j R_j}{\rho^2}\right) \tag{A.11}$$

for ideal gases.

A similar procedure is used to arrive at the partial derivative of total energy with respect to pressure.

$$\frac{\partial \rho e_t}{\partial p} = \sum_{j=1}^{NS} \rho_j \frac{\partial h_j}{\partial T} \frac{\partial T}{\partial p} - 1 \tag{A.12}$$

Equation 2.110 also requires the definition of

$$\frac{\partial q_p}{\partial q_T} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 & 0 & 0 & 0 \\ 0 & 0 & \dots & 0 & 1 & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \frac{\partial p}{\partial T} \end{bmatrix} \tag{A.13}$$

The following relation eliminates the need to explicitly form $\frac{\partial Q}{\partial q_T}$ and $\frac{\partial q_T}{\partial q_p}$

$$\frac{\partial Q}{\partial q_T} \frac{\partial q_T}{\partial q_p} = \frac{\partial Q}{\partial q_p} \tag{A.14}$$

which is required for the temporal discretization shown in Equation 2.110.

APPENDIX B

LEAST SQUARES GRADIENTS

The least squares formulation can be derived by considering a vertex i and its surrounding neighbors j along with their locations in space. Following the dissertation of Hyams [28], using a first order Taylor series expansion, the solution at a node which is a neighbor to i can be written as

$$Q_j = Q_i + (\vec{x_j} - \vec{x_i}) \cdot \nabla Q_i \tag{B.1}$$

This equation can be expressed in a linear system for each neighbor of node i.

$$\begin{bmatrix} \Delta x_1 & \Delta y_1 & \Delta z_1 \\ \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \Delta x_3 & \Delta y_3 & \Delta z_3 \\ \vdots & \vdots & \vdots \\ \Delta x_N & \Delta y_N & \Delta z_N \end{bmatrix} \begin{bmatrix} Q_{xi} \\ Q_{yi} \\ Q_{zi} \end{bmatrix} \begin{bmatrix} Q_1 - Q_i \\ Q_2 - Q_i \\ Q_3 - Q_i \\ \vdots \\ Q_N - Q_i \end{bmatrix} \tag{B.2}$$

Equation B.2 may be modified with the addition of weights to arrive at the weighted least squares formulation. Both the left and right hand sides are are multiplied by weight w.

$$\begin{bmatrix} w_1\Delta x_1 & w_1\Delta y_1 & w_1\Delta z_1 \\ w_2\Delta x_2 & w_2\Delta y_2 & w_2\Delta z_2 \\ w_3\Delta x_3 & w_3\Delta y_3 & w_3\Delta z_3 \\ \vdots & \vdots & \vdots \\ w_N\Delta x_N & w_N\Delta y_N & w_N\Delta z_N \end{bmatrix} \begin{bmatrix} Q_{xi} \\ Q_{yi} \\ Q_{zi} \end{bmatrix} \begin{bmatrix} w_1(Q_1 - Q_i) \\ w_2(Q_2 - Q_i) \\ w_3(Q_3 - Q_i) \\ \vdots \\ w_N(Q_N - Q_i) \end{bmatrix} \tag{B.3}$$

where $\vec{w}$ is a vector of weights. In this case, an inverse distance weighting is used where

$$w_j = \frac{1}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2}} \tag{B.4}$$

This can be rewritten as

$$\begin{bmatrix} M_1 & M_2 & M_3 \end{bmatrix} \begin{bmatrix} Q_{xi} \\ Q_{yi} \\ Q_{zi} \end{bmatrix} = [\tilde{Q}] \tag{B.5}$$

where the notation $\tilde{Q}$ refers to the vector of weighted solution vector jumps. This should not be confused with the vector Q which will be introduced shortly due to factorization. The equation represented above is over constrained. Thus, the choice is made to solve the equation in the least squares minimizing sense. Both sides are premultiplied by the transpose of the matrix to arrive at a square system.

$$A^T Ax = A^T b \tag{B.6}$$

While the system can be solved directly as shown above using the normal equations, this is computationally wasteful. A more efficient method based on QR factorization is used instead.

The QR factorization requires that an orthonormal basis is formed. This is accomplished with the Gram-Schmidt process. These basis vectors are

$$q_1 = \frac{M_1}{\|M_1\|} \tag{B.7}$$

$$q_2 = \frac{M_2 - (q_1^T M_2)q_1}{\|M_2 - (q_1^T M_2)q_1\|} \tag{B.8}$$

$$q_3 = \frac{M_3 - (q_1^T M_3)q_1 - (q_2^T M_3)q_2}{\|M_3 - (q_1^T M_3)q_1 - (q_2^T M_3)q_2\|} \tag{B.9}$$

Using the following definitions to simplify the derivation

$$r_{11} = \|M_1\|^2 \tag{B.10}$$

$$r_{22} = \|M_2 - (q_1^T M_2)q_1\|^2 \tag{B.11}$$

$$r_{33} = \|M_3 - (q_1^T M_3)q_1 - (q_2^T M_3)q_2\|^2 \tag{B.12}$$

$$r_{12} = M_1^T M_2 \tag{B.13}$$

$$r_{13} = M_1^T M_3 \tag{B.14}$$

$$r_{23} = \left(M_2 - \frac{r_{12}}{r_{11}} M_1 \right)^T M_3 \tag{B.15}$$

These definitions are substituted back into the basis derived earlier

$$q_1 = \frac{M_1}{\sqrt{r_{11}}} \tag{B.16}$$

$$q_2 = \frac{M_2 - \frac{r_{12}}{r_{11}} M_1}{\sqrt{r_{22}}} \tag{B.17}$$

$$q_3 = \frac{M_3 - \frac{r_{13}}{r_{22}} M_1 - \frac{r_{23}}{r_{22}} (M_2 - \frac{r_{12}}{r_{11}} M_1)}{\sqrt{r_{33}}} \tag{B.18}$$

These factors are then used to compute a QR factorization. Consider a decomposition of a matrix, A, where

$$A = QR \tag{B.19}$$

where Q is a matrix with orthonormal columns and R is upper triangular. A set of orthogonal vectors is given above for the non-square matrix A which defines the least squares gradient at a vertex, i. Clearly the q vectors above satisfy part of the factorization. The vectors M can now be written in terms of the orthonormal basis.

$$M_1 = (q_1^T M_1) q_1 \tag{B.20}$$

$$M_2 = (q_1^T M_2) q_1 + (q_2^T M_2) q_2 \tag{B.21}$$

$$M_3 = (q_1^T M_3) q_1 + (q_2^T M_3) q_2 + (q_3^T M_3) q_3 \tag{B.22}$$

These equations can be written in terms of a linear system which gives

$$\begin{bmatrix} M_1 & M_2 & M_3 \end{bmatrix} \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} = \begin{bmatrix} q_1^T M_1 & q_1^T M_2 & q_1^T M_3 \\ 0 & q_2^T M_2 & q_2^T M_3 \\ 0 & 0 & q_3^T M_3 \end{bmatrix} \tag{B.23}$$

which represents $A = QR$. This system is easily solved by

$$QR\,x = b \tag{B.24}$$

$$Q^T QR\,x = Q^T b \tag{B.25}$$

$$\tag{B.26}$$

The orthonormal property of Q gives $Q^T Q = I$ which results in

$$R\,x = Q^T b \tag{B.27}$$

The solution to the above system is simple due to the upper triangular nature of R. The solution x is found by back substitution. Make the substitution of the above definitions into Equation B.3 to arrive at

$$\begin{bmatrix} q_1^T M_1 & q_1^T M_2 & q_1^T M_3 \\ 0 & q_2^T M_2 & q_2^T M_3 \\ 0 & 0 & q_3^T M_3 \end{bmatrix} \begin{bmatrix} Q_{xi} \\ Q_{yi} \\ Q_{zi} \end{bmatrix} = \begin{bmatrix} q_1^T \tilde{Q} \\ q_2^T \tilde{Q} \\ q_3^T \tilde{Q} \end{bmatrix} \tag{B.28}$$

Define for convenience

$$r'_{22} = q_2^T M_2 \sqrt{r_{22}} \tag{B.29}$$

$$r'_{33} = q_3^T M_3 \sqrt{r_{33}} \tag{B.30}$$

Using the definitions made in the previous section, the following simplification can be made

$$\begin{bmatrix} \frac{r_{11}}{\sqrt{r_{11}}} & \frac{r_{12}}{\sqrt{r_{11}}} & \frac{r_{13}}{\sqrt{r_{11}}} \\ 0 & \frac{r'_{22}}{\sqrt{r_{22}}} & \frac{r_{23}}{\sqrt{r_{22}}} \\ 0 & 0 & \frac{r'_{33}}{\sqrt{r_{33}}} \end{bmatrix} \begin{bmatrix} Q_{xi} \\ Q_{yi} \\ Q_{zi} \end{bmatrix} = \begin{bmatrix} q_1^T \tilde{Q} \\ q_2^T \tilde{Q} \\ q_3^T \tilde{Q} \end{bmatrix} \tag{B.31}$$

Now, back substitution is used to solve the QR factorization of the least squares system explicitly

$$Q_{zi} = \frac{\left[M_3 - \frac{r_{13}}{r_{11}} M_1 \frac{r_{23}}{r_{22}} \left(M_2 - \frac{r_{12}}{r_{11}} M_1\right)\right]^T \tilde{Q}}{r'_{33}} = (W_z)^T \tilde{Q} \tag{B.32}$$

$$Q_{yi} = \frac{\left[M_2 - \frac{r_{12}}{r_{11}} M_1 - r_{23} W_z\right]^T \tilde{Q}}{r'_{22}} = (W_y)^T \tilde{Q} \tag{B.33}$$

$$Q_{zi} = \frac{\left[M_1 - r_{12} W_y - r_{13} W_z\right]^T \tilde{Q}}{r_{11}} = (W_x)^T \tilde{Q} \tag{B.34}$$

Finally, the gradient at a node i can be written as

$$Q_{xi} = \sum_{j}^{nedges} = W_{x,j}(w_1(Q_j - Q_i)) \tag{B.35}$$

$$Q_{yi} = \sum_{j}^{nedges} = W_{y,j}(w_2(Q_j - Q_i)) \tag{B.36}$$

$$Q_{zi} = \sum_{j}^{nedges} = W_{z,j}(w_3(Q_j - Q_i)) \tag{B.37}$$

The above equations can once again be expanded in terms of the edge geometry and solution jumps only. This expansion is

$$W_{z,j} = \frac{1}{r'_{33}} \left[\bar{\Delta} z_j - \frac{r_{13}}{r_{11}} \bar{\Delta} x_j - \frac{r_{23}}{r_{22}} \left(\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j\right)\right] \tag{B.38}$$

$$W_{y,j} = \frac{1}{r'_{22}} \left[\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j - r_{23} W_{z,j} \right] \tag{B.39}$$

$$W_{x,j} = \frac{1}{r_{11}} \left[\bar{\Delta} x_j - r_{12} W_{y,j} - r_{13} W_{z,j} \right] \tag{B.40}$$

where the following definitions have been expanded as well

$$r_{11} = \sum_{j}^{nedges} (\bar{\Delta} x_j)^2 \tag{B.41}$$

$$r_{22} = \sum_{j}^{nedges} \left[\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j \right]^2 \tag{B.42}$$

$$r_{13} = \sum_{j}^{nedges} \bar{\Delta} x_j \bar{\Delta} y_j \tag{B.43}$$

$$r_{23} = \sum_{j}^{nedges} \left[\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j \right] \bar{\Delta} z_j \tag{B.44}$$

$$r_{33} = \sum_{j}^{nedges} \left[\bar{\Delta} z_j - \frac{r_{13}}{r_{11}} \bar{\Delta} x_j - \frac{r_{23}}{r_{22}} \left(\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j \right) \right]^2 \tag{B.45}$$

$$r'_{22} = \sum_{j}^{nedges} \left[\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j \right] \bar{\Delta} y_j \tag{B.46}$$

$$r'_{33} = \sum_{j}^{nedges} \left[\bar{\Delta} z_j - \frac{r_{13}}{r_{11}} \bar{\Delta} x_j - \frac{r_{23}}{r_{22}} \left(\bar{\Delta} y_j - \frac{r_{12}}{r_{11}} \bar{\Delta} x_j \right) \right] \bar{\Delta} z_j \tag{B.47}$$

where

$$\bar{\Delta} x_j = w_j(\Delta x_j) = w_j(x_j - x_i) \tag{B.48}$$

$$\bar{\Delta} y_j = w_j(\Delta y_j) = w_j(y_j - y_i) \tag{B.49}$$

$$\bar{\Delta} z_j = w_j(\Delta z_j) = w_j(z_j - y_i) \tag{B.50}$$

Note that the least squares gradient coefficients given above are in terms of edges, or control faces, and would require that all of the r coefficients are on an edge by edge basis. This requires an immense amount of storage. In comparison, if the coefficients could be stored via the control volume, or vertex, the storage requirements would decrease by 5 times the number of connecting edges in a mesh on average. The control volume based coefficients are

$$s_{11} = \sum_{j}^{nedges} (\Delta \bar{x}_j)^2 \tag{B.51}$$

$$s_{12} = \sum_{j}^{nedges} \Delta \bar{x}_j \Delta \bar{y}_j \tag{B.52}$$

$$s_{13} = \sum_{j}^{nedges} \Delta \bar{x}_j \Delta \bar{z}_j \tag{B.53}$$

$$s_{22} = \sum_{j}^{nedges} (\Delta \bar{y})^2 \tag{B.54}$$

$$s_{23} = \sum_{j}^{nedges} \Delta \bar{y}_j \Delta \bar{z}_j \tag{B.55}$$

$$s_{33} = \sum_{j}^{nedges} (\Delta \bar{z}_j)^2 \tag{B.56}$$

The edge quantities can then be reconstructed as

$$r_{11} = s_{11} \tag{B.57}$$

$$r_{12} = s_{12} \tag{B.58}$$

$$r_{13} = s_{13} \tag{B.59}$$

$$r_{22} = s_{22} - \frac{r_{12}^2}{r_{11}} \tag{B.60}$$

$$r_{23} = s_{23} - \frac{r_{12}}{r_{11}} r_{13} \tag{B.61}$$

$$r_{33} = s_{33} - \frac{r_{13}^2}{r_{11}} - \frac{r_{23}^2}{r_{22}} \tag{B.62}$$

$$r'_{22} = r_{22} \tag{B.63}$$

$$r'_{33} = r_{33} \tag{B.64}$$

APPENDIX C

QUADRATURE FOR 2D GAUSSIAN

The quadrature used in computing density contributions from a two-dimensional Gaussian plume source are outlined below. The quadrature rules are applied as projections such that vertical surfaces receive no contributions and therefore, mass is only injected from a single plane.

The basis functions for a linear triangle are

$$N_1(\zeta, \eta) = 1 - \zeta - \eta$$

$$N_2(\zeta, \eta) = \zeta$$

$$N_3(\zeta, \eta) = \eta$$

These define a linear mapping from real space to a normalized space shown in Figure C.1 where quadratures are evaluated.

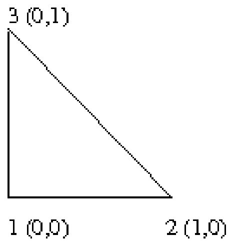


Figure C.1 Illustration of standard triangle

This transformation is represented by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{pmatrix} \begin{pmatrix} N_1 \\ N_2 \\ N_3 \end{pmatrix} \tag{C.1}$$

The quadrature points and weight associated with a fourth order triangle are shown are given in Table C.1.

Table C.1 Quadrature points and weights for a fourth order triangle

	ζ	η	W
1	0.091576213509771	0.091576213509771	0.109951743655322
2	0.445948490915965	0.108103018168070	0.223381589678011
3	0.816847572980459	0.091576213509771	0.109951743655322
4	0.108103018168070	0.445948490915965	0.223381589678011
5	0.445948490915965	0.445948490915965	0.223381589648011
6	0.091576213509771	0.816847572980459	0.109951743655322

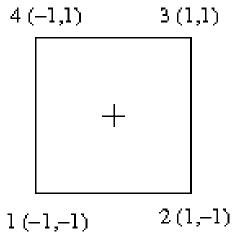


Figure C.2 Illustration of standard quadrilateral

Likewise, the basis functions for a linear quadrilateral are

$$N_1(\zeta,\eta) = \frac{1}{4}(1-\zeta)(1-\eta)$$

$$N_2(\zeta,\eta) = \frac{1}{4}(1+\zeta)(1-\eta)$$

$$N_3(\zeta,\eta) = \frac{1}{4}(1+\zeta)(1+\eta)$$

$$N_4(\zeta,\eta) = \frac{1}{4}(1-\zeta)(1+\eta)$$

$$N_k(\zeta,\eta) = \frac{1}{4}(1+\zeta_k\zeta)(1+\eta_k\eta)$$

Again, the basis functions define a linear mapping from real space to normalized space shown in Figure C.2 to simplify quadrature evaluation. This transformation is

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{pmatrix} \begin{pmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{pmatrix} \tag{C.2}$$

The quadrature points and weight associated with a fourth order quadrilateral are found in Table C.2.

Table C.2 Quadrature points and weights for a fourth order quadrilateral

i,j	ψ	W
1	-0.8611363115940526	0.34785484513745386
2	-0.3399810435848563	0.65214515486254614
3	0.3399810435848563	0.65214515486254614
4	0.8611363115940526	0.34785484513745386

The location of each of the sixteen Gauss points is given by

$$(\zeta_i, \eta_j) = (\psi_i, \psi_j) \tag{C.3}$$

and the quadrature weight is given by

$$W(\zeta_i, \eta_j) = W_i \, W_j \tag{C.4}$$

These basis functions permit an isoparametric mapping, which allow for the evaluation of the Gaussian plume in normalized space. That is

$$x = \sum_{i=1}^{nbasis} N_i(\zeta, \eta)\zeta_i \tag{C.5}$$

$$y = \sum_{i=1}^{nbasis} N_i(\zeta, \eta)\eta_i \tag{C.6}$$

where (ζ_i, η_i) are the Gauss points in normalized space. By substitution a Gaussian source term in normalized space can be written as

$$g(\zeta, \eta) = A\, exp\left(- \left(\frac{\left(\left[\sum_{i=1}^{nbasis} N_i(\zeta, \eta)\zeta_i\right] - x_0\right)^2}{2\sigma_x^2} + \frac{\left(\left[\sum_{i=1}^{nbasis} N_i(\zeta, \eta)\eta_i\right] - y_0\right)^2}{2\sigma_y^2} \right)\right) \tag{C.7}$$

Therefore, the integrated value of the plume source Gaussian is

$$\int_\Omega g(x, y) d\Omega = \sum_{k}^{ngauss} \sum_{j}^{nbasis} W_k\ g(\zeta_k, \eta_k)\ N_j \tag{C.8}$$

where W_k are the quadrature weights associated with each Gauss point i.

VITA

Nicholas Graham Currier was born in Merced, California, on July 27, 1985. He graduated Van Buren County High School in 2004 after beginning full-time studies at Tennessee Technological University in 2003. During his time at TTU, he was a member of Pi Tau Sigma (mechanical engineering honor society), Mortar Board, ASME, and the university's Formula SAE team. He graduated Cum Laude with a Bachelor of Science degree in Mechanical Engineering in December 2007 and joined the University of Tennessee SimCenter as a graduate research assistant shortly thereafter. Nicholas earned an M.S. in Computational Engineering from UTC in August 2010.